

# Rechnernetze

## Kapitel 3: Link Layer

Prof. Dr. Wolfgang Mühlbauer

Fakultät für Informatik

`wolfgang.muehlbauer@th-rosenheim.de`

Wintersemester 2019/2020

Slides are based on:

J. Kurose, K. Ross: Computer Networks – A Top-Down Approach

A. Tanenbaum, D. Wetherall: Computer Networks

- ❑ **Einführung**
- ❑ Rahmenbildung, Fehlererkennung
- ❑ Ethernet 802.3
- ❑ Mehrpunktverbindungen, Vielfachzugriff
- ❑ Punkt-zu-Punkt Verbindungen in „fully switched networks“

# Terminologie

## □ Ende-zu-Ende Pfad:

- Besteht aus vielen, **heterogenen** Links.
- Beispiel: Von HTTP Client zu Webserver über WLAN, Ethernet und Mobilfunknetz.

## □ Hosts und Nodes

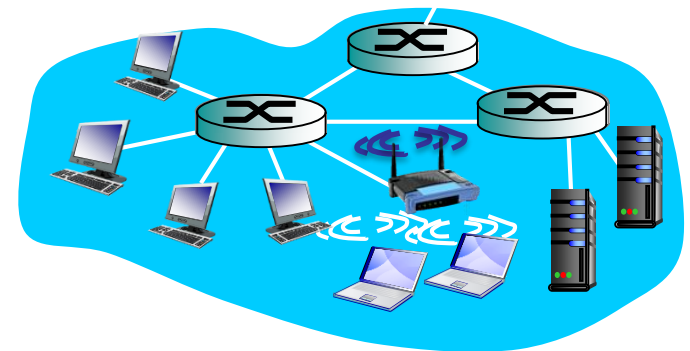
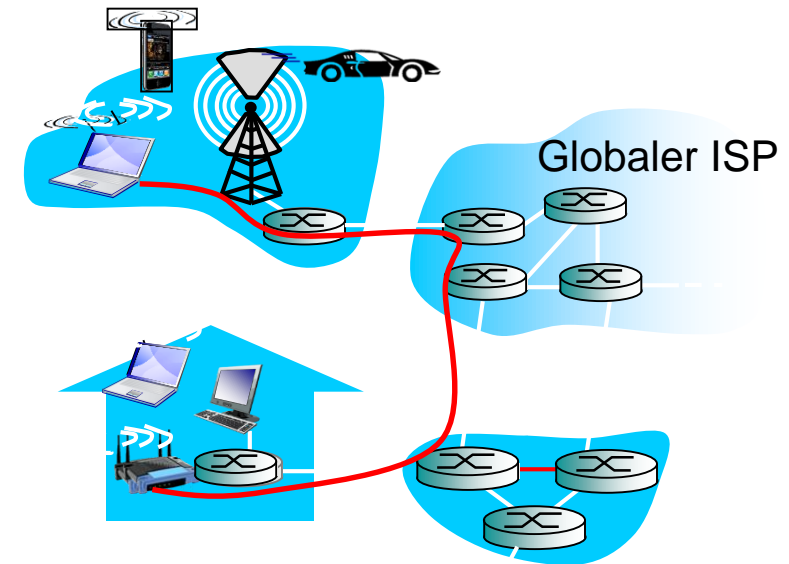
- *Host* == Endpunkt eines Ende-zu-Ende Pfades
- *Node* == Jedes Gerät, das am Netzwerk teilnimmt also Host, Router, Switch, Access Point, usw.

## □ Link

- Verbindet benachbarte "Nodes"
- WLAN, Ethernet, Mobilfunk, (Bluetooth), usw.

## □ Frame

- Nachricht auf Schicht 2
- Frame ist "Briefumschlag" für Schicht 3 Paket.



Link (Schicht 2)

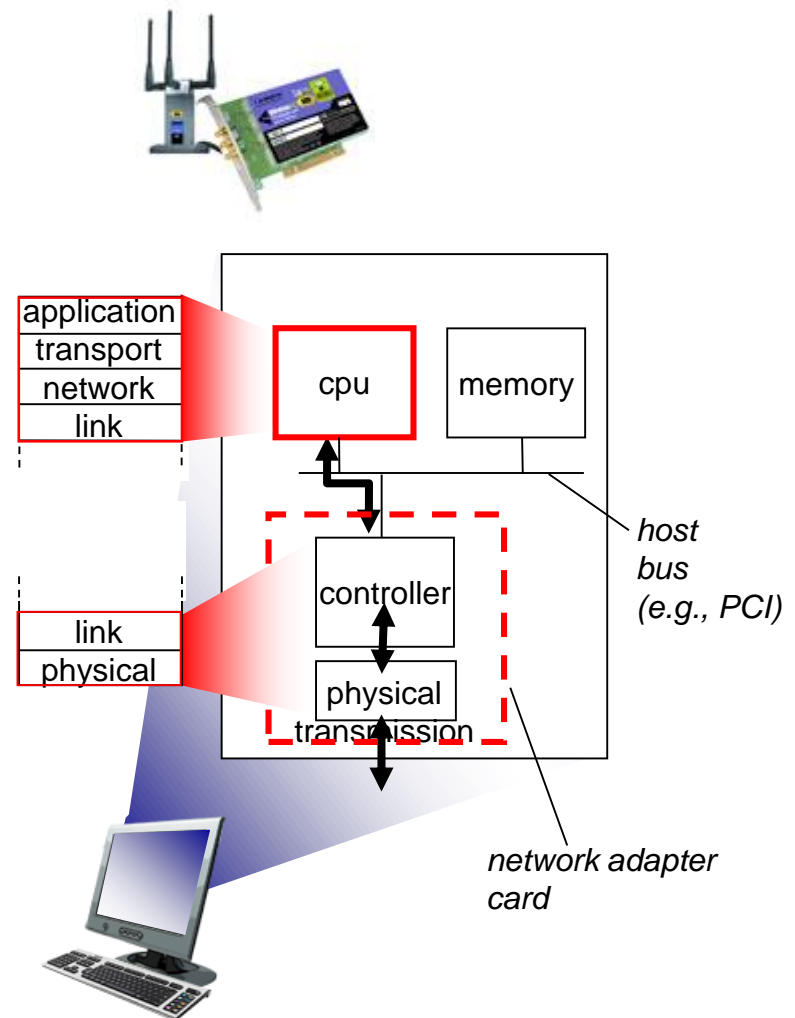
— Ende-zu-Ende Pfad (Schicht 3)

# Dienste der Link Layer (dt. "Sicherheitsschicht")

- ❑ **Übertragung von Frames zwischen benachbarten Nodes**
- ❑ **Rahmenbildung (engl. Framing)**
  - Positionsrichtige Erkennung von Zeichen, Erkennung von Blockbegrenzungen.
  - *Frame == Header + Payload*
  - Hier Payload == IP Paket
- ❑ **Vielfachzugriff:** Wer darf wann das Medium nutzen?
  - Nötig, falls Mehrpunktmedium
  - Beispiele: WLAN, Satellitennetze, Zugangsnetz bei Kabelanschluss
- ❑ **Fehlererkennung und -korrektur**
  - Umgang mit Bitfehlern auf der Physical Layer.
  - Hinzufügen von Redundanz, um Fehler zu erkennen bzw. zu korrigieren.
- ❑ **Zuverlässige Datenübertragung ("Reliable Data Delivery")**
  - Korrektur von Paketverlusten, korrekten Reihenfolge, Vermeidung von Duplikaten
  - Bei WLAN teilweise, bei Ethernet gar nicht!

# Wo implementiert man die Link Layer?

- ❑ In **allen** Nodes
  - Auch Router und Switches!
  - Nicht in Hubs!
- ❑ Implementierung der meisten Funktionalität in Hardware
  - Fehlererkennung
  - Rahmenbildung
  - ...
- ❑ **Network Interface Card (NIC)** oder Netzwerkkarte
  - Implementiert große Teile der Link Layer und der Physical Layer (Leitungscode, etc.).
  - Über Bus mit CPU verbunden.



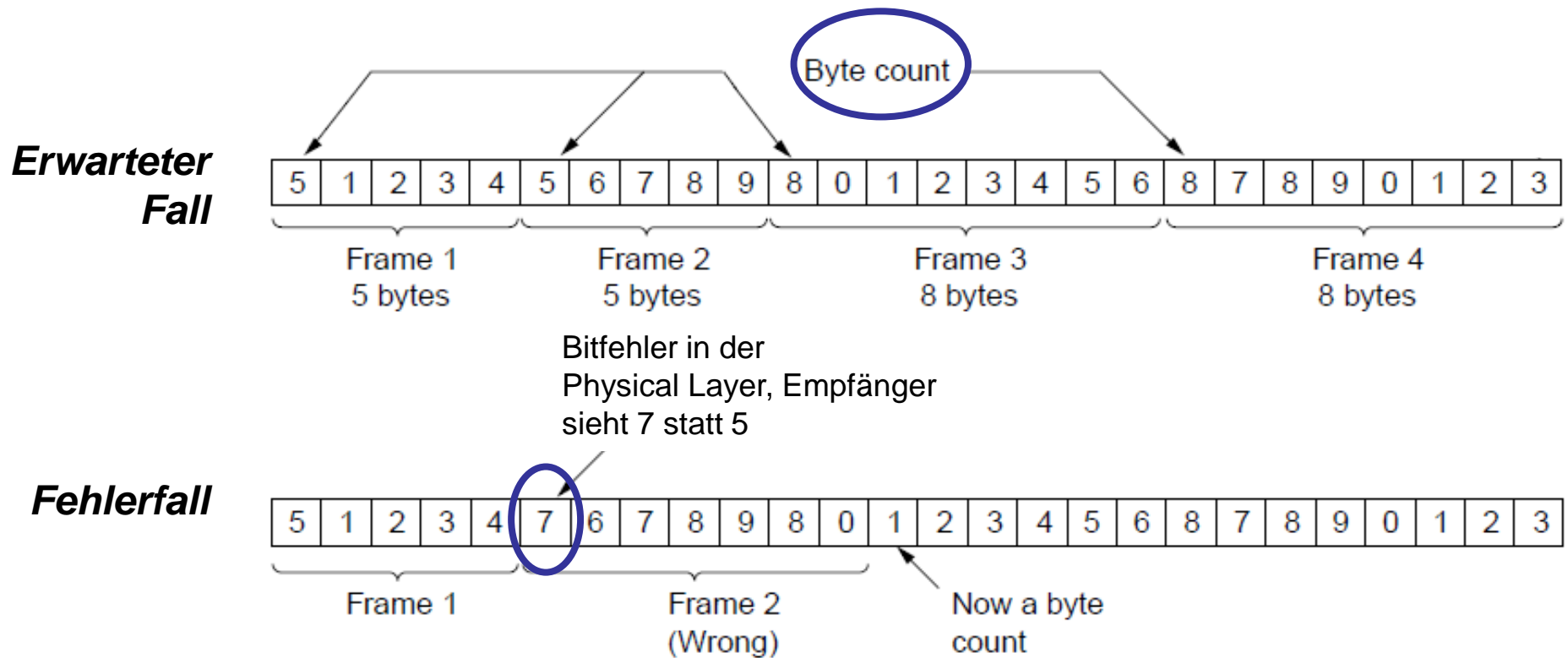
- ❑ Einführung
- ❑ **Rahmenbildung, Fehlererkennung**
- ❑ Ethernet 802.3
- ❑ Mehrpunktverbindungen und Vielfachzugriff
- ❑ Punkt-zu-Punkt Verbindungen in „fully switched networks“

# Rahmenbildung (engl. Framing)

- ❑ Physical Layer empfängt und sendet **Bitstrom**.
- ❑ Fehlerbehandlung durch Link Layer nur möglich
  - falls Bits in endliche Sequenzen (= **Frame**) zerlegt werden.
  - Frame Redundanz hat (z.B. Checksum), siehe nächster Abschnitt.
- ❑ **Probleme**
  - Wie erkennt Empfänger Frameanfang und –ende aus Bitstrom?
  - Wie überträgt man beliebige Bit- und Zeichenkombinationen?
- ❑ **Lösungsansätze**
  - *Byte Count*
  - *Byte Stuffing*
  - *Bit Stuffing*
  - *Coderegolverletzungen*: Blockbegrenzung durch Verwendung ungültiger Codes in Physical Layer

# Byte Count: Längenangabe der Nutzdaten

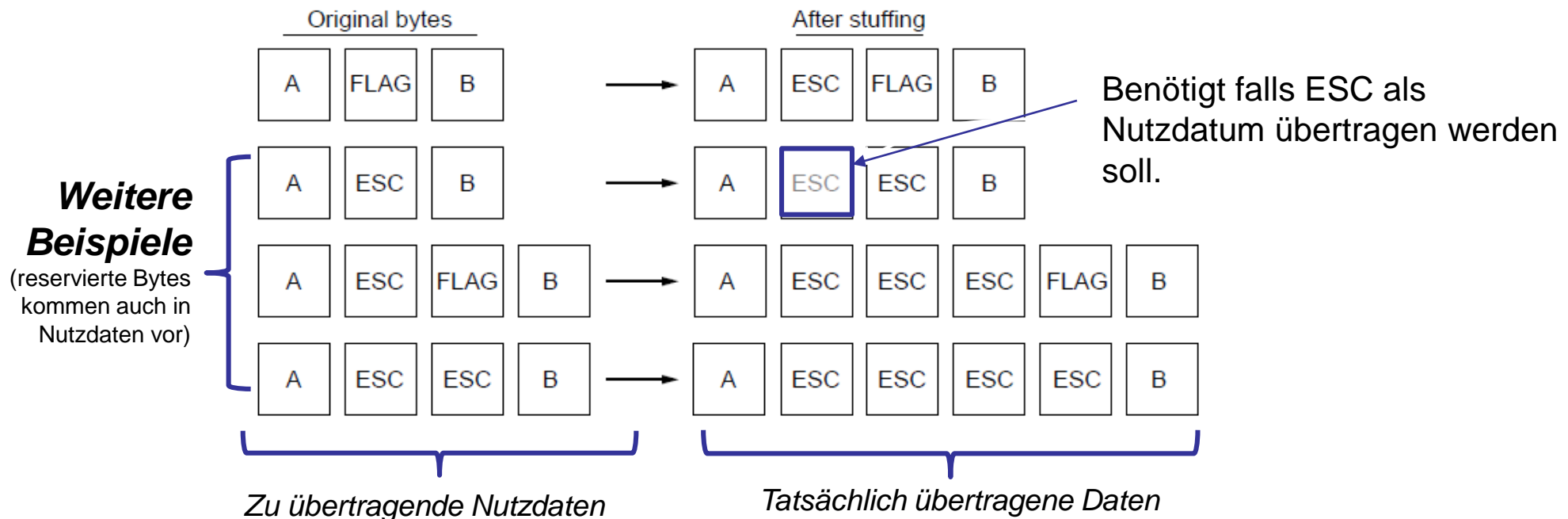
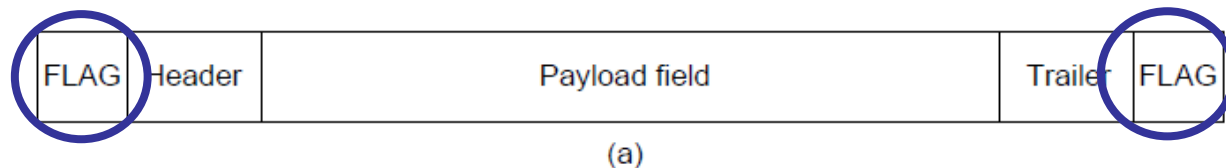
- ❑ Jeder Frame beginnt mit Feld, das Anzahl der enthaltenen Bytes angibt.
- ❑ Nachteil: Erneute Synchronisation nach Fehler schwierig bzw. unmöglich!





# Byte Stuffing: Steuerzeichen und Zeichenstopfen

- ❑ Reserviertes Byte **FLAG** markiert **Frameanfang und -ende**
- ❑ Mögliches Problem: FLAG kommt in Nutzdaten vor
  - Ausweg: Verwenden eines weiteren reservierten Bytes **ESC** (=Escape)
- ❑ Einfache Synchronisation nach Fehler, aber Overhead!



# Bit Stuffing: Begrenzungsfeld und Bitstopfen

- ❑ Vorteil: Framelänge muss **kein Vielfaches** von 8 Bit sein!
- ❑ Jeder Frame beginnt mit **speziellem, reservierten** Bitmuster:
  - Hier im Beispiel: **01111110**
- ❑ **Regeln**
  - Beim **Senden** : Nach 5 1er-Bits wird **immer** ein 0-Bit eingefügt.
  - Beim **Empfang**: Nach 5 1er Bits wird **immer** ein 0-Bit gelöscht.

[illegible]

# Übung: Bit Stuffing

---

- ❑ Wie lautet die Bitsequenz **nach** Bit-Stuffing?
  - 01000111 11100011 11100000 01111110

# Publikums-Joker: Link Layer

2 Network-Layer Pakete (z.B. IP Pakete) seien gleich groß aber inhaltlich unterschiedlich. Auf der Link-Layer wird Bit Stuffing verwendet.

Welche Aussage ist **falsch**?

- A. Bit Stuffing ist auf der Netzwerkkarte (NIC) implementiert.
- B. Die dazugehörigen Frames haben verschiedene Checksums.
- C. Das Propagation Delay beim Senden ist für beide Pakete gleich groß.
- D. Beim Senden der IP Pakete sind die dazugehörigen Frames gleich groß.



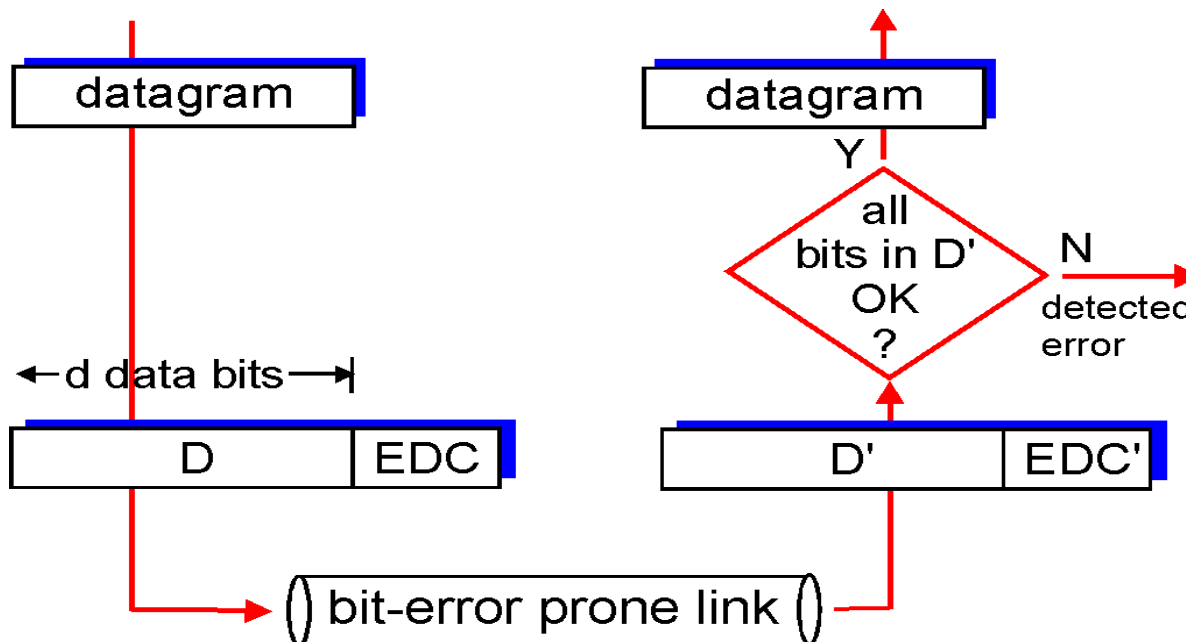
# Umgang mit Bitfehlern

- ❑ ***Ursachen für Bitfehler:*** Rauschen, Dämpfung, Verzerrung, usw.
- ❑ ***Grundidee***
  - Rahmenbildung (engl. „Framing“)
  - Redundanz (z.B. Prüfsumme über Frames)
- ❑ ***Fehlerkorrektur*** durch Redundanz
  - Benötigt viel Redundanz
  - Üblich bei nicht "wiederholbaren" Medien (CD, RAM, etc.), nicht bei TCP/IP
- ❑ ***Fehlererkennung*** durch Redundanz
  - Fehler wird nur erkannt, aber nicht behoben.
  - Maßnahmen:
    - **Ethernet 802.3:** Keine Retransmission. Wiederanforderung des fehlerhaften Blocks möglicherweise nur durch TCP, falls Timeout eintritt.
    - **WLAN 802.11:** Aktive Wiederanforderung des fehlerhaften Blocks durch Link Layer (=Active Repeat Request)

# Allgemeiner Ansatz

## ❑ Bezeichner

- **EDC**: Error Detection und Correction Bits
- **D**: Nutzdaten, die durch Fehlerbehandlung abgesichert werden



## Ansätze

- ❑ Paritätsbits
- ❑ *Checksumme*
  - IP, TCP
- ❑ Cyclic Redundancy Check (CRC)
  - Ethernet, WLAN

EDC == EDC' ?

# Checksum (in IP und TCP Header)

## ❑ Idee: "Addition"

- Betrachte Bits in **Gruppen von 16-Bit Wörter**
- Summiere **alle** 16-Bit Wörter unter Berücksichtigung des Übertrags
- 1er-Komplement des Ergebnisses ist die Checksum

## ❑ Überprüfung beim Empfänger relativ einfach

- Addiere alle übertragenen Wörter **UND** Checksum
- Ergebnis muss aus lauter 1er Bits bestehen, sonst Fehler

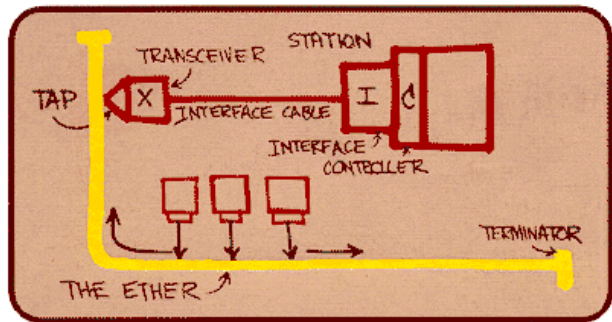
|          |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|----------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1. Wort  | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 2. Wort  | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| <hr/>    |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Übertrag | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| <hr/>    |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Summe    | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| Checksum | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |

- ❑ Einführung
- ❑ Rahmenbildung, Fehlererkennung
- ❑ **Ethernet 802.3**
- ❑ Mehrpunktverbindungen, Vielfachzugriffs
- ❑ Punkt-zu-Punkt Verbindungen in „fully switched networks“

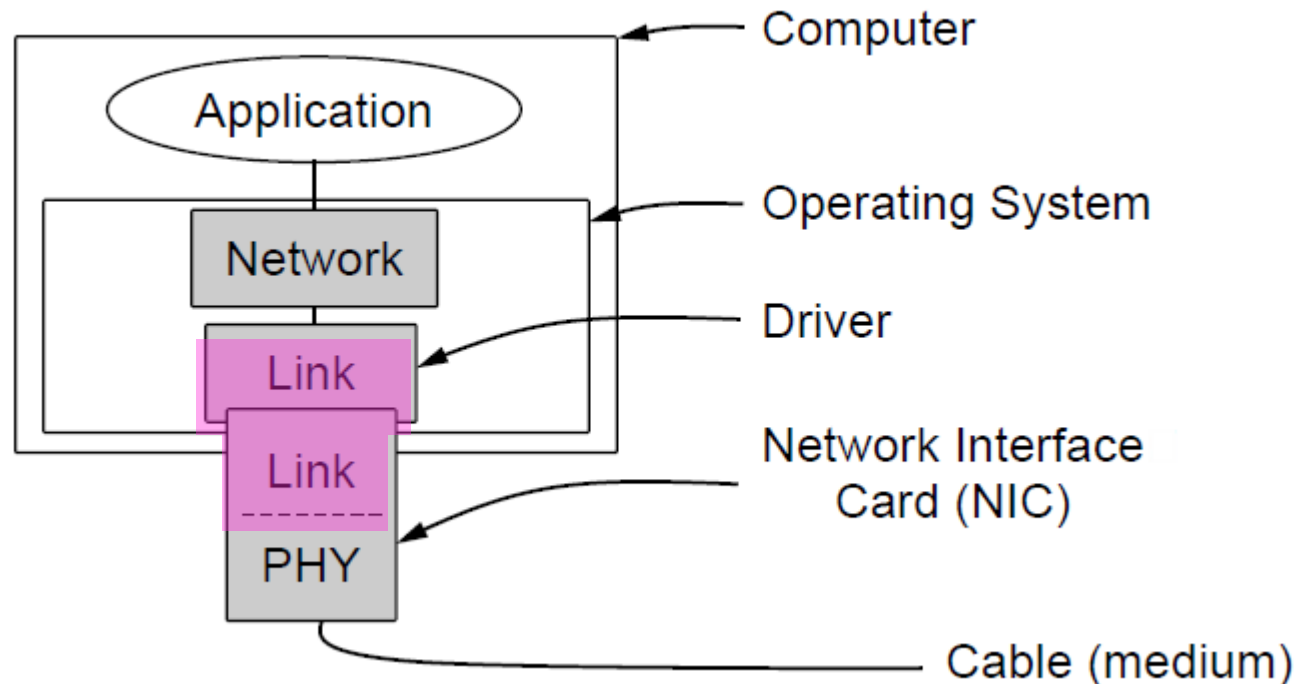


# Ethernet


- ❑ Dominierende LAN Technologie
- ❑ Netzwerkkarten sind preiswert (< 3 Euro)
- ❑ Geschwindigkeiten nahmen ständig zu: 10 Mbps – 10 Gbps



*Ethernet Schema von Metcalf*

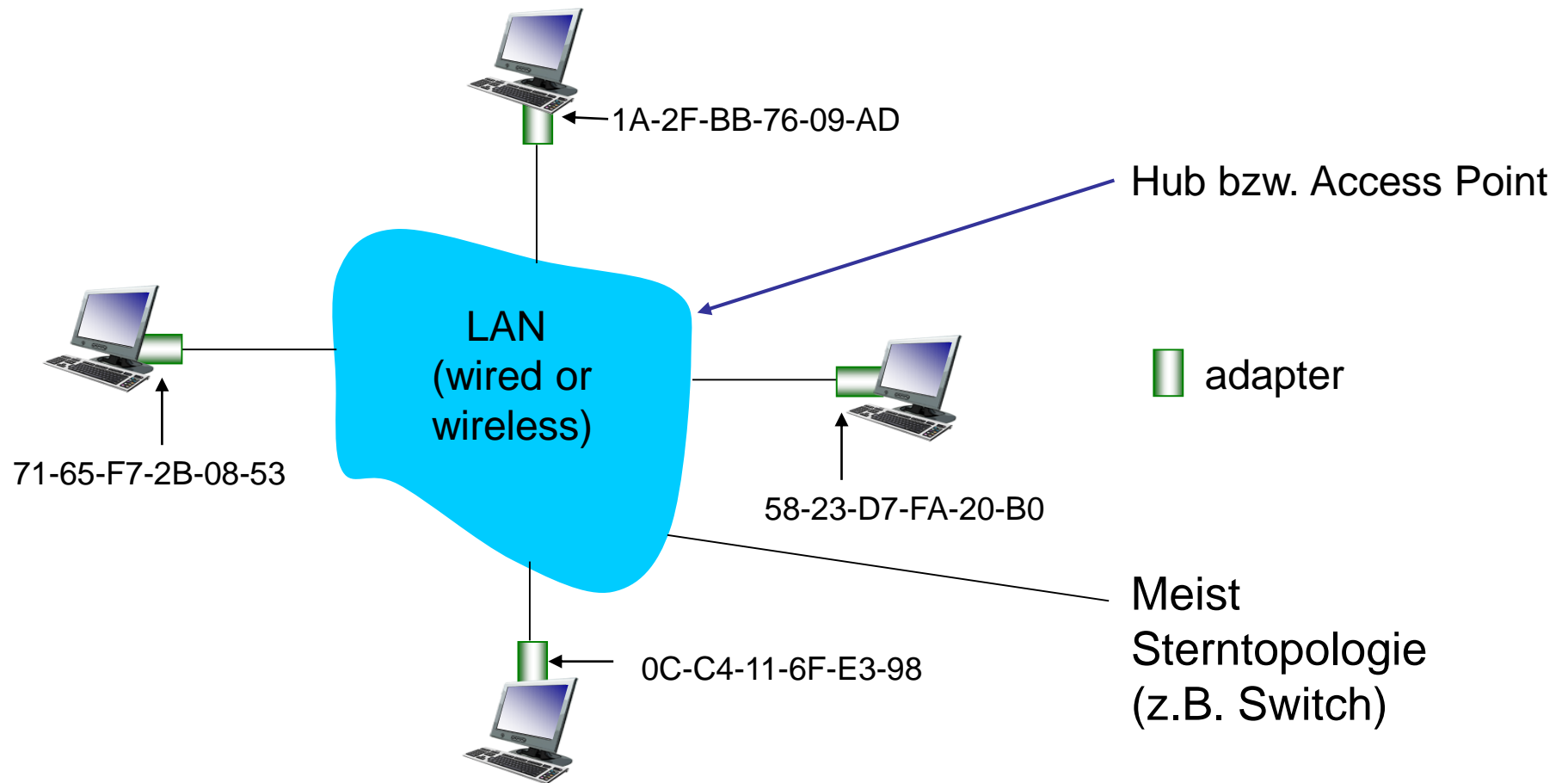


# MAC Adresse

- ❑ **Adresse** der Link Layer
  - Identifiziert Nachbarn, wichtig vor allem bei Mehrpunktverbindungen.
  - Nur lokal gültig (LAN, WLAN).
  
- ❑ Jedes Interface eines Hosts / Routers hat eigene MAC Adresse
  - Ein Gerät kann also mehrere MAC Adressen haben.
  
- ❑ Ethernet und WLAN: 48 Bit
  - Teils fest mit Netzwerkkarte verknüpft
  - Manchmal per SW änderbar
  - Beispiel: 1A-2F-BB-76-09-AD 
  - Broadcast-Adresse: FF-FF-FF-FF-FF-FF
  
- ❑ Adressen werden durch IEEE zugewiesen
  - Hersteller kaufen Adressräume

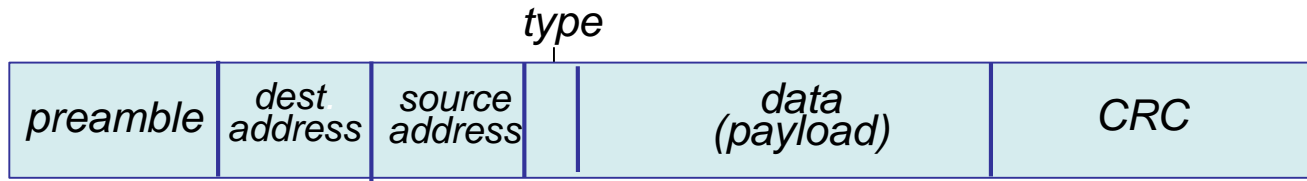
Hexadezimal  
(jede Nummer  
repräsentiert 4 Bits)

# LAN Adressen



Jede Netzwerkkarte muss **eindeutige** MAC Adresse haben!

# Ethernet 802.3 Frames



## ❑ **Präambel**

- Zu Beginn: 7mal 10101010, dann 1mal 10101011
- Synchronisation von Sender- und Empfänger, Start des Frames.

## ❑ **Adressen**

- Jeweils 6 Byte Sende- und Empfänger MAC Adresse
- *In der Regel:* Netzwerkkarte leitet empfangenen Frame nur an Betriebssystem weiter, falls Dest. MAC der eigenen MAC entspricht. Ausnahmen:
  - Dest. MAC ist FF:FF:FF:FF:FF:FF
  - *Promiscuous Mode*

## ❑ **Type**

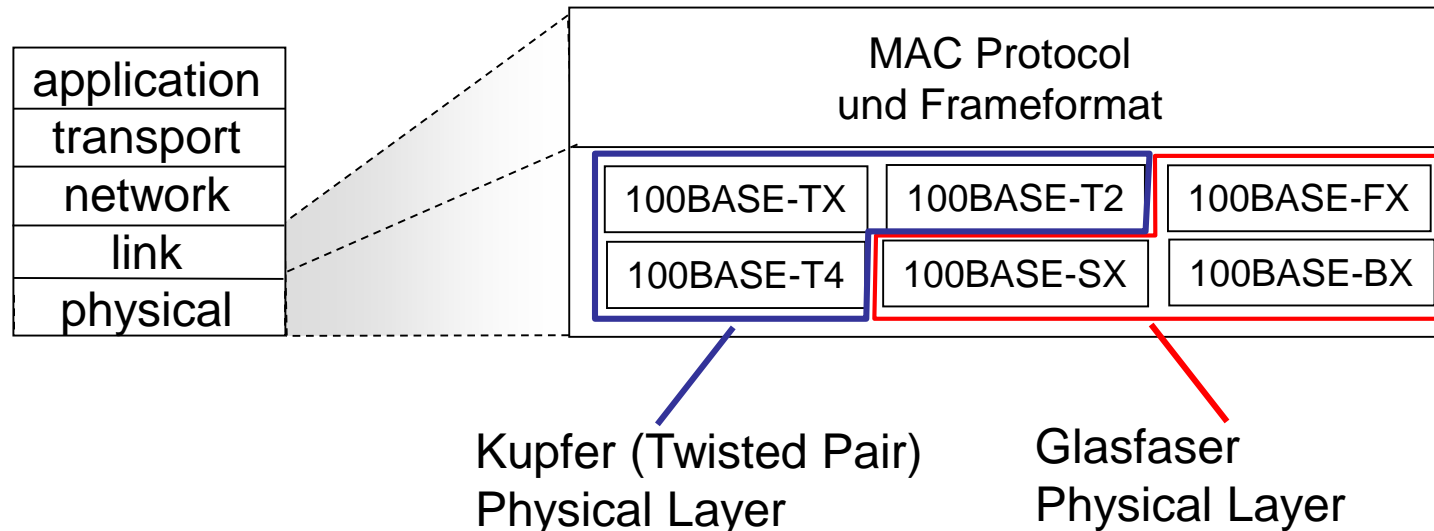
- 2 Byte → spezifiziert Art des Netzwerkprotokolls
- IPv4=0x0800, IPv6=0x86DD, ...

## ❑ **CRC**

- 4 Byte

# Ethernet 802.3: Eigenschaften

- ❑ **Verbindungslos**
  - Kein Verbindungsaufbau vor Datenaustausch
- ❑ **Keine zuverlässige (reliable) Verbindung**
  - Verlust von Frames möglich
  - Absicherung muss durch höhere Schichten erfolgen.
- ❑ **Vielfachzugriff**
  - Nur bei Broadcast: Unslotted CSMA/CD mit Binary Backoff, siehe später.
- ❑ **Unterstützt verschiedene Übertragungsmedien**
  - Beispiel: 100BASE-SX, 100BaseTX



# Publikums-Joker: MAC Adressen

Welche der folgenden Aussagen ist **falsch**?

(Annahme: alle Geräte sind ans Internet angebunden und haben ausschließlich Ethernet Interfaces)

- A. Ein klassischer Router hat mehrere MAC Adressen.
- B. Ein klassischer Switch hat mehrere MAC Adressen.
- C. Ein Host kann mehr als 1 MAC Adresse haben.
- D. Die MAC Adresse lässt sich teils leicht per SW ändern.



- ❑ Einführung
- ❑ Rahmenbildung, Fehlererkennung
- ❑ Ethernet 802.3
- ❑ **Mehrpunktverbindungen, Vielfachzugriff**
- ❑ Punkt-zu-Punkt Verbindungen in „fully switched networks“

# Zwei Arten von „Links“

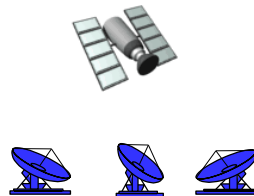
- ❑ **Link:** Kommunikation zwischen **benachbarten** Hosts, Routern und Switches
- ❑ **Punkt-zu-Punkt**
  - 2 kommunizierenden Nodes haben einen **eigenen dedizierten** Link für jede Richtung.
  - Beispiel: Ethernet LAN, das nur Switches verwendet und PPP für SONET und DSL
- ❑ **Mehrpunktverbindungen**
  - > 2 kommunizierende Nodes **teilen** sich einen Link.
  - WLAN 802.11, Bluetooth 802.15, altes Ethernet (Hub), Last Mile bei Kabelnetz



Geteilte Leitung  
Klassisches Ethernet



Geteiltes HF  
802.11



Geteiltes HF  
(Satellit)



Menschen auf einer Party  
(geteilte Musik)



# Vielfachzugriff (engl. Media Access)

## ❑ Wer darf wann senden?

## ❑ **Annahmen**

- Geteilter Broadcastkanal
- Interferenz == **Kollision** falls mehrere Stationen gleichzeitig senden.

## ❑ **Multiple Access Control**

- *Verteilter* Algorithmus, der entscheidet, wann Host senden darf
- Entscheidung muss "**inband**" getroffen werden (kein extra Kanal)

## ❑ **Anforderungen** (Annahme: Link hat Kapazität $R$ )

- Nur 1 Host möchte senden → Host sendet mit Rate  $R$ !
- $M$  Hosts senden → Jeder Host bekommt Rate  $R/M$  („*Fairness*“)
- Dezentral ohne koordinierende Station
- Einfach zu implementieren

# Klassifizierung von Multiple Access Control (MAC)

## ❑ **Multiplexverfahren**

- Zeit-, Frequenzmultiplexverfahren
- Jeder Sender darf nur zu bestimmter Zeit, mit einer bestimmten Frequenz senden.
- Bereits behandelt, siehe Physical Layer.

Im Folgenden behandelt

## ❑ **Random Access Verfahren**

- Kanal wird nicht „aufgeteilt“, Kollisionen werden zugelassen
- Mechanismen, um sich von Kollisionen zu erholen
- **Schwerpunkt dieses Abschnitts!**

## ❑ **Token-Verfahren (nicht behandelt)**

- Kollisionen werden grundsätzlich verhindert.
- Nur wer Token hat darf auf Kanal zugreifen

# Random Access Verfahren

- ❑ Kollision wird zugelassen
  - Falls  $>2$  Stationen senden, tritt **Kollision** auf
  
- ❑ **Zu lösen ist:**
  - Wie **erkennt** man Kollision?
  - Wie **reagiert** man auf eine Kollision?
    - Erneutes Übertragen solange bis Erfolg.
    - Ggfs. zufällige Wartezeit, um weitere Kollisionen zu verhindern.
  
- ❑ **Beispiele** von Random Access Verfahren
  - Slotted ALOHA, Unslotted ALOHA
  - CSMA, CSMA/CD, **CSMA/CA (WLAN)**

# Carrier Sense Multiple Access (CSMA)

- ❑ Häufig wird die Zeit in Slots unterteilt.
- ❑ **Carrier Sensing** == Mitlauschen auf dem Kanal.
  - *Falls Kanal frei*: Übertragung beginnen.
  - *Falls Kanal belegt*: Verschiebe Übertragung.
- ❑ **Kollisionen** (=2 Stationen senden gleichzeitig).
  - Sind grundsätzlich möglich.
  - Bei Erkennen von Kollisionen erneute Übertragung.
  - Wie können Kollisionen erkannt werden?
    - Durch Mitlauschen des Senders, z.B. altes Ethernet (CSMA/CD)
    - Ausbleibendes ACK signalisiert dem Sender, dass Empfänger Paket nicht erhalten hat.
- ❑ **Varianten** falls Kanal belegt:
  - *1-persistent*: Sende, sobald Kanal wieder frei wird.
  - *p-persistent*: Sende im nächsten Slot mit der Wahrscheinlichkeit  $p$  falls Kanal frei ist
  - *Non-persistent*: Warte eine zufällig Zeit und prüfe erneut ob Kanal frei ist

# CSMA/CA bei WLAN 802.11: Konzept

## ❑ **Carrier Sensing**

- Höre das Medium vor dem Senden ab.

## ❑ **Congestion Avoidance (CA)**

- Versuche Kollisionen soweit als möglich zu vermeiden.
- Dennoch: Kollisionsbehandlung notwendig.

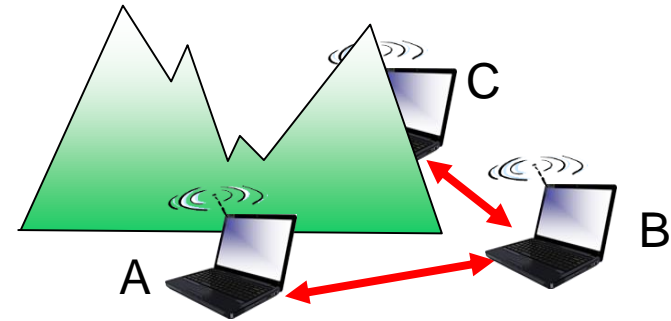
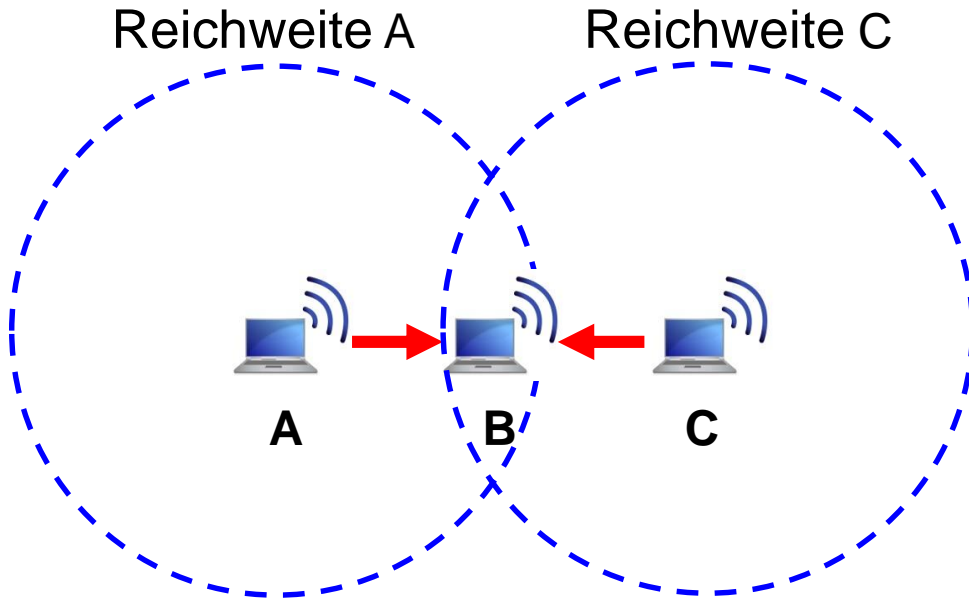
## ❑ **Binary (exponential) Backoff**

- Nach der  $m$ . Kollision, wähle zufällig ein  $K$  aus  $\{0, 1, 2, \dots, 2^m - 1\}$ .
- Warte dann  $K$  Zeitslots, bevor erneut ein Sendeversuch gestartet wird.
- Zufall hilft, eine erneute gleichzeitige Übertragung zu vermeiden.
- Längere Wartezeiten wahrscheinlich, falls hintereinander wiederholt Kollisionen auftreten.

## ❑ **Problem:** Bei WLAN ist das Erkennen von Kollisionen schwierig.

- WLAN ist **halbduplex**
  - Meist kein Mithören während des Sendens implementiert, da empfangenes Signal sehr schwach im Vergleich zu gesendetem Signal.
- WLAN Stationen können sich *nicht alle gegenseitig hören*.
  - **Hidden Station Problem** (siehe nächste Folie)

# Hidden Station Problem



- ❑ Versteckte Station: A kann Mitbewerber C nicht hören
  - A und C senden gleichzeitig → Kollision bei B
- ❑ Mögliche Ursachen:
  - C zu weit von A entfernt oder Hindernis zwischen A und C.
- ❑ Eigentlich müssen ***Kollisionen beim Empfänger erkannt werden.***

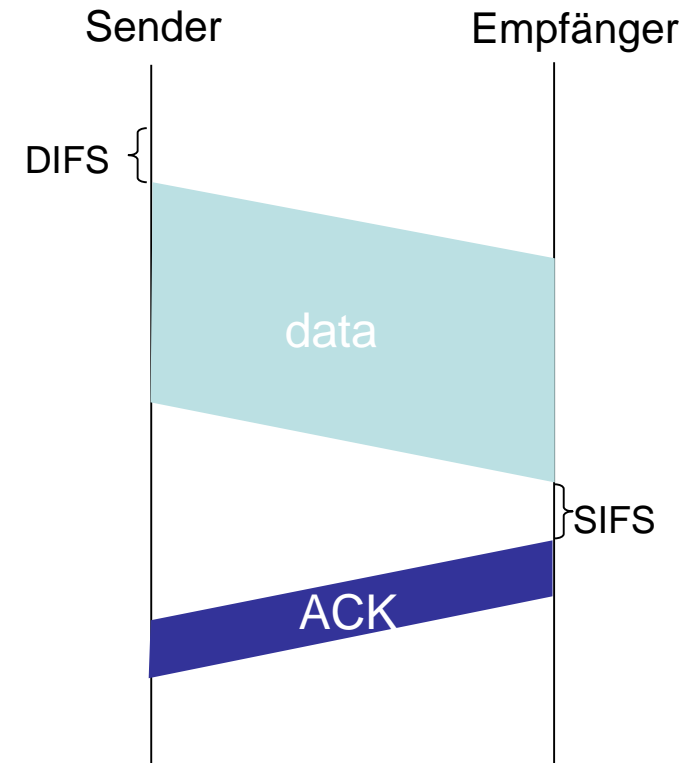
# CSMA/CA Algorithmus

## □ Sender (bei Sendewunsch)

- **Frei:** Kanal mind. für Zeitspanne DIFS frei
  - Sende *kompletten* Frame (ohne Carrier Sense)
- **Belegt:** Kanal gerade belegt.
  - Bereits hier Exponential Backoff
    - Unterschied zu CSMA/CD!
  - Höre Kanal ständig ab, **dekrementiere Timer nur** während Zeiten, in denen Kanal frei.
  - Erneute Übertragung wenn Timer ausläuft
- Falls **kein ACK** eintrifft
  - Gehe in "Belegt"-Fall
  - Vergrößere ggfs. Backoff Intervall.

## □ Empfänger

- Bestätigt Datenempfang durch ACK nach Zeitspanne SIFS (=Kollisionserkennung beim Empfänger)



**SIFS kürzer als DIFS: Priorisierung von ACKs!**

# CSMA/CA: Beispiel

- ❑ Zufällige Wartezeiten auch ohne Kollision
  - Wenn Carrier Sense bei Sendewunsch ergibt, dass Kanal gerade belegt.
- ❑ Backoff Timer zählt nur runter, wenn Kanal auch wirklich frei.
- ❑ Bleiben ACKs aus → Retransmissions (hier nicht gezeichnet)

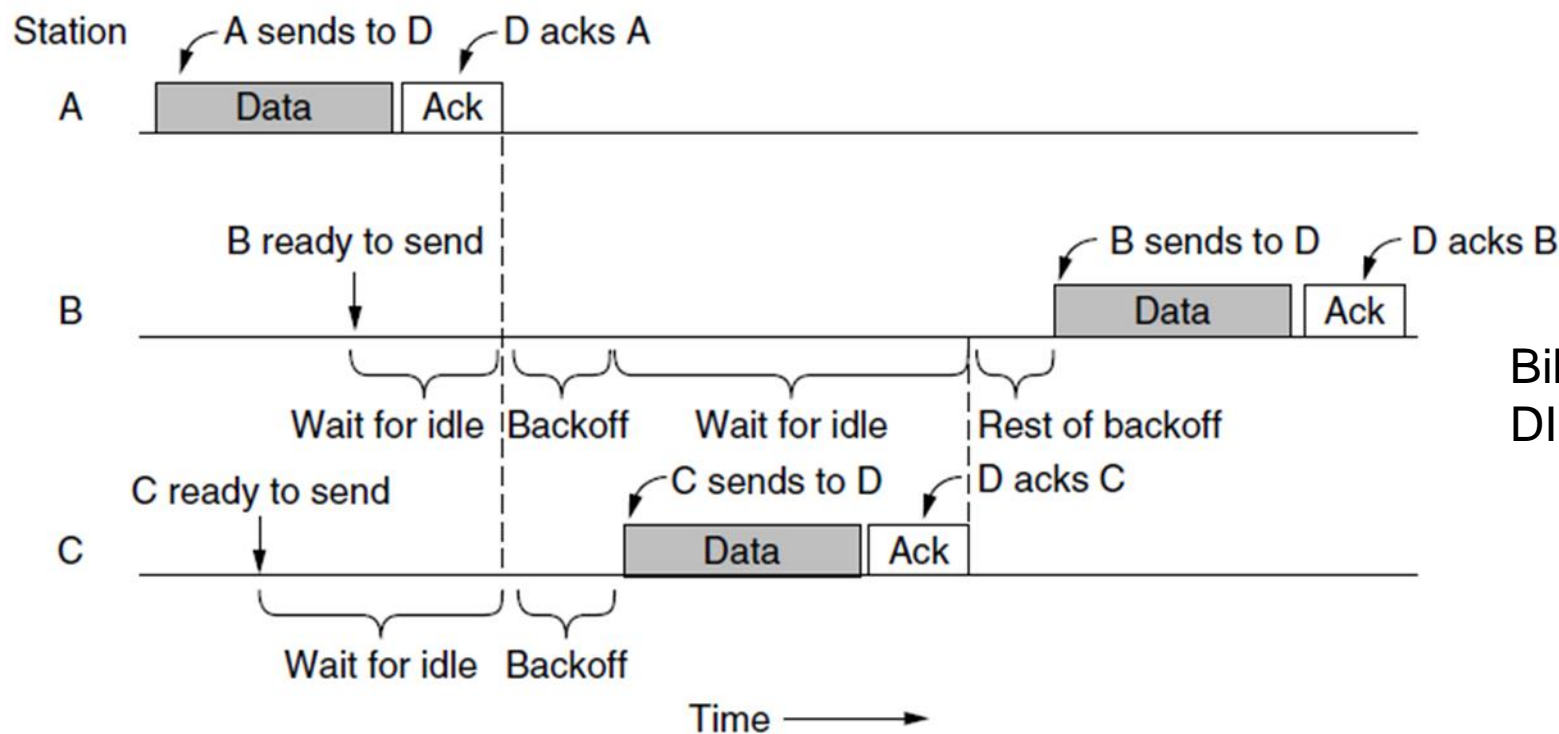


Bild ohne SIFS und  
DIFS Wartezeiten!

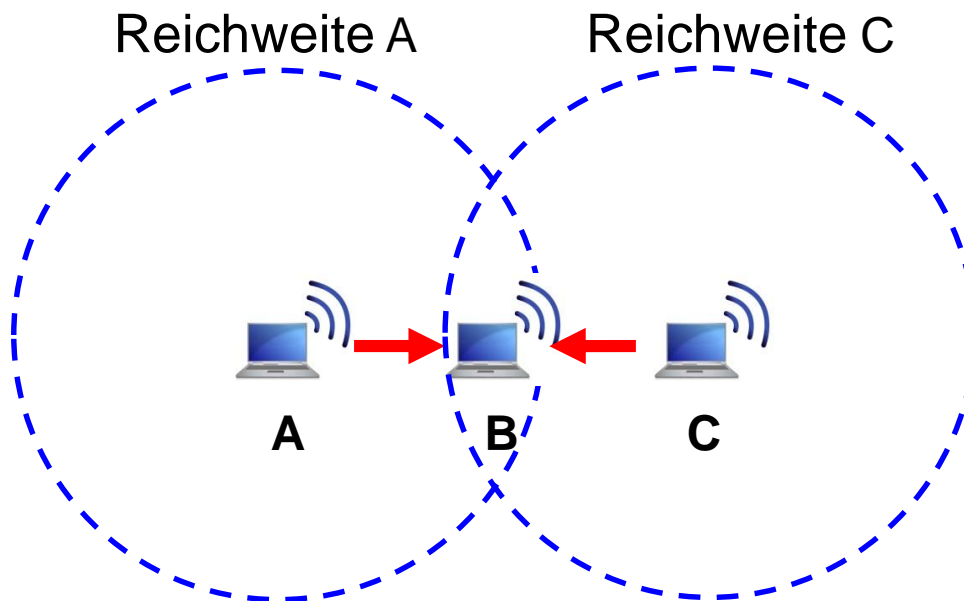
Quelle: Tanenbaum



# Publikumsjoker

## ❑ Vermeidet CSMA/CA Kollisionen im Hidden Station Problem *immer*?

- A = JA
- B = NEIN



- ❑ Einführung
- ❑ Rahmenbildung, Fehlererkennung
- ❑ Ethernet 802.3
- ❑ Mehrpunktverbindungen, Vielfachzugriff
- ❑ **Punkt-zu-Punkt Verbindungen in „fully switched networks“**

# Zwei Arten von „Links“

## □ **Punkt-zu-Punkt**

- 2 kommunizierenden Nodes haben einen **eigenen dedizierten** Link für jede Richtung.
- Beispiel: Ethernet LAN, das nur Switches verwendet und PPP für SONET und DSL

## □ **Mehrpunktverbindungen**

- > 2 kommunizierende Nodes **teilen** sich einen Link.
- WLAN 802.11, Bluetooth 802.15, Klassisches Ethernet (Hub), Last Mile bei Kabelnetz

## □ **Autonegotiation:** Ethernet Host erkennt, ob andere Hosts am Medium sind

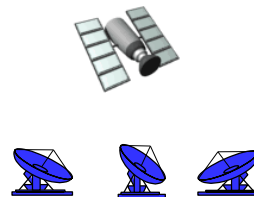
- Weitere Infos (hier nicht behandelt): <https://de.wikipedia.org/wiki/Autonegotiation>



Geteilte Leitung  
Klassisches Ethernet



Geteiltes HF  
802.11



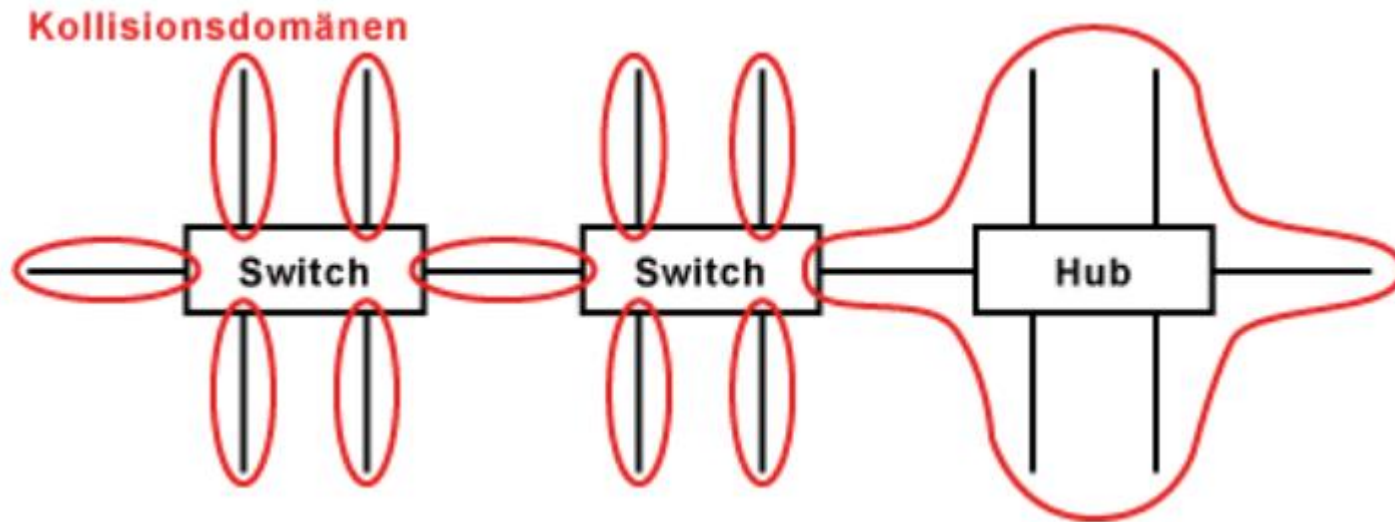
Geteiltes HF  
(Satellit)



Menschen auf einer Party  
(geteilte Musik)

# Switched Ethernet

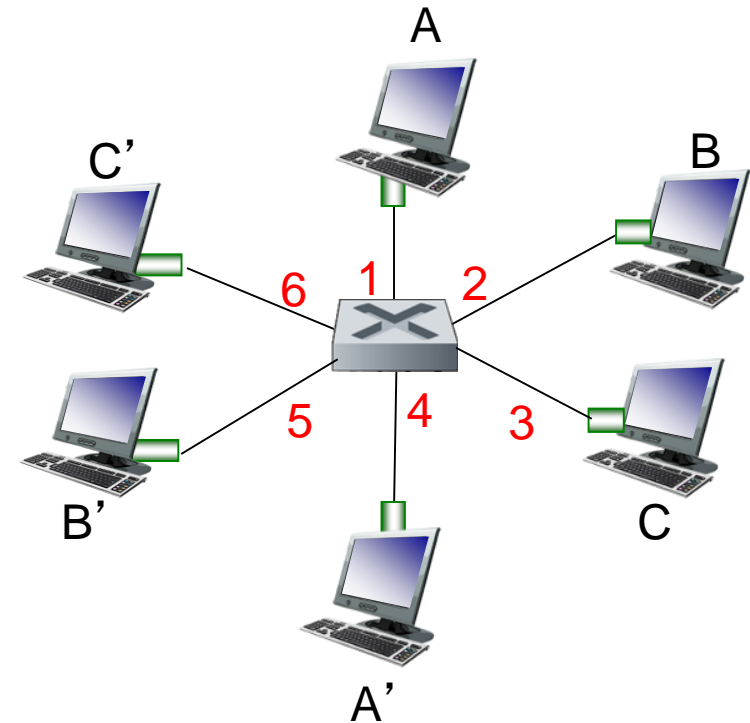
- **Hub:** Alle Leitungen sind quasi miteinander verbunden
  - Eine einzige **Kollisionsdomäne** (== Bereich, in dem nur 1 Host gleichzeitig sprechen darf)
  - Es muss CSMA/CD verwendet werden.
- **Switch:** Isoliert jeden Port in eine eigene Kollisionsdomäne
  - Bei Vollduplex-Kabeln: Kein CSMA/CD nötig!
- **Hinweis:** Nicht verwechseln mit **Broadcastdomäne**
  - == Reichweite eines Ethernet Broadcast-Frames (FF:FF:FF:FF:FF:FF)



Quelle:  
<https://www.elektronik-kompodium.de/sites/net/1406181.htm>

# Switched Ethernet: Das „moderne“ Ethernet

- ❑ Jeder Host direkt mit Switch-Port verbunden.
  - Jedes Kabel ist ein Punkt-zu-Punkt Netz
  - Keine Kollisionen möglich, falls Vollduplex.
  - Kein CSMA/CD nötig
- ❑ Aufgaben von Switches
  - Zwischenspeichern von Frames
  - Weiterleiten von Frames
- ❑ Hinweis. Gleichzeitige Übertragung von A zu A' und B zu B' möglich



*Switch mit 6 Interfaces  
(1,2,3,4,5,6)*

# Ethernet Switch

## ❑ **Arbeitet auf Link Layer**

- Empfang, Zwischenspeicherung und Weiterleitung von Ethernet Frames
- Untersucht MAC Adresse der ankommenden Frames und leitet Frame selektiv nur an "richtigen" Port weiter.
- Klassischer Link-Layer Switch hat keine IP Adresse!

## ❑ **Transparenz**

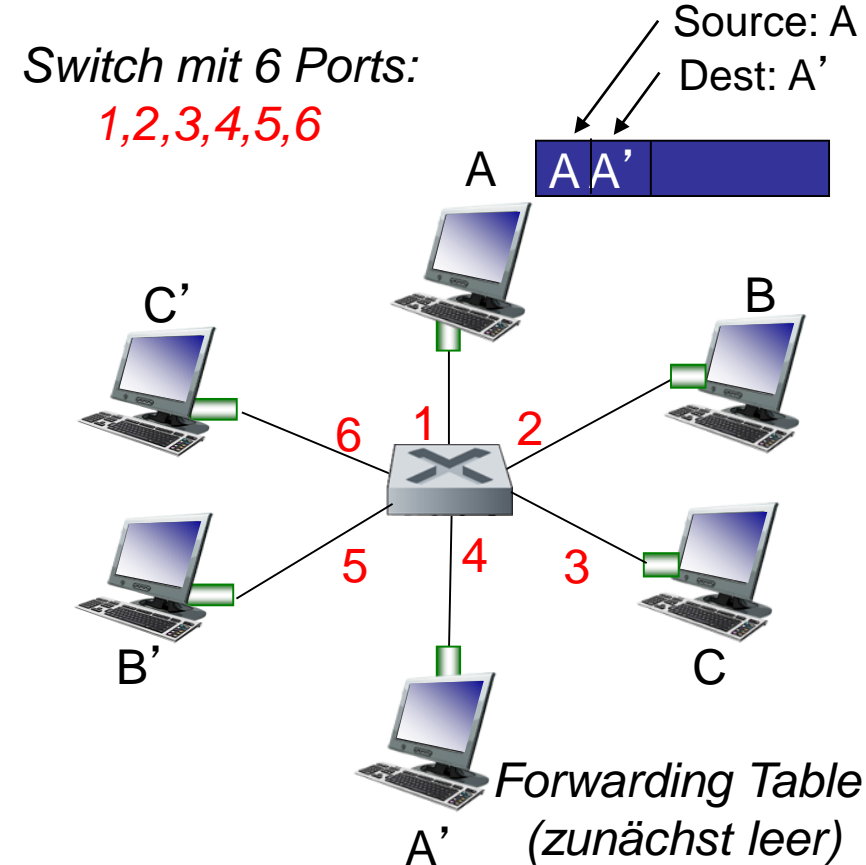
- Ethernet Hosts merken nichts von Anwesenheit eines Switches

## ❑ **Plug-and-Play**

- Selbstlernend
- Switches müssen nicht konfiguriert werden

# Switch: Forwarding

- ❑ Zu welchen Ports muss Frame weitergeleitet werden?
  - Nachschlagen in Forwardingtabelle
- ❑ Einträge der **Forwardingtabelle**:
  - **MAC** des Zielhosts
  - **Port** des Zielhosts
  - **Time-to-Live (TTL)**: nach bestimmter Zeit wird Eintrag gelöscht
- ❑ Switches sind **selbstlernend**
  - Jeder empfangene Frame wird untersucht und für den Aufbau der Forwardingtabelle verwendet.
  - **Ankommender** Frame: Eintragen von Port und MAC des **Senders**



| MAC Adr | Port | TTL |
|---------|------|-----|
| A       | 1    | 60  |

## Bei Empfang eines Frames

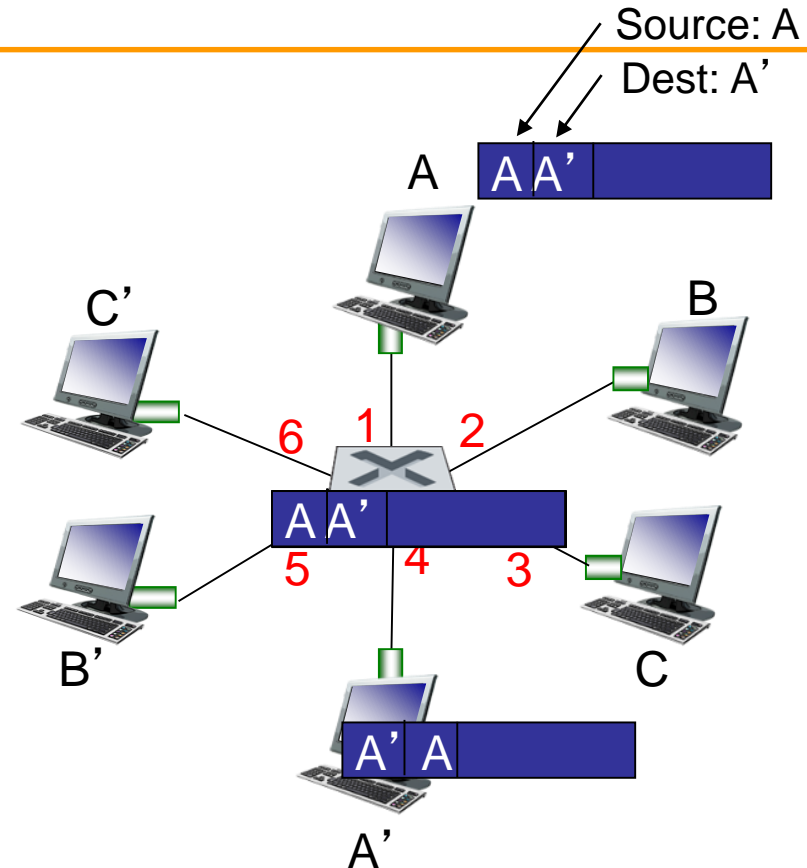
- ❑ Switch merkt sich Eingangsport und MAC Adresse des **Senders**
  - Eintrag in Switch Forwarding Tabelle
- ❑ Nachschlagen ob Eintrag für **MAC Zieladresse** bereits in Forwarding Tabelle:
  - **Falls** Eintrag vorhanden: Ermitteln des Zielports
    - **Falls** Zielport == Quellport: Frame verwerfen
    - **Sonst:** Leite Frame an entsprechenden Zielport weiter
- ❑ **Sonst:** Fluten
  - Weiterleiten an alle Hosts mit Ausnahme des Senders.



# Switch: Forwarding Beispiel

- ❑ Zielport A' unbekannt
  - Fluten
- ❑ Zielport A bekannt:
  - Leite Frame nur an entsprechenden Port weiter

## Fluten



| MAC addr | interface | TTL |
|----------|-----------|-----|
| A        | 1         | 60  |
| A'       | 4         | 60  |

*Forwarding Tabelle  
(zunächst leer)*

# Publikums-Joker: Switches

Ein Host sendet an einen Switch mit  $n > 1$  Ports einen korrekten, sinnvollen Ethernet Frame. Welche der folgenden Fälle ist **nicht möglich**?  
(Annahme: Keinerlei Firewalls!)

- A. Er leitet Frame an **keinen** Port weiter.
- B. Er leitet Frame an **1** Port weiter.
- C. Er leitet Frame an  **$n-1$**  Ports weiter.
- D. Er leitet Frame an **alle Ports** weiter.



- ❑ Einführung
  - Network Interface Cards (NIC)
- ❑ Rahmenbildung
  - Byte Count, Byte Stuffing, Bit Stuffing
- ❑ Fehlererkennung und Fehlerkorrektur
  - Parität, Checksumme, CRC
- ❑ Ethernet 802.3
  - Frameformat, MAC Adressen
- ❑ Broadcast Networks: Problem des Vielfachzugriffs
  - ALOHA, CSMA, CSMA/CD, CSMA/CA, TokenRing
- ❑ Switched Networks (dt. vermittelte Netze)
  - Hub vs. Switch, Forwarding, Lernalgorithmus