Verteilte Systeme
(Betriebssysteme II)

Kapitel 3: Netze

Prof. Dr. Wolfgang Küchlin

Arbeitsbereich Symbolisches Rechnen
Wilhelm-Schickard-Institut für Informatik
Fakultät für Informations- und Kognitionswissenschaften

Universität Tübingen

Steinbeis Transferzentrum
Objekt- und Internet-Technologien (OIT)

Wolfgang.Kuechlin@uni-tuebingen.de
http://www-sr.informatik.uni-tuebingen.de

Überblick

- Konzept des Protokolls
- OSI-Modell zur Kommunikation in Netzen
- Nicht OSI Protokolle
- Local Area Networks
- Internetworking mit den TCP/IP Protokollen

Kommunikation

- **Kommunikation**
  - in traditionellen Systemen: über gemeinsamen Speicher
  - in verteilten Systemen: Nachrichtenaustausch (*message passing*)
- **Voraussetzung für Kommunikation**
  - Lesbarkeit der Signale / Buchstaben / Datensätze
  - Übereinkunft über die Bedeutung der Signale
  - Übereinkunft über Zustand und Ablauf der Interaktion
- **Beispiele**
  - Umstellung des Telefon-Freizeichens auf Dauerton
  - Kulturelle Unterschiede im menschlichen Verhalten

Protokolle

- Übereinkünfte zur Kommunikation heißen **Protokolle.**

Definition *(Kommunikationsprotokoll, Datenprotokoll)*
Ein **Protokoll** (protocol) *ist ein Satz von Regeln, Datenformaten und Konventionen, die den Datentransfer und seine Bedeutung zwischen Kommunikationsteilnehmern regeln.*

- **Datenprotokolle**
  - legen Datenformate fest
    - z.B. XML-Schema; IEEE 794-1985 float
- **Kommunikationsprotokolle**
  - legen Ablauf der Kommunikation fest (endl. Automat)
    - z.B. remote procedure call (RPC); Verbindungsaufbau TCP

OSI-Modell

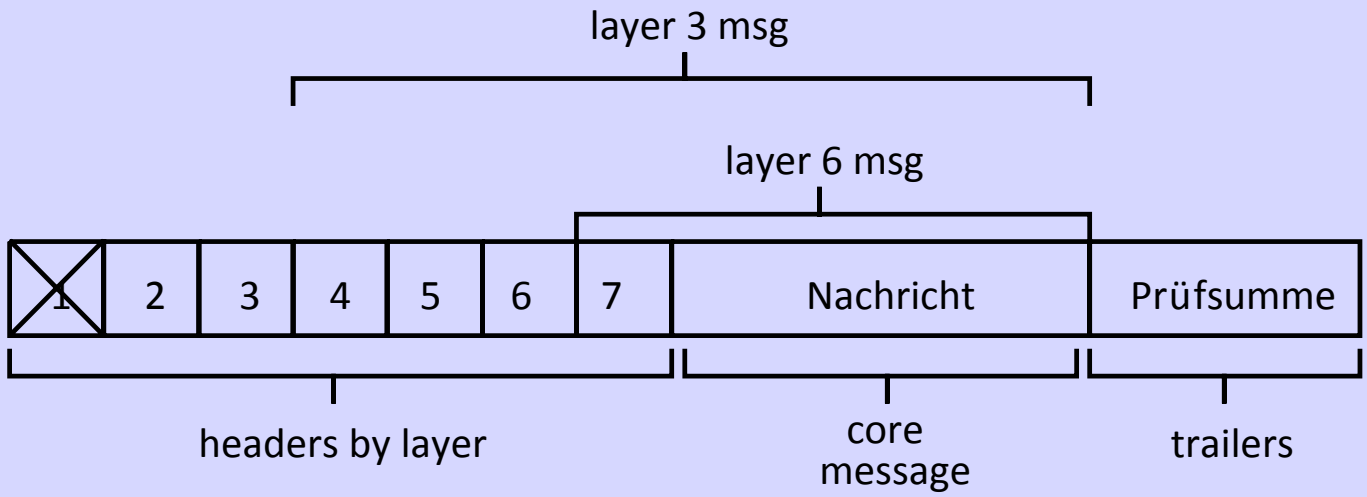
- **Standardisierung der Protokolle** (z. B. von der ISO),
  - ermöglicht **offene Systeme** (*open systems*): Verschiedene Einzelsysteme können über die Standardschnittstellen zu heterogenen Gesamtsystemen vernetzt werden
- **Standardisierungsgremien**
  - International Standards Organization (ISO)
  - IEEE (Institute of Electrical and Electronics Engineers, www.ieee.org)
  - Diverse Industrie-Konsortien (inoffiziell)
- **Open System Interconnect(ion) Reference Model (ISO - OSI)**
  - Referenzmodell der ISO
  - Hierarchie von Schichten (*layers*), die sich um verschiedene Aspekte der Kommunikation kümmern

OSI-Modell

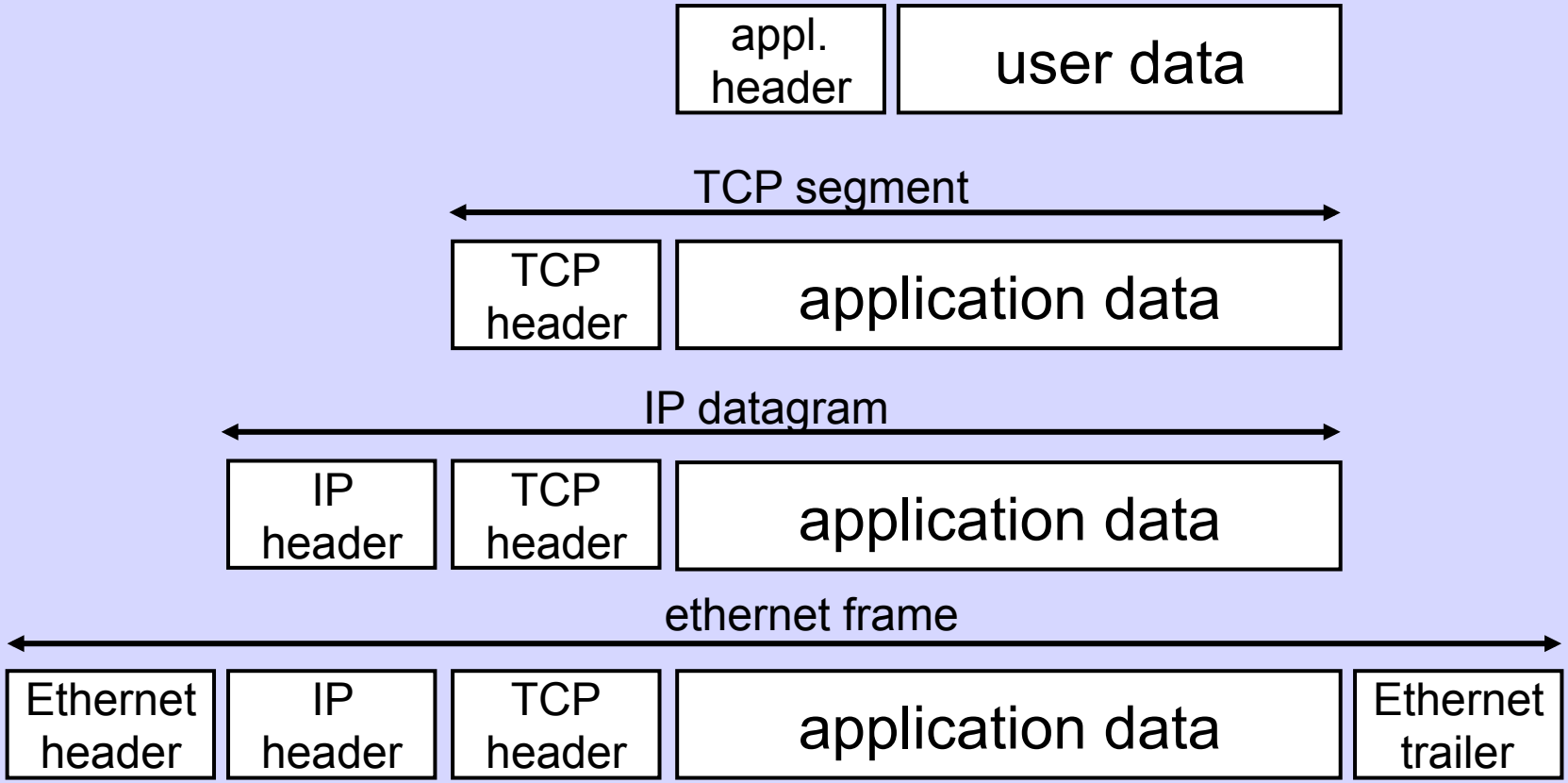
Schicht		Name	Stichwort
7.	Application	Anwendungsdienste	File Transfer Electronic Mail UNIX R-Dienste
6.	Presentation	Darstellung	Heterogener Datenaustausch
5.	Session	Sitzung (Kommunikations- steuerung)	Prozesskommunikation
4.	Transport	Transport	Prozessverbindung
3.	Network	Vermittlung	Paket über das Internet von Rechner zu Rechner
2. LAN	Data Link	Verbindung	Frame Transport auf Draht (sicherer Bitstrom)
1.	Physical	Physikalische Bit Transport	Übertragungs-Hardware

OSI-Modell

- Jede Schicht präsentiert nach oben eine Schnittstelle und bietet festgelegte Dienste an
  - tiefere Dienste werden von höheren Schichten benutzt
  - Höhere Schicht stellt *Header* voran und gibt Nachricht nach unten



Encapsulation



OSI-Modell – Beschreibung der Schichten

(1) Physical Layer

- elektrotechnische und mechanische Spezifikationen der Kabel (z.B. Stecker)
- Ermöglicht Übertragung einzelner Bits.

(2) Data Link (Bitstrom auf Draht)

- zuverlässige Übertragung einer Bitsequenz.
- Senden:
  - gruppiert Bitstrom in **Rahmen** (*frames*)
  - fügt Prüfsummen an
  - wiederholte Übertragung beschädigter Frames
- Empfangen:
  - Zusammensetzen der Frames in ihrer korrekten Reihenfolge
- Beispiele
  - IEEE 802 (Ethernet protocol), PPP, ATM Transmission Convergence Sublayer

OSI-Modell – Beschreibung der Schichten

(3) Network (Paket vom Rechner über das Netz zum Rechner)

- *Routing*:
  Explizit adressierte Nachricht wird über Zwischenstationen zum Empfänger (Rechner) geschleust
- *Pakete*:
  Adressierbare Nachrichten
- Versand von Paketen kann
  - unabhängig sein (*connectionless*) oder
  - es wird eine Verbindungen aufgebaut (*connections*)
- Beispielprotokolle:
  - X.25 (*connection*) und **IP** (**Internet-Protocol**) (*connectionless*)

OSI-Modell – Beschreibung der Schichten

(4) Transport (Prozess–zu–Prozess – Verbindung)

- Punkt-zu-Punkt Verbindung zwischen Prozessen
- Transportprotokolle können unabhängig von Schicht 3
  - *connectionless* oder
  - verbindungsorientiert sein
- **sockets** erlauben Zugriff auf Verbindungsprotokolle aus dem Betriebssystem bzw. Prozess heraus

OSI-Modell – Beschreibung der Schichten

(5) Session (Dialogsteuerung)

- Erweiterung der Transportschicht
  - Benutzer auf verschiedenen Maschinen können eine gemeinsame **Sitzung** eröffnen und durchführen.
  - erlaubt z.B. eine Dialogsteuerung, damit nur jeweils **eine** Partei spricht (Daten sendet).
- Verwandt dazu ist **token management** (Zugangskontrolle)
  - Besitz von *tokens* kann Zugangsberechtigung darstellen → verteiltes **mutex**
- weiterer Service: Synchronisation von Datentransfers
  - Wiederaufsetzpunkte (*checkpoints*), so dass bei Zusammenbruch der Verbindung nur ab diesem Punkt weiter übertragen werden muss

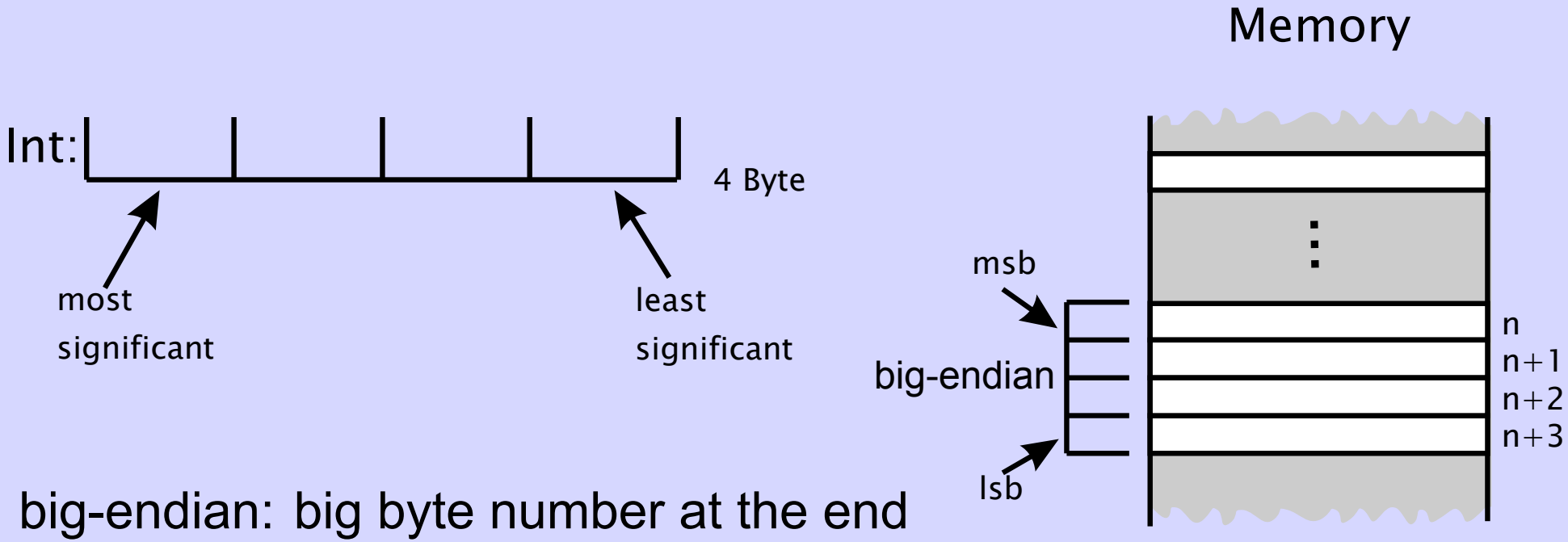
OSI-Modell – Beschreibung der Schichten

(6) Presentation Layer (Darstellungsschicht)

- Daten in verschiedenen rechnerspezifischen Darstellungen können ausgetauscht werden
  - statt Bits und Bytes jetzt höhere Datentypen (Characters, Integers, Floats, …)
  - Probleme:
    - EBCDIC, ASCII, little-endian, big-endian, 1's compl, 2's compl, . . .
    - IBM Mainframes, SUN: big-endian; Intel: little-endian
- weitere Dienste:
  - Datenkompression
  - Verschlüsselung

OSI-Modell – Beschreibung der Schichten

- Erinnerung: little-endian und big-endian



big-endian: big byte number at the end
(MSB at lowest address)
little-endian: little byte number is at the end
(LSB at lowest address)

OSI-Modell – Beschreibung der Schichten

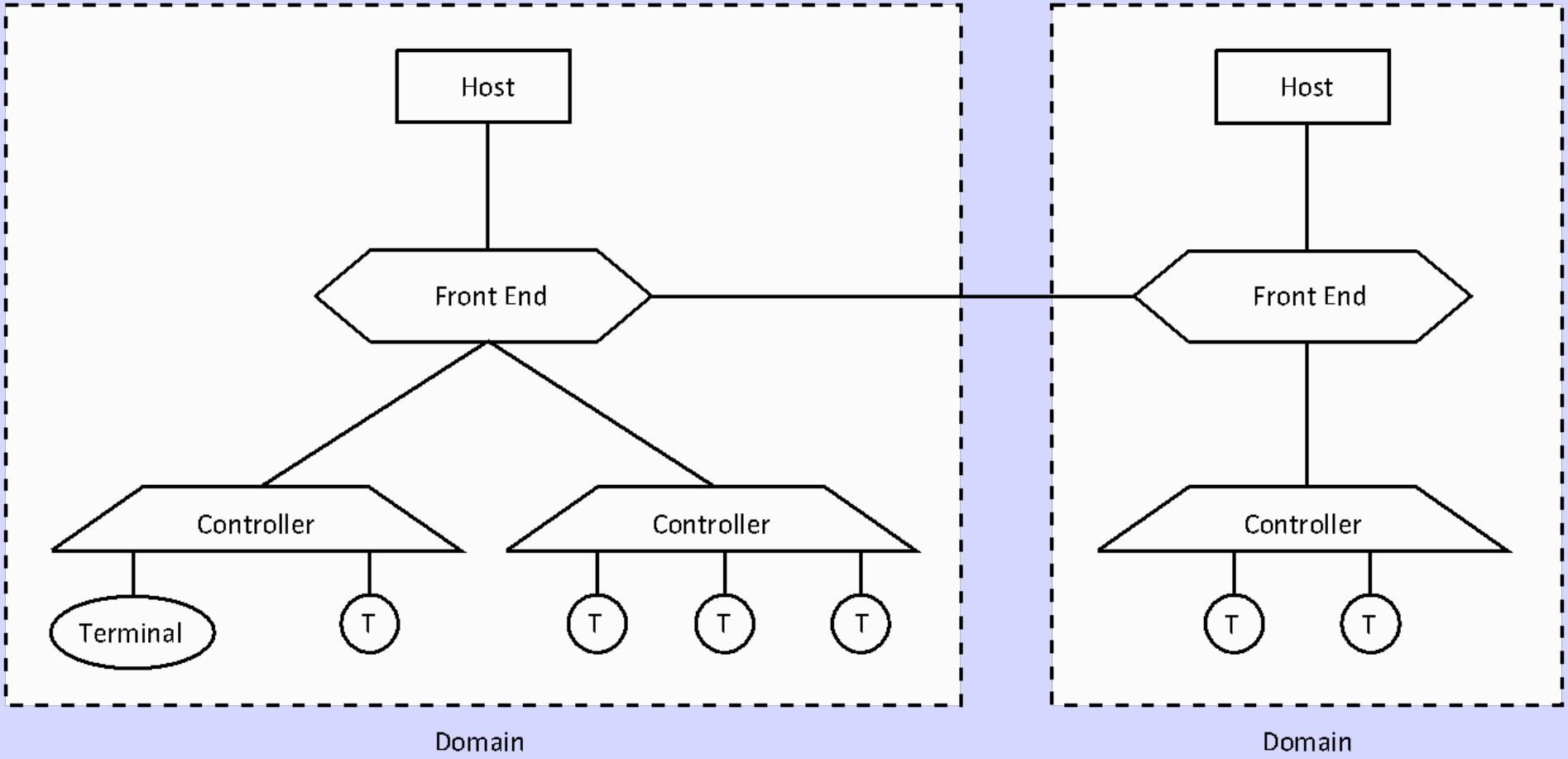
(7) Application Layer (Anwendungsschicht)

- verschiedene Protokolle für die anwendungsorientierte Benutzung eines Netzes
- virtuelles Terminal, das Anwendungen wie Editoren etc. benutzen können, und dessen Steuersignale auf existierende reale Terminals abgebildet werden können.
- Außerdem Protokolle für
  - Remote Login
    - OSI: VT
    - TCP/IP: Telnet
  - File Transfer
    - OSI: FTAM (File Transfer – Access and Management)
    - TCP/IP: FTP

Nicht OSI Protokolle – SNA

- System Network Architecture (SNA) wurde 1974 von IBM freigegeben
  - hat OSI wesentlich beeinflusst
- NAUs (*network addressable units*)
- SNA ist generell sitzungsorientiert

Nicht OSI Protokolle – SNA



Nicht OSI Protokolle – SNA

	SNA-Schicht	Funktionen	
	OSI		
		Network services	7
6	NAU Services	Session Services - Verbindungen	
		Presentation Services (e.g. compression)	6
5	Data Flow Control	Sprecherlaubnis	5
		(Encryption/Decryption, Multiplexing	
4	Transmission Control	Priorities, Buffer management, flow control) management	4
		connection setup, deletion	
3	Path Control	NAU-NAU logical path setup	
		(virtual path, explicit path, line usage)	3
2	Data Link Control	SDLC: secure data link	2
		Token ring LAN	
1	Physical Physical	Transport	1

Nicht OSI Protokolle – Net BIOS

- IBM Produkt von 1984 für IBM-PC-Netzwerke
- basiert auf Bus-Verbindung

Schicht	Funktion	OSI
3	User Process	7,6,5
2	Name Service Session Service (connection) Datagram Service General Commands	4,3
1	Hardware Interface	2,1

Nicht OSI Protokolle – UUCP

- UNIX-UNIX Copy
- Ausgangspunkt uucp:
  - Übertragung von Files von einem Rechner zum anderen mittels Wahlleitung und Modem
    - kopieren der Aufträge in `/usr/spool/uucp`
    - `cron` Dämon aktiviert `uucp`-Dienst

Local Area Networks – Ethernet (IEEE 802)

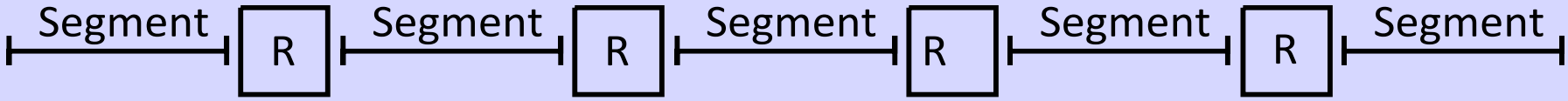
- entspricht OSI Schichten 1 und 2
- von XEROX PARC in den 1970ern entwickelt
- Ethernet basierte zunächst auf einem durch ein Koax Kabel realisierten Bus mit 10 Mb/s
  - inzwischen
    - 100 Mb/s bis 1 Gb/s, 10 Gb/s, …
    - Twisted-Pair-Kabel, Glasfaser, …
  - Übertragung in Rahmen zu mindestens 512 Bits.
  - Jeder Rahmen beginnt mit einem Synch.-Muster von 64 Bits: 1010 . . . 1011.

Local Area Networks – Ethernet

- Zugriffsverfahren: **CSMA/CD** (*Carrier Sense Multiple Access/ Collision Detection*).
- **carrier sense multiple access**
  - sendewillige Station hört, ob Bus frei ist (*carrier sense*)
    - Wenn Bus frei: Senden
  - Problem: Wegen Kabellänge Überschneidungen möglich
    - bei CSMA: Fertig senden und ACK abwarten
    - Bei Ausbleiben (Störung) wird Sendung wiederholt
- **collision detection**
  - Sendung aller Beteiligten durch separates **jam** Signal unterbrechen
  - Jeder Sender wartet *zufällig* gewählte Zeitspanne (*back-off*), bevor er erneut sendet
  - genauer: *binary exponential back off*
    - Jeder Sender wartet 0 oder 1 Zeiteinheiten
    - bei nochmaliger Störung 0, 1, 2 oder 3
    - bei n Störungen von 0 bis 2^n

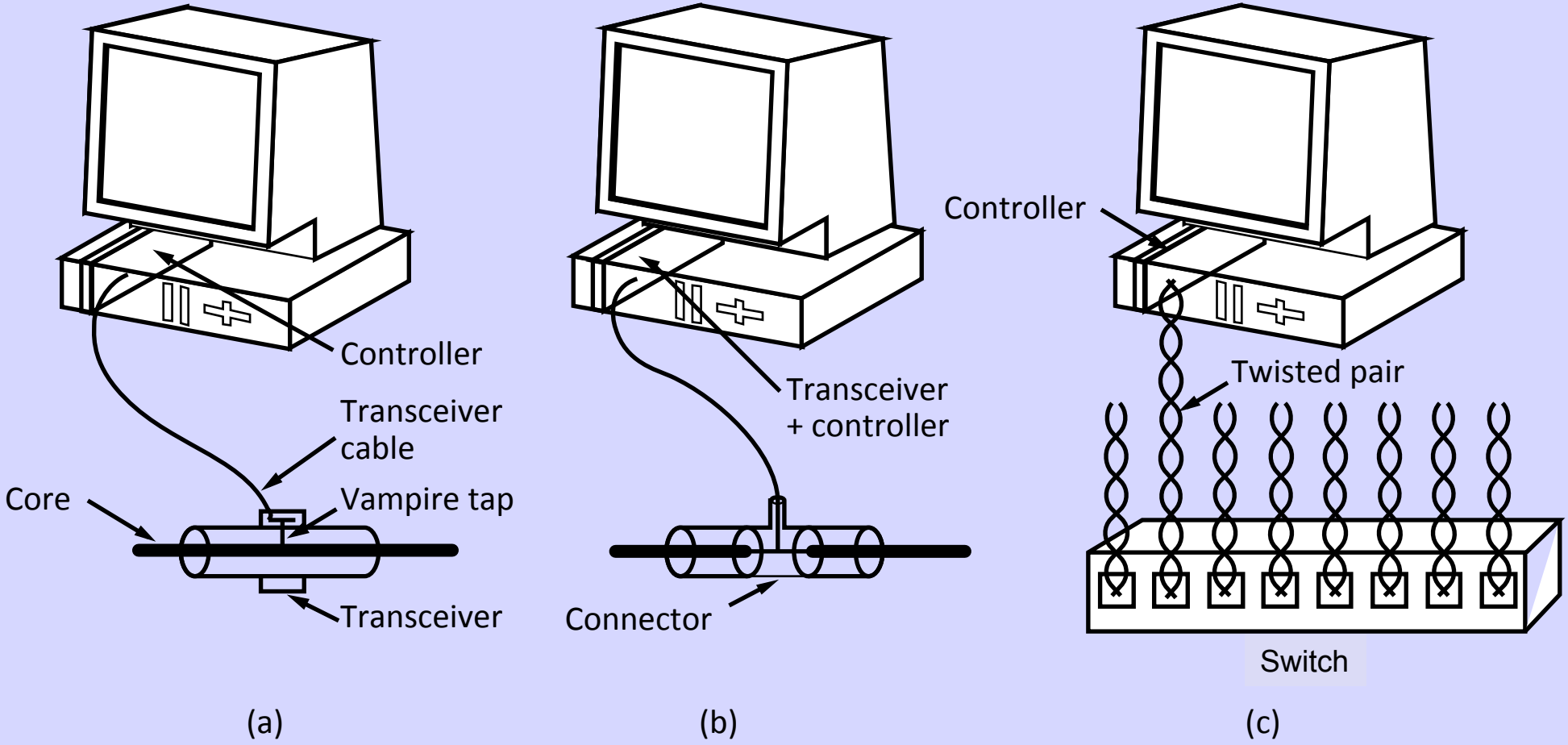
Local Area Networks – Ethernet

- max. Kabellänge von 5.2 km
  - ergibt sich aus Rahmenlänge, Signalausbreitungsgeschwindigkeit und Reaktionszeiten bei Kollision
  - Sender muß *während* der Sendung von Kollisionen erfahren, da kein ACK geschickt wird
- Verstärker nach 500m wegen Eintreten der Dämpfung erforderlich



- Verbindung verschiedener Ethernets durch **Bridges**
  - Weiterleitung der Rahmen, deren Adresse auf einem anderen Netzteil liegt

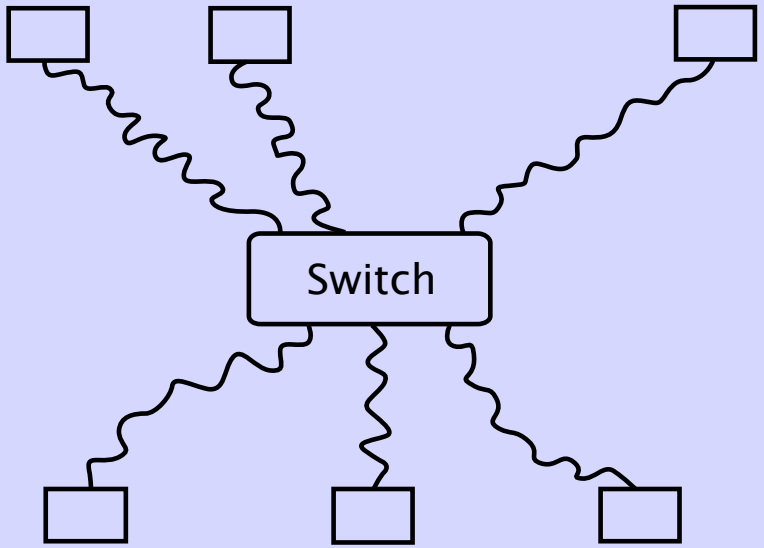
Local Area Networks – Ethernet



Tanenbaum, Netzwerke (3rd Ed.), Abb. 4-18

Local Area Networks – Ethernet

- Jedes Ethernet Board trägt 48 Bit Adresse (*MAC-Adresse*)
- ursprünglich
  - erste 3 Bytes für Hersteller reserviert (von IEEE)
  - letzte 3 Bytes für Gerätenummer
- Heute: Point-to-Point Twisted Pair switched Ethernet



Local Area Networks – Ethernet

- Aufbau eines Ethernet-Rahmens
  - Kompromiss zwischen Verpackungsoverhead und Gefahr von Kollision
  - Paketende wird automatisch erkannt (Manchester-Code)
  - zwischen Rahmen muss 9.6 µs Pause sein

Präambel	8 Bytes	Synchronisation
Empfänger	6 Bytes	Adresse (1-1 = Broadcast)
Sender	6 Bytes	Adresse
Länge	2 Bytes	Länge des Datenfeldes
Daten	≤ 1500 Bytes	
Pad	≤ 46 Bytes	Falls Daten < 46 Bytes, wegen Rahmenmindestlänge von 64 Bytes (ohne Präambel).
Prüfsumme	4 Bytes	Cyclic Redundancy Check

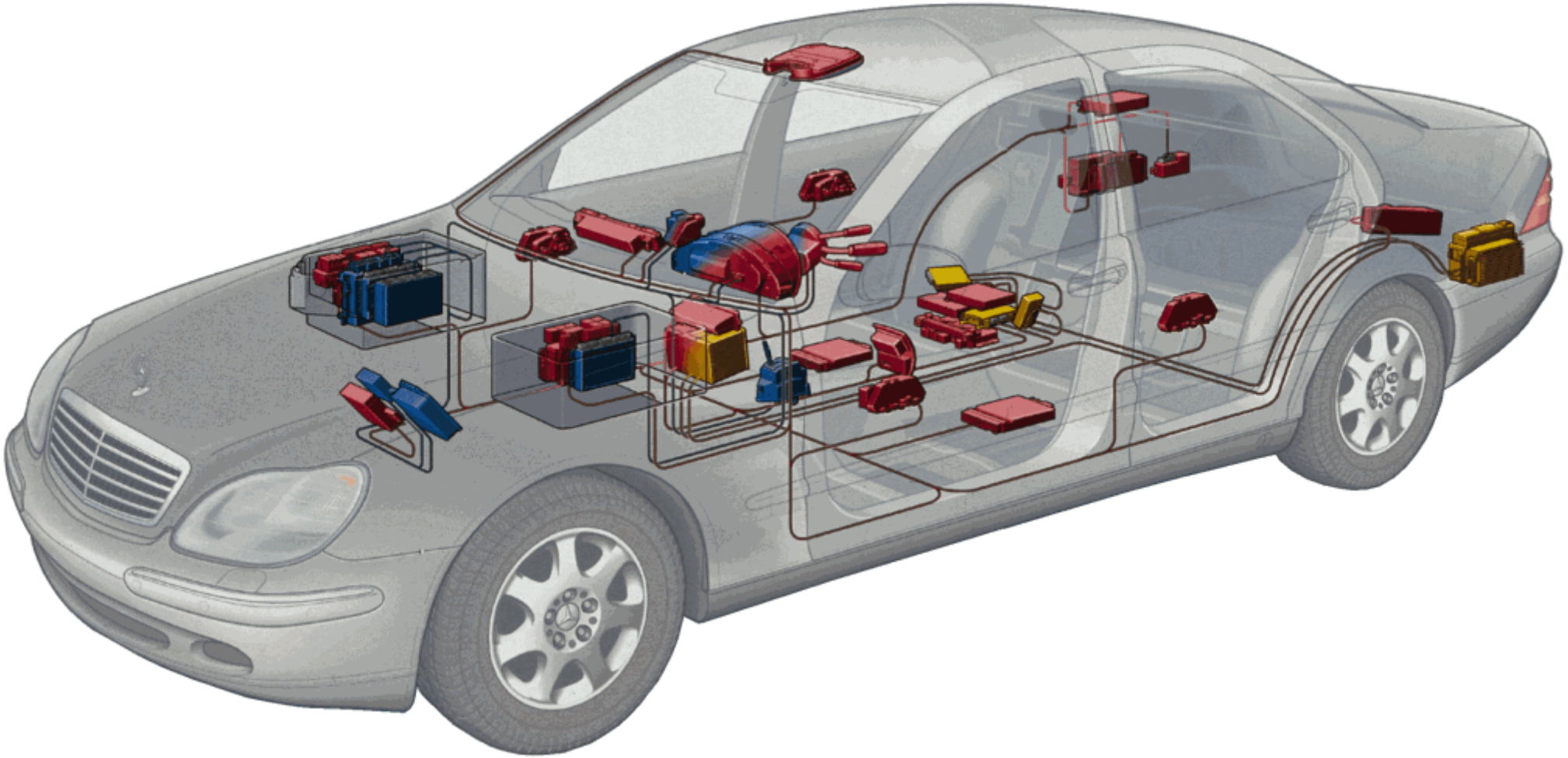
Local Area Networks – Ethernet

- Broadcast
  - Ethernet bietet auf physikalischer Schicht **broadcast**
    - Adressierung geht aber i. a. auf Data Link Layer an eine Station
  - Unterstützung von Gruppenadressen auf dieser Ebene
    - (high-order bit 1 bei Gruppenadressen; 0 bei Einzeladressen)
  - logischer **Broadcast** zu allen Stationen des Netzes: Empfängeradresse 1111. . . 111
- Verwendung von Ethernet
  - Ethernet wird von vielen verschiedenen höheren Protokollen benutzt.
  - Beispiele:
    - TCP/IP
    - XNS Protokolle von Xerox
    - Netzwerkprodukte der Firma Novell (PC Netze mit einem PC als Netzwerk-Server unter NetWare)

Local Area Networks – Feldbus CAN

- Typischer Einsatz:
  - Steuerung aller Einzelgeräte in einem Großgerät (z.B. PKW, LKW)
- Anforderungen an Bussystem für Realzeitbereich (Automatisierungs-Bus, Feldbus)
  - Prioritätsklassen für angeschlossene Geräte
  - Obergrenze für Zeitdauer des Nachrichtenaustauschs
  - gute Abschirmung gegen Störsignale
- unwichtigere Kriterien
  - Datenrate (ca. 1 MBit/sec ausreichend)
  - Länge (höchstens einige Dutzend Meter nötig)
- Nutzfahrzeug- und Automobiltechnik: Bussystem CAN (Controller Area Network) führend
  - ursprünglich von Bosch für Mercedes entwickelt
  - S-Klasse drei Bussysteme
  - Maybach vier: Entertainment, Motor Control, Complex Devices, Simple D.

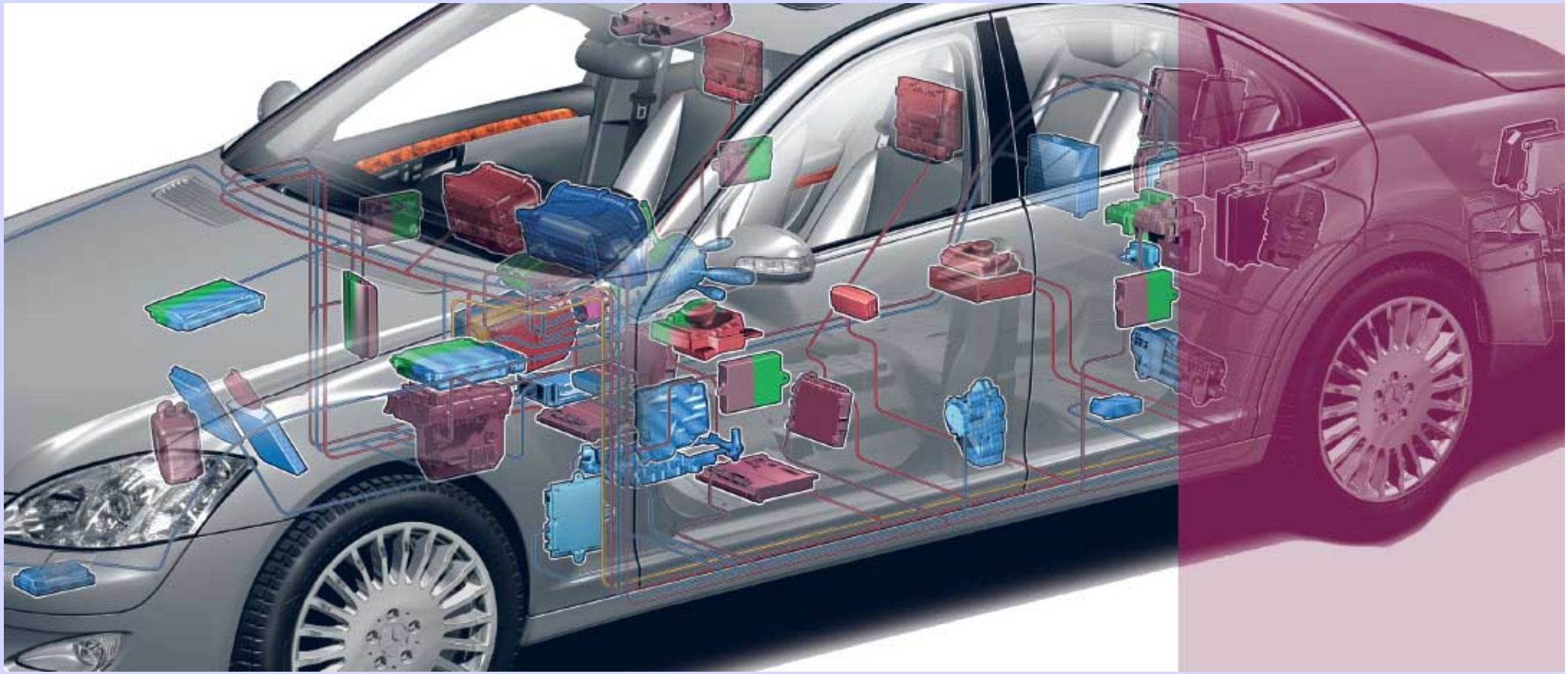
(z.B.: MOST, CAN-C / TT-CAN / FlexRay, CAN-B, ..)



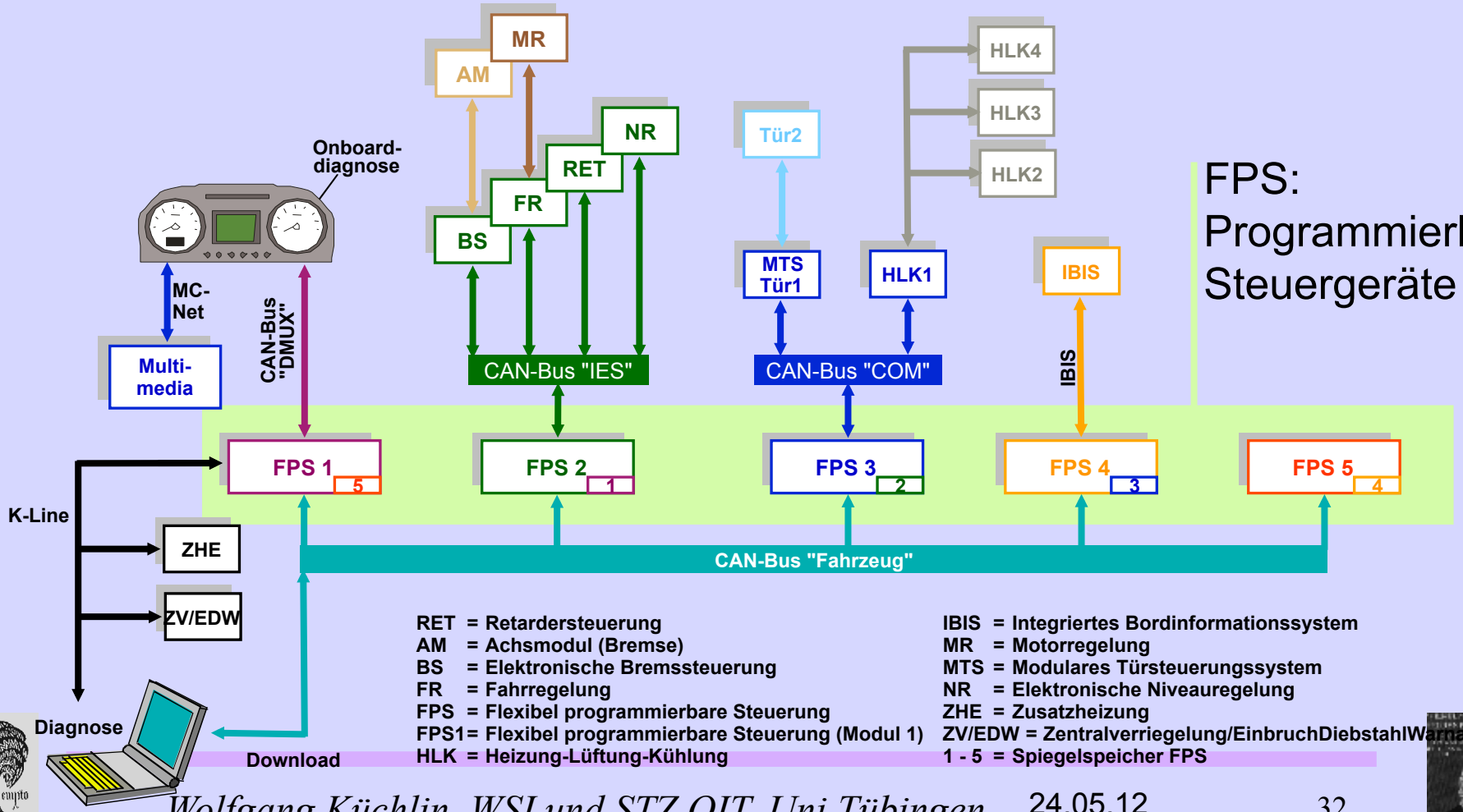
Vernetzung W220 (ältere S-Klasse)

- CAN-B, Innenraum-CAN, rot
- CAN-C, Motorraum-CAN, blau
- D2B, MOST, Multimedia-Daten, gelb

Vernetzte Steuergeräte im PKW



EE-Architektur des Omnibus TC400 / Travego

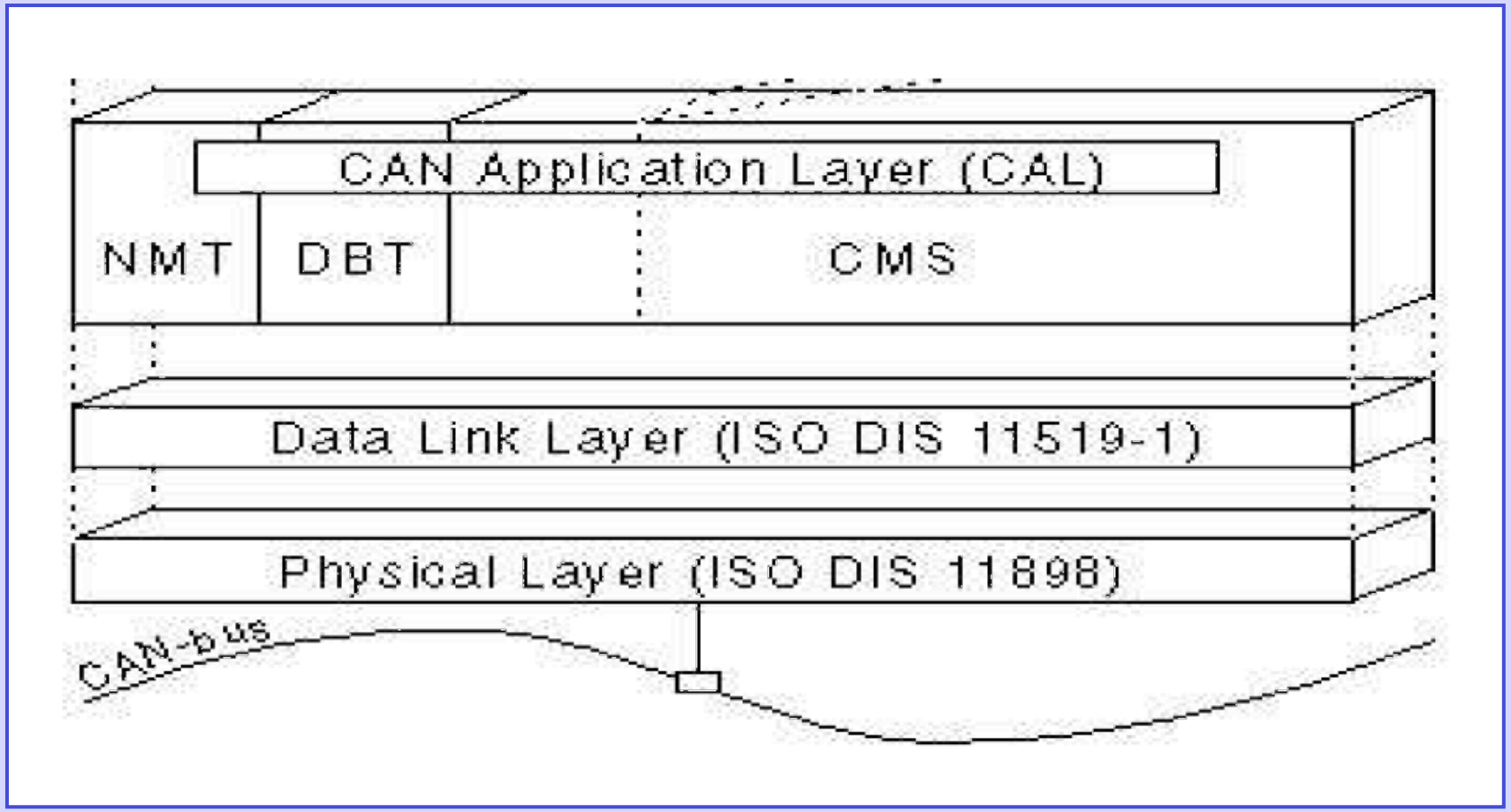


FPS: Programmierbare Steuergeräte

RET = Retardersteuerung
AM = Achsmodul (Bremse)
BS = Elektronische Bremssteuerung
FR = Fahrregelung
FPS = Flexibel programmierbare Steuerung
FPS1 = Flexibel programmierbare Steuerung (Modul 1)
HLK = Heizung-Lüftung-Kühlung

IBIS = Integriertes Bordinformationssystem
MR = Motorregelung
MTS = Modulares Türsteuerungssystem
NR = Elektronische Niveauregelung
ZHE = Zusatzheizung
ZV/EDW = Zentralverriegelung/EinbruchDiebstahlWarnanlage
1 - 5 = Spiegelspeicher FPS

Local Area Networks – CAN



Local Area Networks – CAN

- Ausdehnung/Datenrate:
  - ca. 1 Mbit/sec bei 40 m
  - 0.5 Mbit/sec bei 100 m
  - 0.1 Mbit/sec bei 500 m
- Priorisierung: Identifier der Nachricht bestimmt Priorität
- CSMA/CA (Carrier Sense Multiple Access/Collision Arbitration)
  - jeder Teilnehmer hört die Identifier der Nachrichten bitweise mit
  - jede Station kann Nachricht auf dem Bus mit niedrigerer Priorität überschreiben
  - ursprünglicher Sender erfährt vor dem nächsten Bit vom Überschreiben und schweigt
- Technische Grundlage für Überschreiben:
  - dominante und rezessive Bits.
  - Jeder Busteilnehmer kann den Buspegel (durch Erden) erniedrigen
  - Hohe Pegel: rezessive (r-)Bits
  - niedrige Pegel: dominante (d-)Bits

Local Area Networks – CAN Nachrichtenformate

- Datentelegramm (*Data Frame*)
  - Datenübertragung vom Sender zu einem oder mehreren Empfängern auf Initiative des Senders.
- Datenanforderungstelegramm (*Remote Frame*)
  - Anforderung eines Data Frame mit demselben Identifier
  - z.B. von Bus Master an ein E/A-Gerät verschickt
- Fehlertelegramm (*Error Frame*)
  - Signalisation eines von einem Busteilnehmer erkannten Fehlers
- Überlasttelegramm (*Overload Frame*)
  - Netzknoten senden das nächste Data Frame oder Remote Frame mit einer Verzögerung

Local Area Networks – CAN Datentelegramm

Layout:

Feld	#Bits
SOF	(1 Bit)
Identifier/Arbitration	(11 Bit)
RTR (Remote Transmission Request)	(1 Bit)
Control fiel	(6 Bit)
Data field	(64 Bit)
CRC Segment	(15 Bit)
CRC Delimiter	(1 Bit)
ACK Slot	(1 Bit)
ACK Delimiter	(1 Bit)
EOF field	(7 Bit)
	108 Bit
Interframe Space	(3 Bit)

Local Area Networks – CAN Datentelegramm

- *SOF (Start of Frame)*
  - SOF dominantes Bit: Zeigt Beginn eines Identifiers an
- *Arbitrierungsfeld*
  - 11 Bit Object Identifier + 1 RTR Bit
  - Station mit niedrigstem Object Identifier übernimmt das Telegramm
  - RTR Bit: Unterscheidung von Data Frame und Remote Frame eines Object Identifier
  - Data Frame Bit ist dominant, d.h. bei gleichzeitiger Anforderung und Sendewunsch setzt sich Senden durch
- *Steuerfeld*
  - 4 niederwertige Bits: Länge des Datenblocks in Byte
- *Datenfeld*
  - 0 bis 8 Byte, höchstwertiges Bit in Byte zuerst.

Local Area Networks – CAN Datentelegramm

- *CRC Field*
  - 15 Bit CRC Checksum + Begrenzungsbit rezessiv.
- *ACK Field*
  - 1 Bit ACK Slot und 1 Bit Begrenzung, beide rezessiv.
  - Alle Busteilnehmer bestätigen fehlerfreien Empfang durch Senden eines d-Bits im ACK Slot
  - Feststellung eines Übertragungsfehlers, Empfänger schickt separates Fehlertelegramm
- *EOF Feld*
  - 7 rezessive Bits
  - Während Übertragung melden Empfänger Übertragungsfehler durch Aufpressen von d-Bits
  - Sender kann dann Telegramm wiederholen
  - 7 Bit nötig, da ein Empfänger u.U. erst nach 6 Bit erkennt, dass EOF Feld statt eines erwarteten Datenfeldes übertragen wird
    - Bit-Stuffing nach 5 Bit im Datenfeld

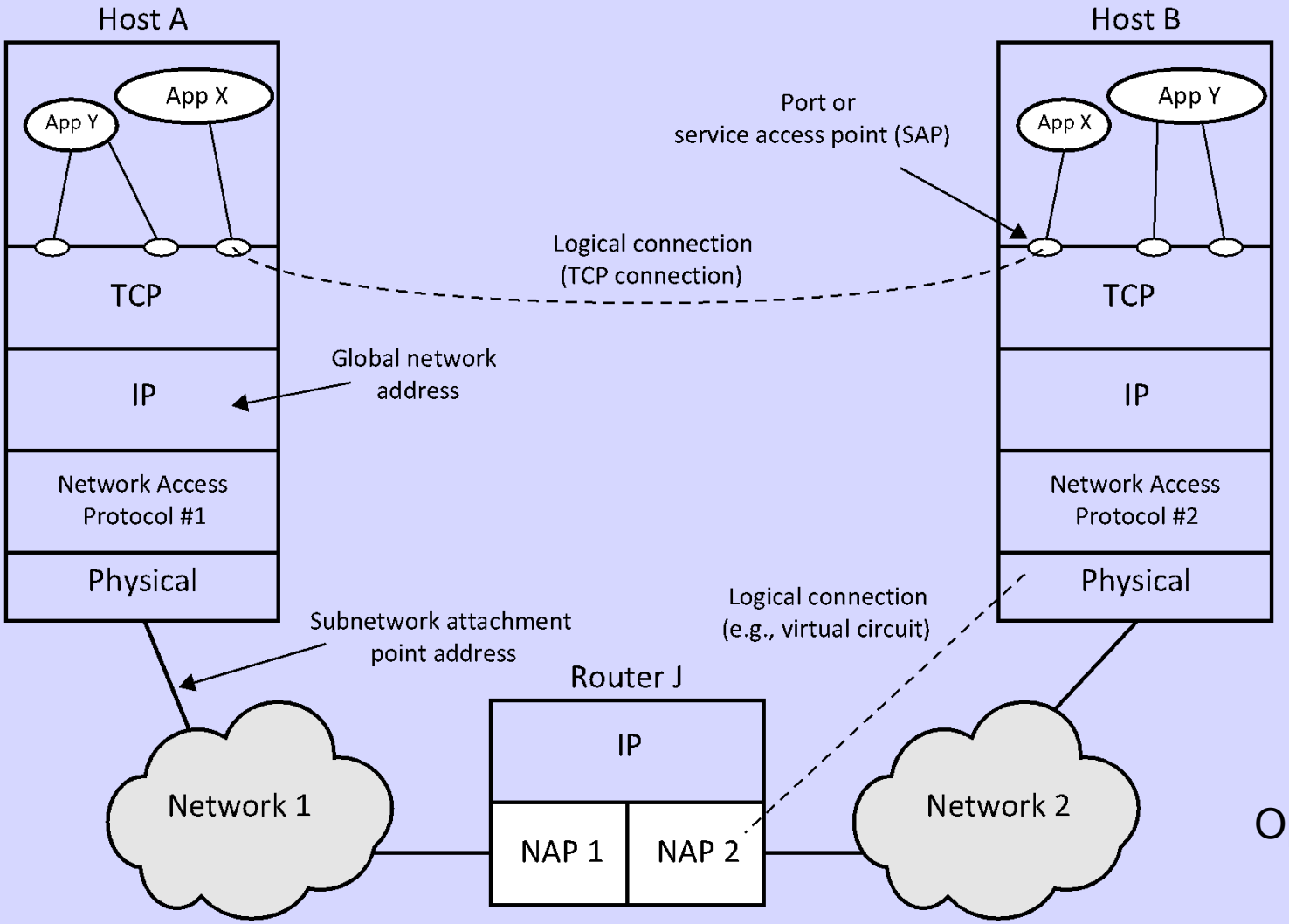
Local Area Networks – CAN Anwendungsschicht

- CAN Schicht 2: lediglich Data-Frames
  - keine Nachrichten oder Bedeutungen
- Can Application Layer CAL
  - relevante Anwendungsdienste
    - Netzwerk-Management (NMT)
    - Management der Protokollschichten (LMT)
    - Verteilung der Nachrichtenprioritäten (DBT)
  - Nachrichtenformate (-Typen).
- Schicht 7 Norm: CANopen
- OSI-Schichten 3-6 nicht explizit als CAN-Protokoll-Software implementiert
  - z.T. extern realisiert (Schicht 4)
  - z.T. entfallen (Schicht 3)
  - z.T. in andere Schichten integriert in Form von Spezifikationen in Schicht 7 (Schicht 5 und 6)

Internetworking mit den TCP/IP Protokollen

- Protokolle sind Teil des Internet
  - Ursprung: ARPA-Net der DARPA (Defense Advanced Research Projects Agency) (1969).
- militärisches Netz wurde von den US Hochschulen für zivile Zwecke adaptiert
  - seit 1982 wurden TCP/IP in Berkeley UNIX integriert
- Internet Kommunikationspartner heißen **hosts**.
  - jeder hat eine Adresse + verbunden durch das Netz
- Die Internet Protokolle werden vom IAB (Internet Architecture Board) gepflegt und weiterentwickelt.
- Standardisierungsvorschläge werden als *request-for-comments* publiziert.
  - Von diesen gibt es über 3000
  - z.B. RFC 791 für IP
  - bilden de facto schon die Standards

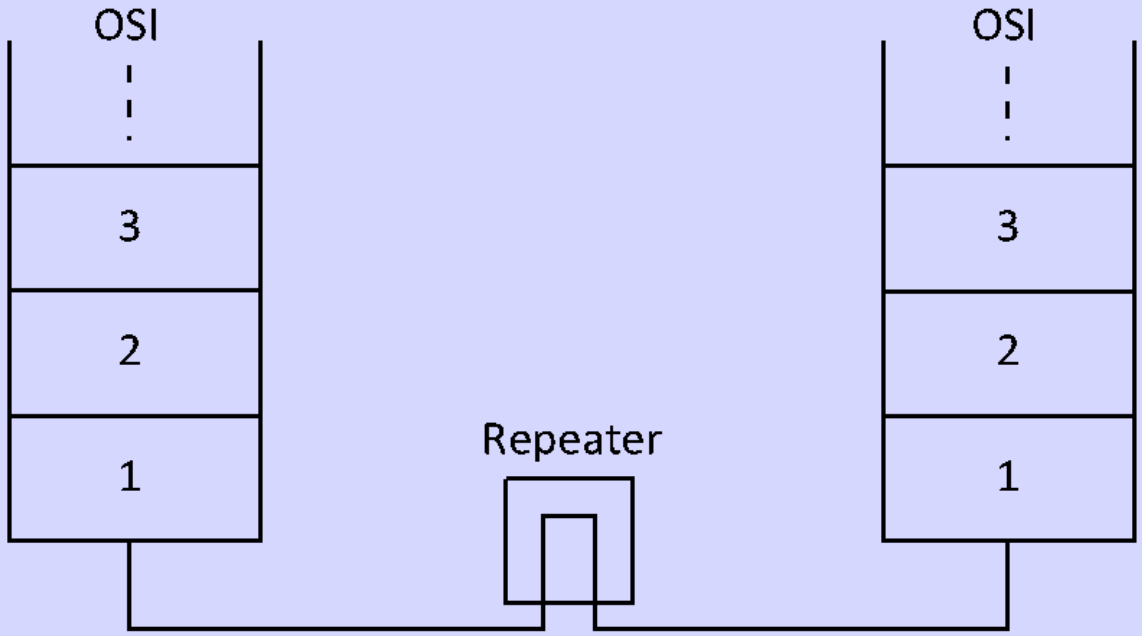
TCP/IP Konzept



Stallings, Operating Systems, Abb. A1

Netz-Netz-Verbindungen

- Verbindungen zwischen verschiedenen Netzwerken oder Netzteilen.
- Unterscheidung nach der Höhe der Protokollschicht, auf der die Umsetzung erfolgen muss.
- **Repeater**
  - Auf Schicht 1
  - Bits werden von einem Netzteil (-Segment) auf ein anderes geschleust
  - Paketaufbau wird nicht angetastet.

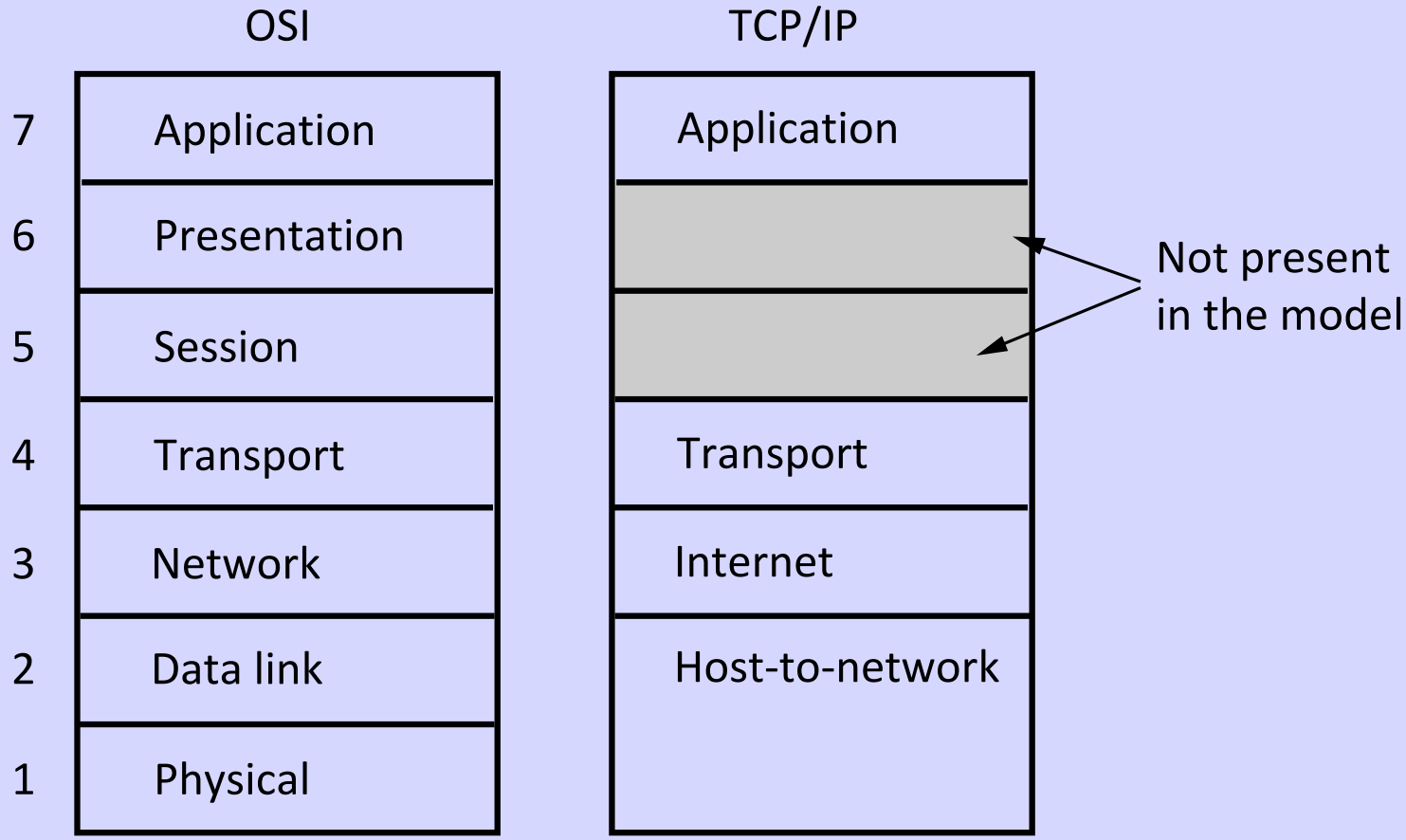


Netz-Netz-Verbindungen

- **Bridge** (Brücke)
  - formt Paketformate in der Sicherungsschicht des physikalischen Netzes um, z.B. zwischen Ethernet und Token Ring
  - Da Adresskonversion geleistet wird, können auch Pakete gefiltert werden
- **Router**
  - verbindet Netze auf der Vermittlungsschicht (OSI 3)
  - Verbindung von LAN mit Internet
  - Verbinden von z.B. verbindungslosen mit verbindungsorientierten Netzen
  - Entscheidungen über Pfadplanung (routing)
- **Gateway**
  - arbeitet auf höheren Ebenen
  - Oft vollgültiger Rechner, der verschiedenste Netze verbinden kann.

Vergleich: OSI und TCP/IP

- TCP/IP ist 10 Jahre älter als OSI und benutzt 4 Schichten.

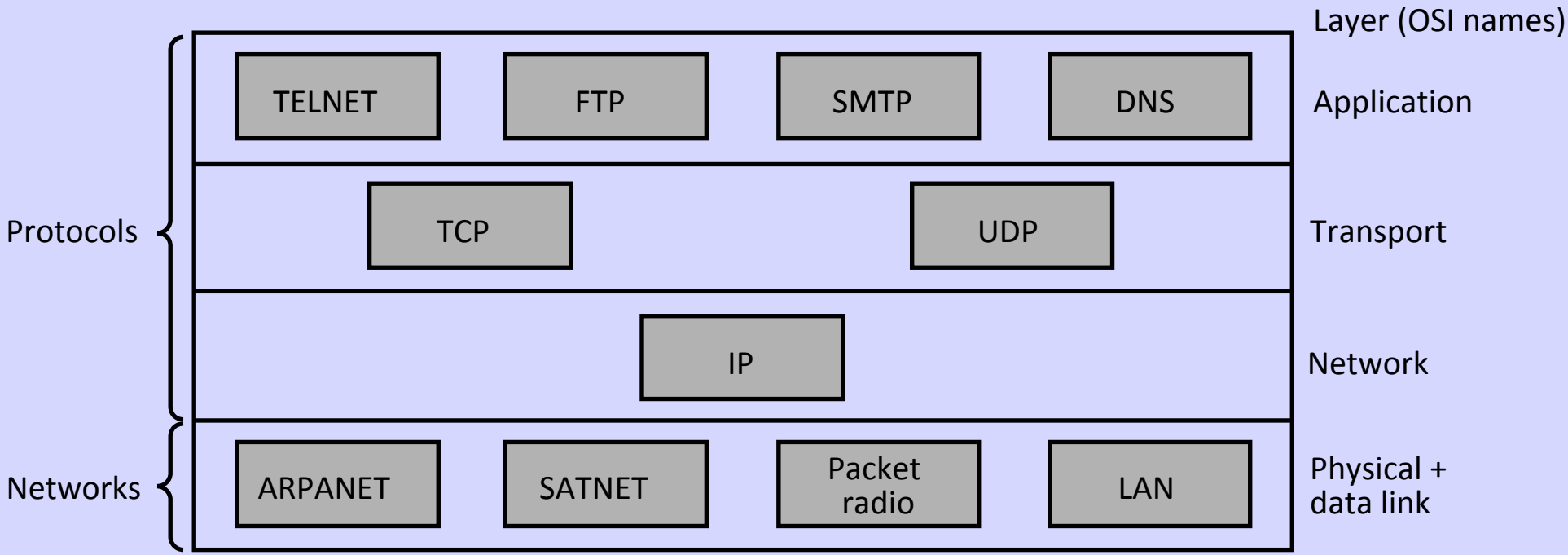


Tanenbaum, Networking, Abb. 1-21

TCP/IP Dienste

OSI	TCP/IP Schichtenaufbau	TCP/IP Dienste und Protokolle		
7 Anwendung 6 Darstellung	Anwendung (Application Level)	TELNET FTP SMTP	NFS XDR RPC	
5 Kommunikation 4 Transport	Transport (Transmission Level)	TCP, UDP	UDP	
3 Vermittlung	Vermittlung (Internet Level)	IP (ARP, RARP) { ICMP }		
2 Sicherung 1 Bitübertragung	Übertragung (Network Level)	(CSMA/CD) Ethernet (Bus)	Token Bus	Token Ring

Überblick



Tanenbaum, Networking, Abb. 1-22

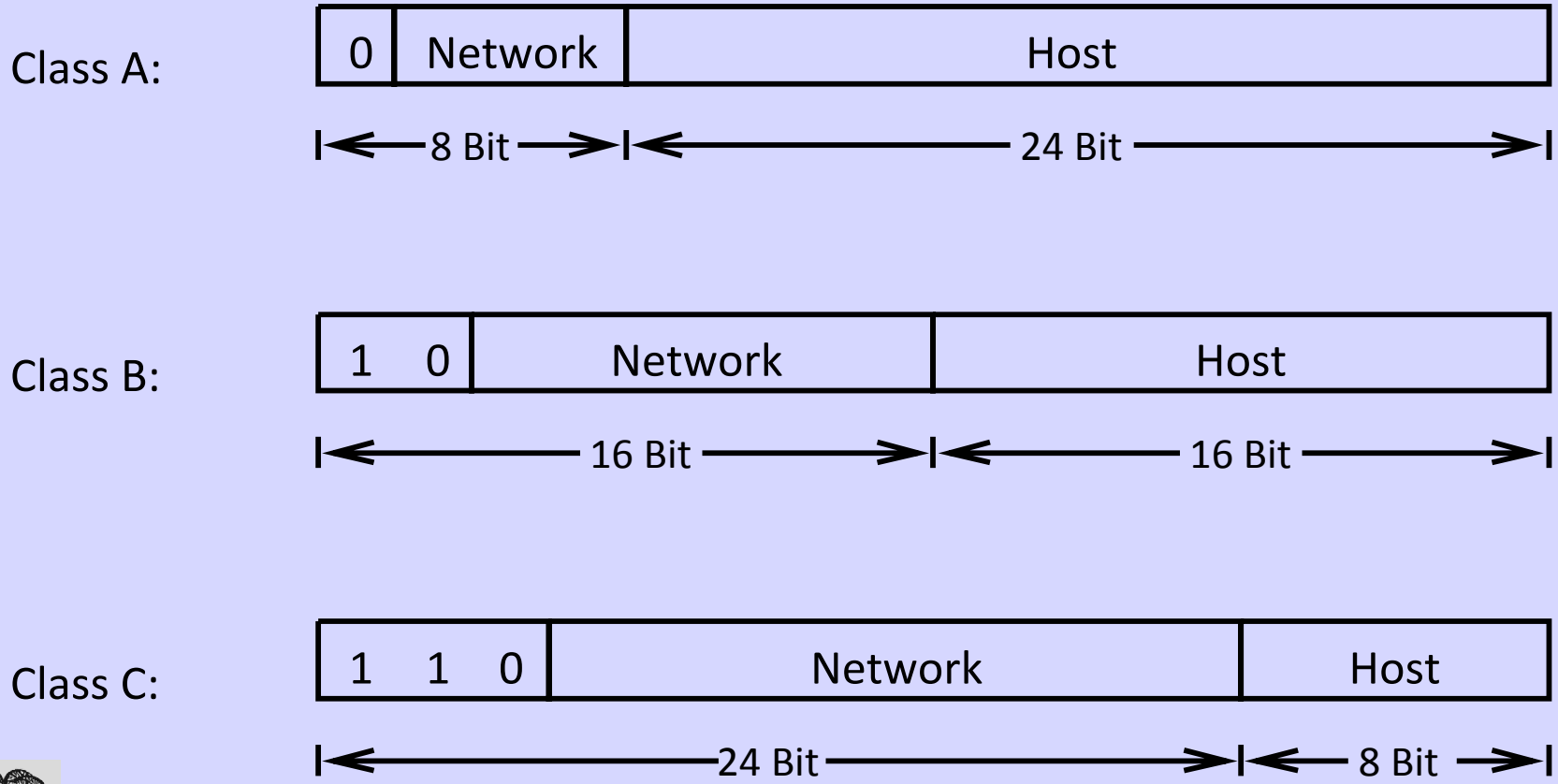
Internet Adressen

- **Internet Host Adresse**
  - = 32-Bit Zahl
  - Adressiert ein Netzwerk und innerhalb dessen einen Rechner
- Vergabe von Internet Adressen
  - Netzadressen von einer zentralen Stelle
  - Rechner-IDs vom lokalen Administrator
- Beispiel
  - chaq = 134.2.8.200 oder morlaix = 134.2.12.6 (siehe /etc/hosts)
- Das ARPANET Adressformat
  - IMP (*Interface Message Processor*) ist eine Schaltstelle im Arpanet
  - *Logical Host* ist ein logischer Zugangspfad zum Host, z.B. in einer Multiplex-Verbindung.

Network	IMP	Logical Host	Host
1 Byte	1 Byte	1 Byte	1 Byte

Internet Adressen

- Die Beschränkung auf 255 Netze bald zu eng.
- Einführung von drei neuen 32-Bit Adressformaten.



Internet Adressen

- Es gibt auch noch Class D und Class E Adressen:
  - Class D: beginnt mit 1110 (und reicht von 224.0.0.0 bis 239.255.255.255)
    - **multicast Adressen**.
  - Class E: beginnt mit 1110 (und reicht von 240.0.0.0 bis 247.255.255);
    - für zukünftige Erweiterungen reserviert
- Es gibt auch spezielle Adressen, z.B.:
  - 0.0.0.0: Dieser Rechner (nur für Boot-Vorgang).
  - 255.255.255.255: Broadcast im lokalen Netz.
  - 255.255.255, a.b.255.255, a.b.c.255: Broadcast in entferntem Netzwerk
    - (je nachdem, ob Class A, B oder C Netzwerk; gilt auch allgemein für Subnetze, siehe unten)
  - 127.x.y.z: loopback Test

Internet Adressen

- Nachteil der Einteilung in Class A, B, C Netze:
  - oftmals suboptimale Netzgröße
- **Subnetze**:
  - Möglichkeit, eine Adresse in einen Subnet-Teil und den Host-Teil aufzuteilen (innerhalb eines A, B, oder C Netzwerkes).
- Beispiel:
  - Class-B Adresse zu *n* Bit und (*16-n*) Bit.
- Netzwerk-Maske definiert, welche Bits zur Network- und Subnet-Adresse gehören.
- Beispiele für Netzwerk-Masken:
  - 255.255.255.0
  - 255.255.248.0

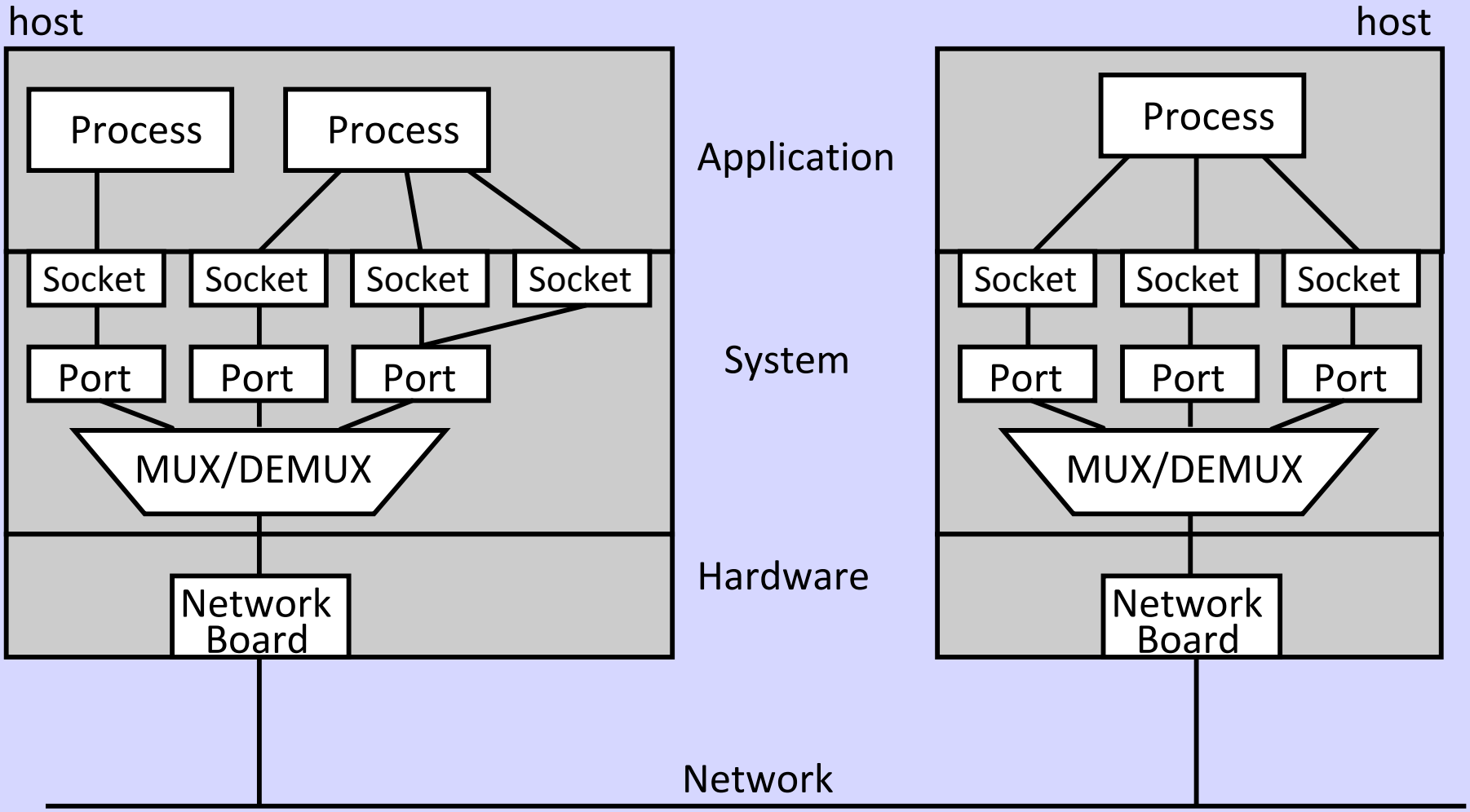
Address Resolution Protokoll (ARP)

- Umsetzung IP-Adresse in Data-Link-Adresse (z.B. Ethernet)
  - Verwendung für Rechner im selben Subnetz
  - ARP-Anfrage-Paket enthält IP-Adresse und wird an alle Rechner im Subnetz geschickt (Broadcast)
  - Rechner mit entsprechender IP Adresse schickt ARP-Antwort-Paket, das die Data-Link-Adresse enthält
- ARP Cache speichert Adressumsetzungen für eine bestimmte Zeitdauer (in der Regel einige Minuten)
- RARP (Reverse Address Resolution Protokoll)
  - Umsetzung Ethernet-Adresse nach IP-Adresse
  - Verwendung bei Rechnern, die übers Netz gebootet werden

Ports

- Verbindung von Prozessen
  - auf der OSI Transportschicht (4)
  - host-bezogene Internet-Adressen müssen deshalb ergänzt werden
- Protokollnummer
  - IP-Protokoll auf Schicht 3 adressiert nur Rechner (*Hosts*), enthält aber im Protokollkopf auch eine 8-Bit Protokollnummer → Identifizierung des Protokolls, an das die Nachricht weitergeleitet werden soll.
- *Ports* (Daten-Pforten)
  - Kennzeichnung durch einen zusätzlichen Integer (16 Bit bei TCP/UDP)
  - Endpunkt der Netzwerk-Kommunikation
- Abbildung von *Ports* zu Prozess-IDs oder Deskriptoren
  - Auslieferung der Nachricht über das lokale Betriebssystem
  - Betriebssystem-Konzept der *Sockets*

Ports


host
host
Process
Process
Process
Application
Socket
Socket
Socket
Socket
Socket
Socket
Socket
Port
Port
Port
Port
Port
Port
System
MUX/DEMUX
MUX/DEMUX
Hardware
Network Board
Network Board
Network

Ports

- Prozess kann mehrere Ports benutzen
- Prozess muss Daten, die am Port ankommen, interpretieren können
  - Zuordnung: Protokoll – Port – Prozess (protocol handler)
- **Well known ports**
  - Wohlbekannte Standardprozesse (z.B. FTP=21) haben auf allen hosts die gleiche Portnummer.
  - 1 – 1023: für den Systemverwalter reserviert,
  - 1024 – 5000: werden auf Anfrage von der Portverwaltung vergeben
  - ≥ 5001: Zuordnung durch den Anwender

Verbindungen

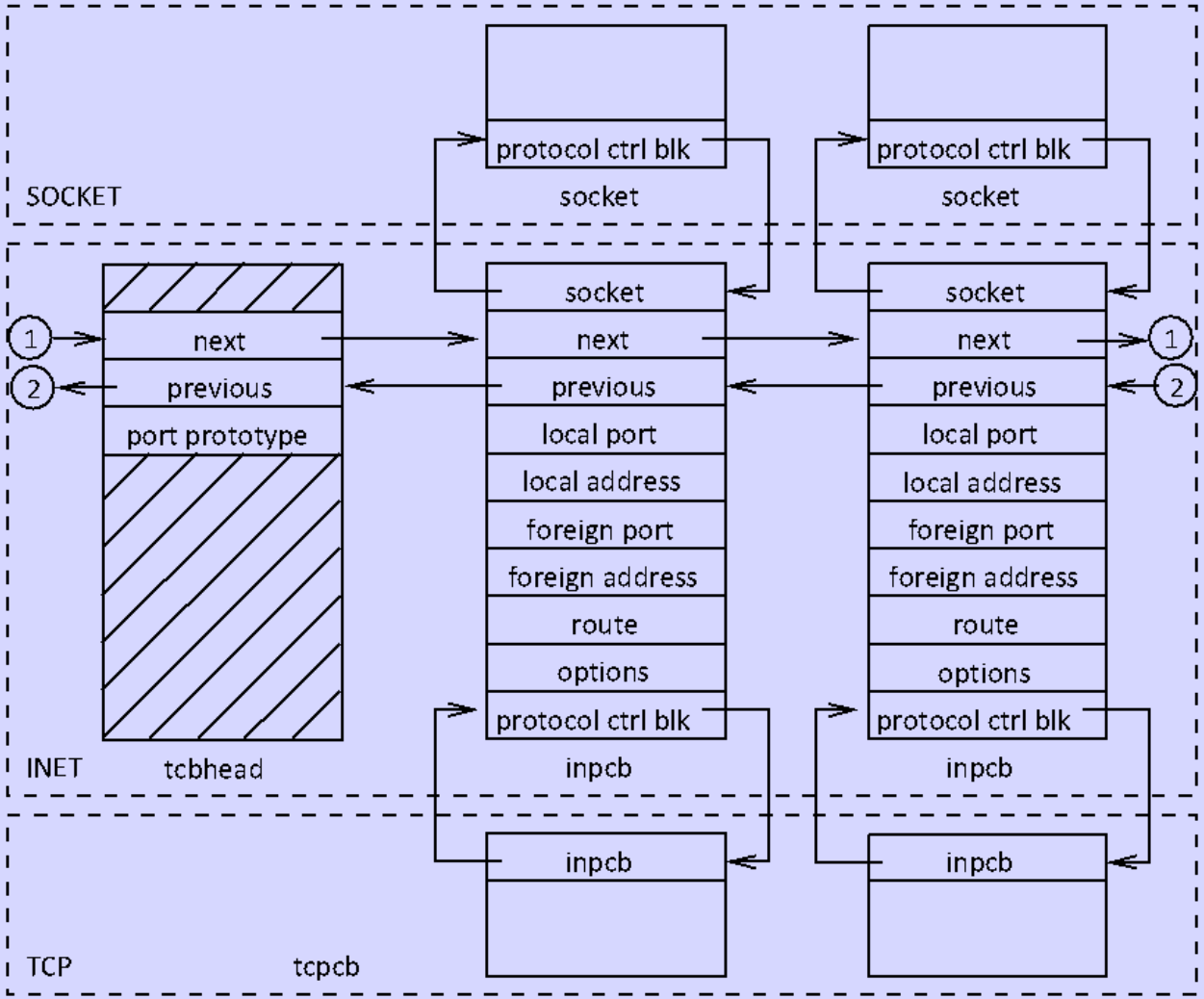
- **Endpunkt einer Verbindung** adressiert als Paar <Internet-Adresse, Port>
- Eine **Verbindung ist charakterisiert** durch: <Protokoll, lokale IA, lokale Port #, ferne IA, ferne Port #>
- Beispiel: (TCP, 134.2.16.7, 1013, 134.2.12.6, 21)
  - TCP-Verbindung zwischen port 1013 auf Utu und port 21 auf morlaix, auf dem das morlaix ftp-Programm Kommandos entgegennimmt.
- Mehrere Verbindungen über den selben Port möglich
  - Beispiel: Web Server

Socket

- UNIX: Ende einer Verbindung ≙ **socket** (Steckdose)
  - (Protokoll, IA, Port #)
- Intuition:
  - Eine Verbindung kommt zustande, wenn ein Netzkabel an beiden Enden in eine Socket gestöpselt ist.
- Behandlung wie eine Datei
  - Auf eine *local socket* werden Daten wie auf eine Datei geschrieben unter Angabe der fernen *socket*.
  - Dort können sie dann wie von einer Datei gelesen werden.

Repräsentation einer Verbindung

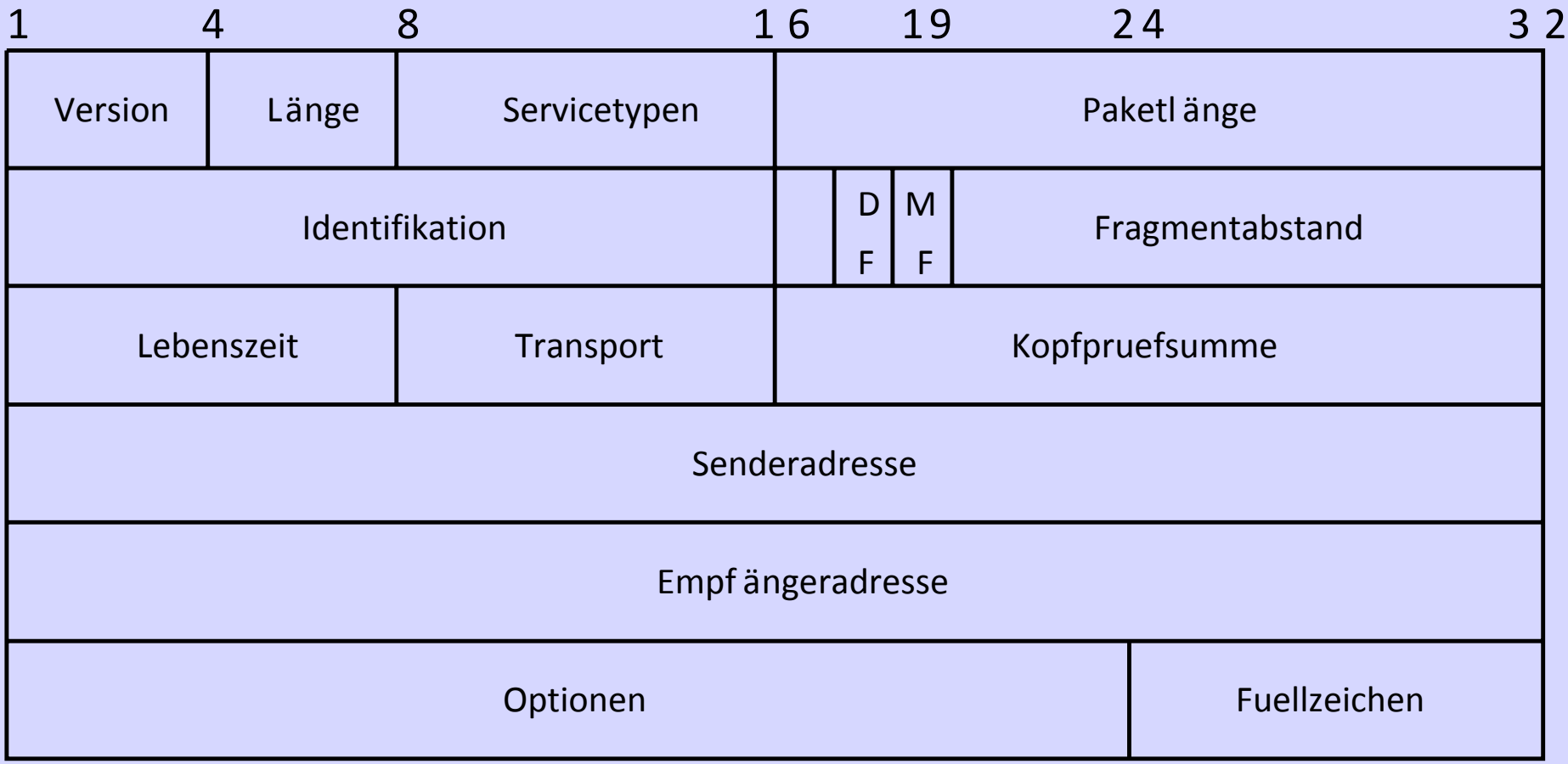
- Interne Repräsentation der Verbindungen durch Leitblöcke
- UNIX BSD: Für jede Organisations-schicht (Socket, Internet, Protokoll) gibt es separate Leitblöcke, die verkettet werden.



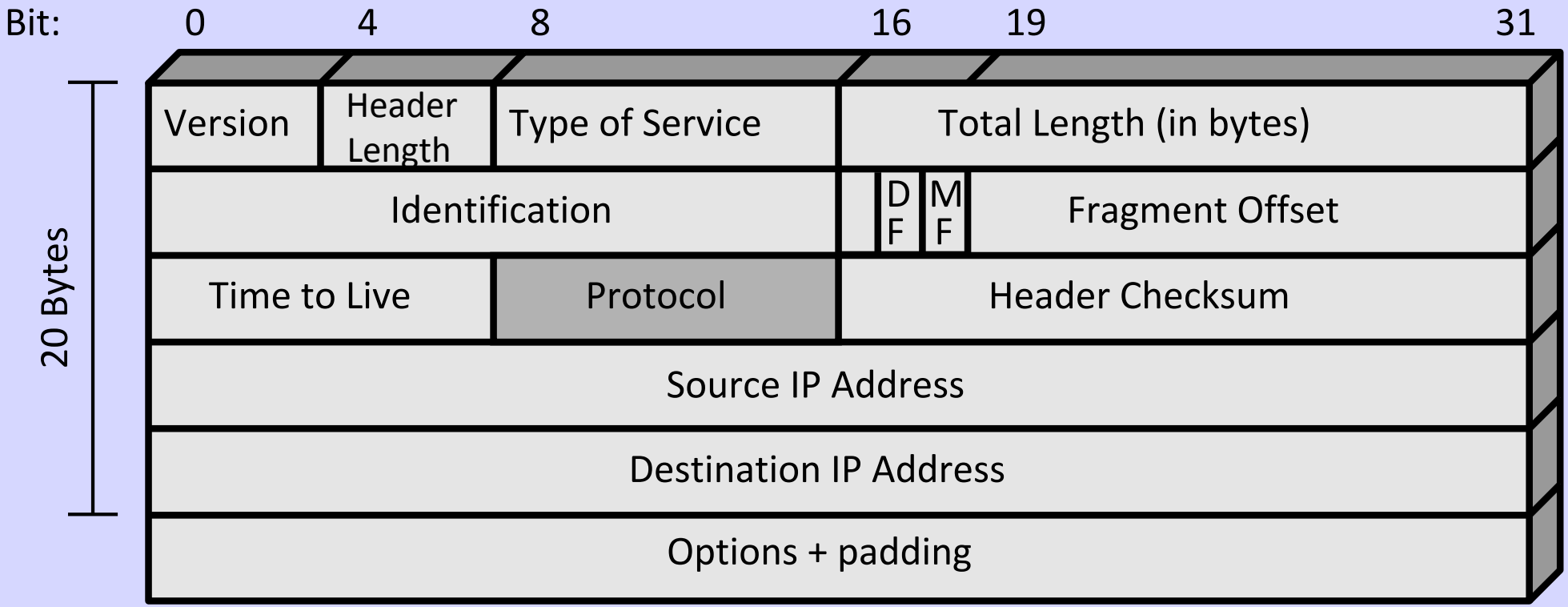
IP (Internet Protocol)

- unsicheres verbindungsloses Protokoll
  - spezifiziert in RFC 791
  - 20 Byte header und maximale Paketlänge von 65535 Byte (= 2 Byte Längenangabe).
- Je nach Schicht 1 und 2 müssen IP Pakete fragmentiert werden
  - Vergabe von Fragmentnummern
  - Genauer: Im Header wird der Offset des Datenfragments innerhalb des Gesamtpakets gespeichert
- IP zerstört Pakete, die sich zu lange im Netz aufhalten → Puffer Überlauf
  - Integer *Lebensdauer*, die bei jedem „Hop“ heruntergezählt wird
  - Lebensdauer == 0 → Paket wird vernichtet
  - Lebensdauer ist in einem Byte gespeichert → maximal 255 Hops

IP (Internet Protocol) – Kopf



IP (Internet Protocol) – Kopf

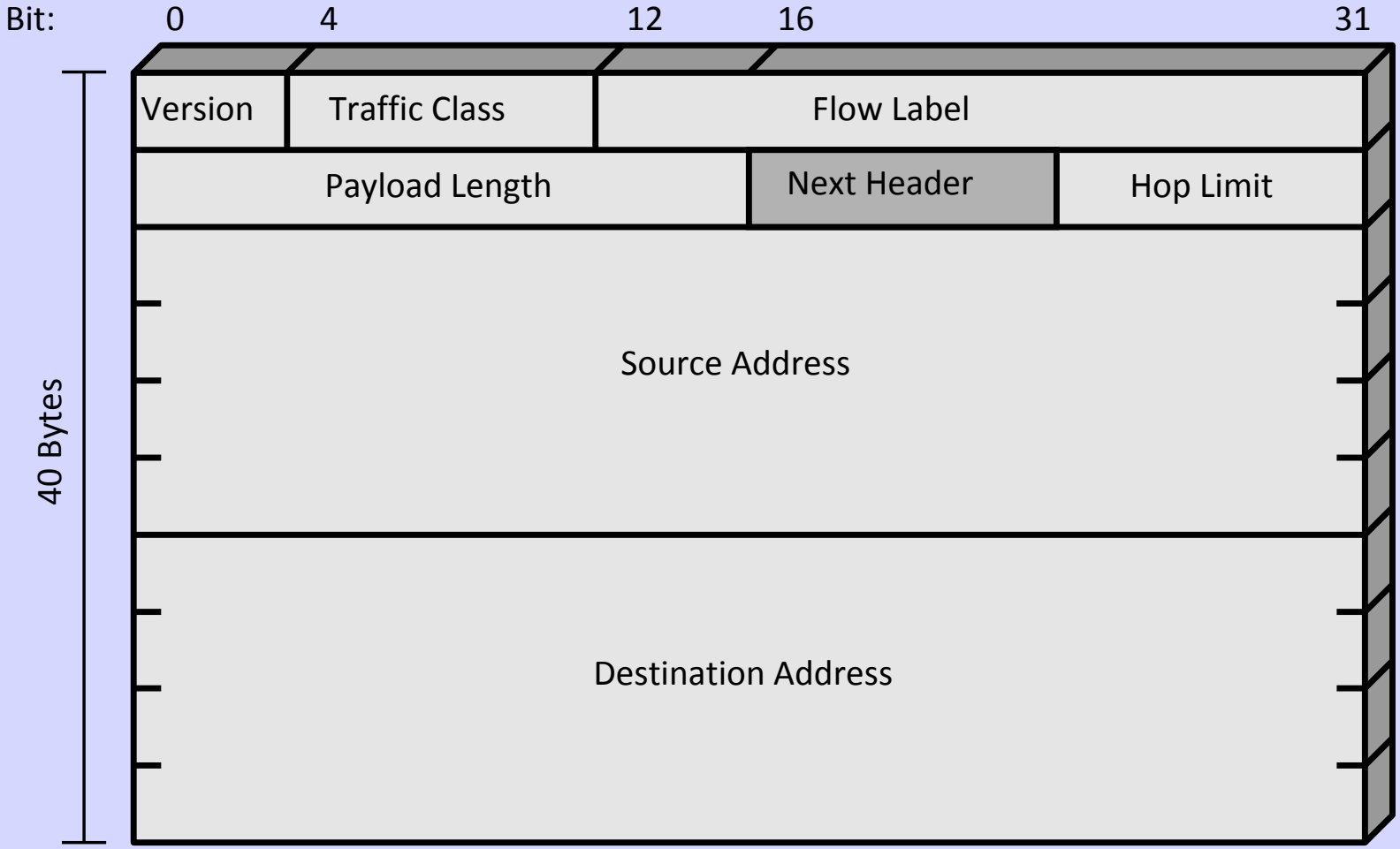


nach: Stallings, OperatingSystems, Abb. A.4

IP (Internet Protocol) – Kopf

- **Type of Service**
  - Bits 0-2 : Präzedenz
  - Bit 3 : normale/geringe Verzögerung
  - Bit 4 : normaler/hoher Durchsatz
  - Bit 5 : normale/hohe Verlässlichkeit
  - Bits 6-7 : reserviert für zukünftige Anwendungen
- **Identification**
  - steuert Zusammensetzen von zuvor fragmentierten IP Datenpaketen
    - zusammen mit Flags und Offset
  - Eindeutige Kennung eines Datagramms
- **Flags**
  - Bit 0 : reserviert (muss 0)
  - Bit 1 : DF (*don 't fragment*): Darf/darf nicht zerlegt (fragmentiert) werden
  - Bit 2 : MF (*more fragments*): Letztes Fragment/weitere Fragmente folgen
- **Protocol**
  - Beispiele: 0x01 = ICMP, 0x06 = TCP, 0x11 = UDP, 0x29 = IPv6, 0x79 = SMP

IP version 6



Stallings, Operating Systems, Abb. A.4

IP version 6

- Bisher beschrieben: IP Protokoll Version 4
- Wichtigste Neuerung: 16 Byte (=128 Bit) Adressen.
- **Version**
  - IP-Versionsnummer
  - Parallelbetrieb von IPv4 und IPv6 möglich
- **Traffic Class**
  - *„available for use by originating nodes and/or forwarding routers to identify and distinguish between different classes or priorities of IPv6 packets.“ (RFC2460)*
- **Flow Label**
  - Zufallszahl
  - *„label sequences of packets for which it requests special handling by the IPv6 routers“ (RFC2460)*
- **Payload Length**
  - Länge des IPv6-Paketinhaltes (nach dem Header)
- **Next Header**
  - Identifiziert den Typ des nächsten Headers
- **Hop Limit** TTL

Versand eines IP Pakets

- Beispiel: IP Ausgabe-Routine in 4.3 BSD

```
error = ip_output(   struct mbuf* m,    // Nachricht
                     struct mbuf* opt,  // Optionen
                     struct route* ro,  // Pfad
                     int flags)
```

  - m: mbuf-Liste mit dem zu sendenden Paket (Nachricht), das schon ein Header-Skelett enthält;
  - opt: optionaler mbuf mit weiteren IP Optionen, die nach dem header eingefügt werden sollen;
  - ro: zeigt optional auf eine Pfadstruktur (rtentry) in einer Routing Tabelle.

Versand eines IP Pakets

- Ablauf:
  - Trage IP Optionen ein
  - Fülle die Kopffelder aus (IP Version, offset 0, header Länge, Paket ID)
  - Stelle den nächsten Schritt im Pfad fest. (Ausgabe-Schnittstelle und Partner-Adresse)
  - Lege fest, ob per broadcast gesendet werden soll.
  - Stelle fest, ob das IP Paket die maximale Paketgröße der Ausgabeschnittstelle (Schicht 2) überschreitet.
    - Wenn nicht, so errechne die Prüfsumme und rufe die Ausgaberoutine der Schnittstelle auf.
    - Wenn ja, fragmentiere in kleinere IP Pakete und schicke die Fragmentpakete nacheinander.

Empfang eines IP Pakets

- Wenn ein Schicht-2-Paket von außen in die Netzwerk-Karte (Netzwerk-Schnittstelle) gelangt ist, verursacht der Controller einen Hardware-Interrupt.
- Die Interrupt-Service-Routine übernimmt das Paket in einen Puffer und kettet diesen in eine Warteschlange ein.
- Danach generiert sie einen Software-Interrupt (*software-initiated interrupt*) niedrigerer Priorität für die weitere Bearbeitung des Pakets durch Protokolle höherer Schichten. Dadurch wird der Prozessor sofort frei für die Übernahme weiterer Pakete.
  - In traditionellem UNIX übernimmt der Interrupt-Controller die Priorisierung
  - In einem multithreaded kernel wie Solaris 10 läuft jede Interrupt-Serviceroutine auf einem Kernel Thread entsprechender Priorität
- Für IP auf 4.3 BSD heißt die Interrupt-Serviceroutine `ipintr()`

Empfang eines IP Pakets

- Ein IP Paket wird auf eine von vier Arten behandelt:
  - Weitergabe an höhere Protokolle
  - Fehlermeldung an den Absender
  - Vernichtung (Fallenlassen, *dropping*) wegen eines Fehlers
  - Weiterleitung auf dem Pfad zum Empfänger

Empfang eines IP Pakets

- ipintr() durchläuft die folgenden Schritte:
  - Prüfe, ob das Paket mindestens die Länge eines IP Kopfs hat und kopiere den Kopf falls nötig in einen zusammenhängenden Datenbereich.
  - Prüfe den Header anhand der Prüfsumme und vernichte das Paket im Fehlerfall.
  - Prüfe, ob das Paket mindestens die im Kopf angegebene Länge hat; schneide evtl. Füllmaterial am Ende ab. Vernichte das Paket im Fehlerfall.
  - Bearbeite IP Optionen. (Z.B. hänge die Zwischenstation an das Pfadprotokoll an.)
  - Prüfe, ob dieser Host der Adressat ist. Falls nicht, leite das Paket weiter, wenn dies möglich ist, sonst vernichte das Paket.
  - Falls das Paket ein Fragment ist, bewahre es auf, bis alle anderen Teile angekommen sind oder bis es zu alt geworden ist. Setze das Original IP-Paket aus den Fragmenten zusammen.
  - Leite das Paket an das nächst höhere Protokoll weiter.

ICMP (Internet Control Message Protocol)

- IP: unsicherer Datagramm Dienst
- ICMP: Alternative, um auf höherer Ebene leichter einen sicheren Dienst einrichten zu können.
- ICMP benutzt IP Pakete und verschickt in deren Datenteil Kontrollnachrichten, etwa
  - Aufforderungen zur Reduzierung der Datenrate,
  - zum Neusenden eines Pakets oder
  - um sich als aktiver Rechner zu erkennen zu geben.
- Beispiel: `ping -s chaq 1000 3`
  - Senden von drei ICMP Paketen zu je 1000 Bytes an den host chaq
  - chaq sendet diese, falls TCP/IP aktiv ist, zurück
  - → Statistik über die Qualität der Verbindung

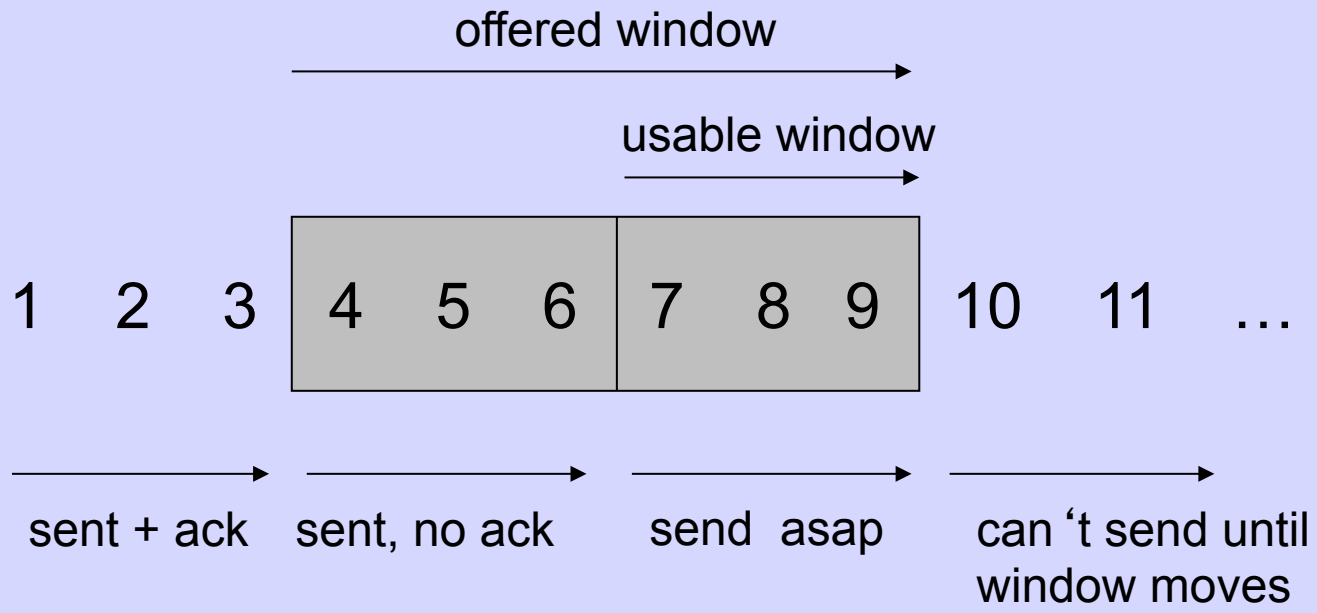
UDP (User Datagram Protocol)

- User Datagram Protocol
  - Datagramm Protokoll der Transportschicht
  - unsicher, aber sehr effizient
  - für höhere Dienste empfehlenswert, falls sie selbst auf einem relativ sicheren physikalischen Netz aufbauen
- UDP wird im wesentlichen direkt auf IP abgebildet
  - Unterschied: UDP adressiert einen Port, IP einen host
- UDP header enthält 8 Bytes:
  - Sender Port (2 Byte)
  - Empfänger Port (2 Byte)
  - Paketlänge (2 Byte)
  - Kopfprüfsumme (2 Byte)

TCP (Transmission Control Protocol)

- sichere bidirektionale Punkt-zu-Punkt-Verbindung zwischen zwei Ports.
- Benutzer sieht logisch einen Bytestrom
- TCP zerlegt den Strom in Segmente, die einzeln als IP Pakete verschickt werden
- Jedes Segment erhält Sequenznummer (Nummerierung der Bytes)
  - Zuverlässigkeit (Korrekte Reihenfolge und Vollständigkeit)
  - Sequenznummern werden um Anzahl der übertragenen Bytes erhöht
- Bytes werden quittiert
  - eine Quittung sagt an, bis zum wievielten Byte korrekt empfangen wurde
- Der Sender sendet fortlaufend
  - die maximale Anzahl noch unquittierter Bytes ist allerdings durch die Größe eines **sliding window** begrenzt.

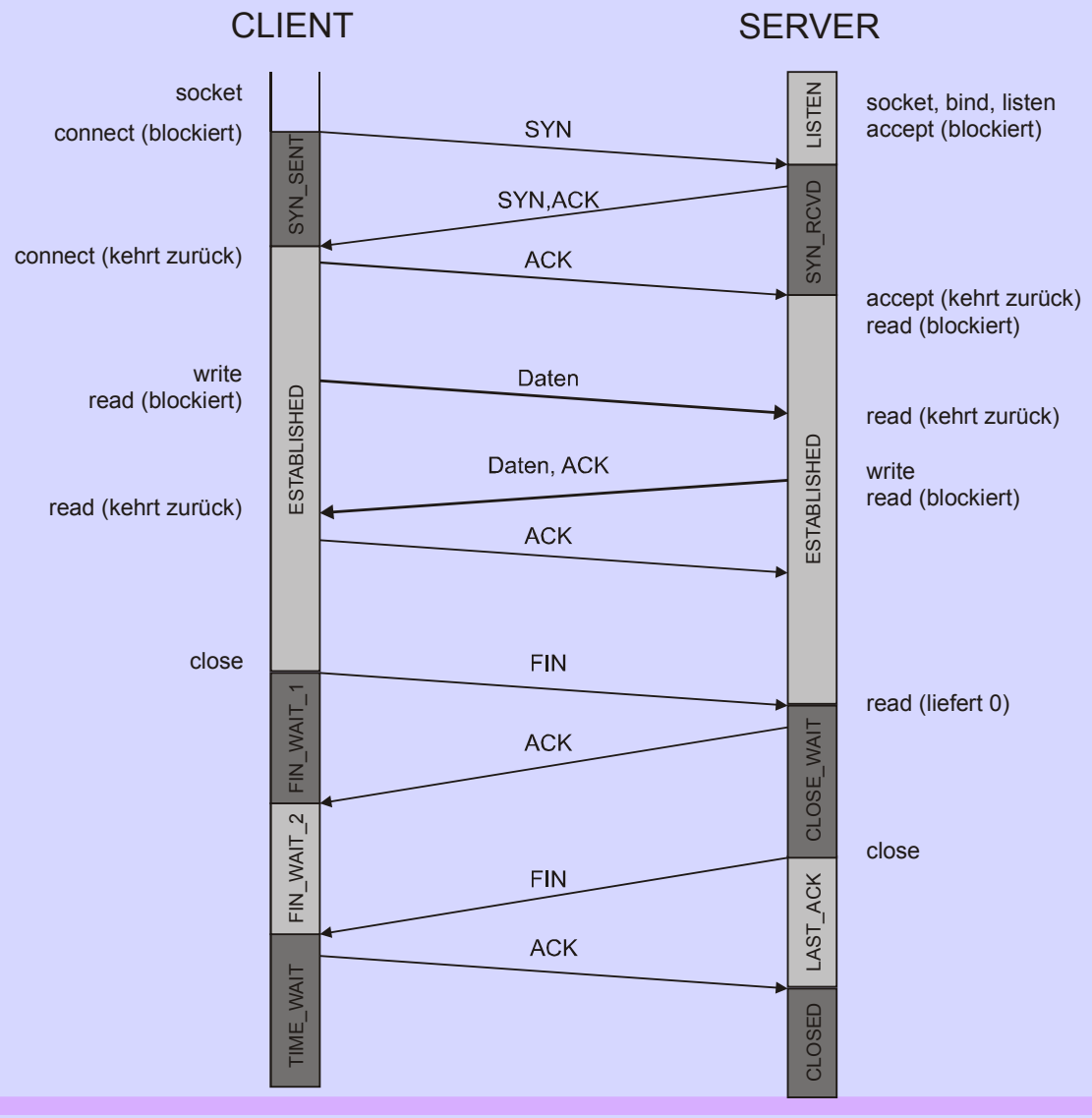
Sliding Window Protocol



TCP (Transmission Control Protocol)

- Der Protokollkopf von 20 Bytes enthält u.a.
  - Sender Port (2 Bytes)
  - Empfänger Port (2 Bytes)
  - Sequenznummer (4 Bytes)
  - Quittungsnummer (4 Bytes)
  - Flags (2 Bytes)
  - Fenstergröße (2 Bytes)
  - Kopfprüfsumme (2 Bytes)
  - Urgent Pointer (2 Bytes)
  - Optionen (2 Bytes)

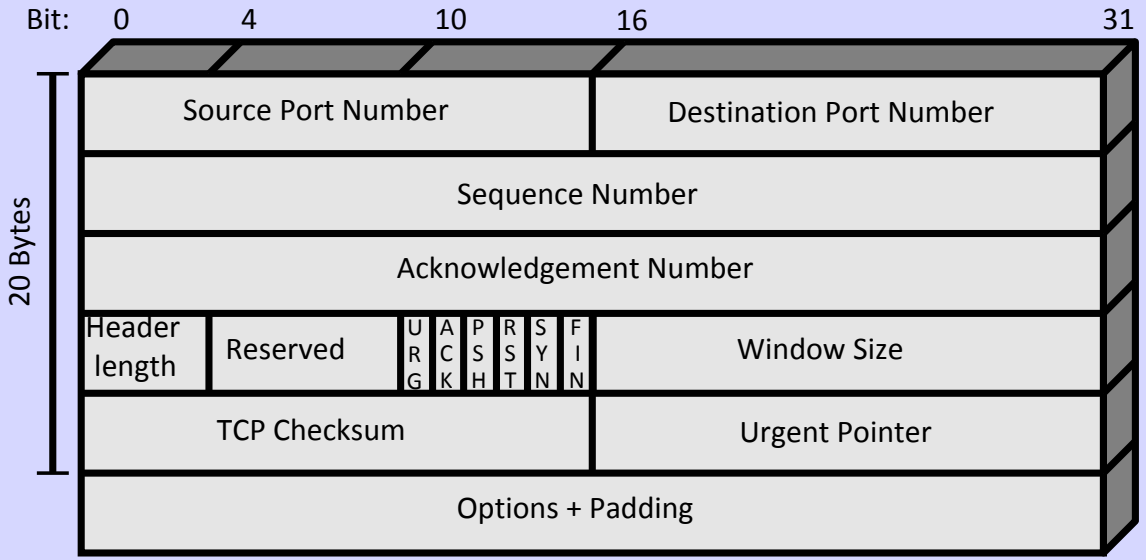
TCP Verbindungsaufbau- und abbau



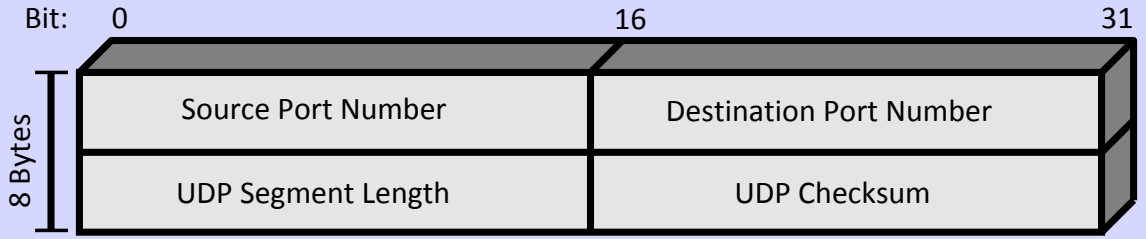
UDP vs. TCP

- UDP: effizient, aber unzuverlässig
- TCP: zuverlässig, aber Protokolloverhead
- UDP muss für broadcast oder multicast Anwendungen verwendet werden
- UDP kann verwendet werden bei
  - Übertragung von Multimedia Daten
    - Fehlende Daten können interpoliert werden
  - Einfachen Protokollen
    - Auf Flusskontrolle, usw. kann verzichtet werden
    - Eigenes Protokoll zur Herstellung der Zuverlässigkeit notwendig
- Sonst Verwendung von TCP angebracht

UDP vs. TCP



(a) TCP Header



(b) UDP Header

nach:
Stallings,
Operating Systems, Abb.
A.3

UDP vs. TCP

- **Sequence Number**
  - nach der Datenübertragung zur Sortierung der TCP-Segmente
- **Acknowledgment Number**
  - Sequenznummer, die der Sender dieses TCP-Segmentes als nächstes erwartet
- **Urgent Pointer**
  - Zeiger auf das Ende einer Sequenz mit dringenden Daten
  - Anfang der Sequenz muss Anwendung selbst ermitteln
- **Flags**
  - **Urgent-Bit**
    - Daten auf die Urgent Pointer zeigt, werden sofort bearbeitet
    - Mechanismus ähnlich Softwareinterrupt
  - **Push-Bit**
    - Daten unter Umgehung des Buffers sofort an Anwendung weiterleiten
  - **Reset-Bit**
    - Abbruch der Verbindung
    - Beispiel: Abweisung von unerwünschten Verbindungen

Verwaltungsdateien

- Verwaltungsinformationen für TCP/IP Netze werden in Textdateien gehalten
- Unter UNIX gibt es die folgenden Verwaltungsdateien:

/etc/protocols

- Format: Name Kodierung Alias Kommentar
- z.B. udp 17 UDP # user datagram protocol

/etc/networks

- Namen der installierten Internet-Subnetze und ihrer Nummern
- loopback-net 127 ist ein Pseudo-Netz zu Testzwecken; es gibt Pakete sofort zurück
- z.B. loopback-net 127 software-loop

/etc/hosts

- Zuordnung Host-Namen und Internet-Adressen mit Aliasen
- Heute: I.a. nur lokale Hosts; die Zuordnung von Host-Namen zu Internet-Adressen geschieht dynamisch via **DNS** (Domain Name System)

/etc/services

- Zuordnung der wohlbekannten Dienste zu ihren Ports mit den verwendeten Protokollen