

Kommunikationstechnik Grundlagen

für dipl. Techniker HF Automation

Autor : Alexander Ott

Edition mit Lösungen

4. Januar 2019: V1.0



Inhaltsverzeichnis

Literaturverzeichnis	v
Glossar	viii
Vorwort	xi
Einleitung	xiii
1 Grundlagen der Kommunikation	1
1.1 Kommunikation	2
1.2 Informationstheorie	4
2 Grundlagen Repetition	9
2.1 Elektrotechnik Grundlagen	10
2.2 Digitaltechnik Grundlagen	16
2.3 Zahlensysteme	30
2.4 Schaltungstechnik	34
2.5 Mathematik	35
3 Die serielle Schnittstelle RS-232 / UART	37
3.1 Grundlagen	38
3.2 LPT : Parallele Schnittstelle	46
3.3 RS-232 : Serielle Schnittstelle	47
3.4 UART : Universal Asynchronous Receiver Transmitter	51
3.5 Fazit	54
4 Standardisierung und das OSI-Kommunikationsmodell	55
4.1 Standardisierung	56
4.2 ISO/OSI Modell	60
4.3 Zusammenfassung	69
5 Kanal / Leiter	71
5.1 Übersicht	73
5.2 Aufbau eines Leitungssystems	75
5.3 Skin-Effekt	77
5.4 Allgemeines Zweidrahtsystem	79
5.5 Twisted Pair	81
5.6 Koaxial Kabel	86
5.7 Lichtwellen-Leiter LWL	88
5.8 Ausbreitungsgeschwindigkeit	90
5.9 Fazit	92
6 Wireless / Drahtlos	93
6.1 Was ist Wireless?	94
6.2 Wellenarten	95
6.3 Was ist eigentlich eine Antenne?	96
6.4 Freiraum Übertragungsleistung	96
6.5 WLAN	100
6.6 RFID	102

6.7 NB-IoT / M2M	102
6.8 Zusammenfassung	104
7 Leitungstheorie	105
7.1 Geschichte	106
7.2 Leiter Impedanz / Admittanz	107
7.3 Reflexionsfaktor	112
7.4 Time-Domain Reflektometer	114
7.5 Elektrische Länge eines Leiters	115
8 UART (2)	117
8.1 Rekapitulation	118
8.2 Leitungs-Abschluss	118
8.3 Fehlererkennung / Error Detection	119
8.4 UART im OSI-Modell	121
8.5 Zusammenfassung	121
9 Leitungscodes	123
9.1 Grundlagen	124
9.2 Übersicht Leitungscodes	127
9.3 Non-Return-to-Zero NRZ	128
9.4 Return-to-Zero RZ	129
9.5 Alternate Mark Inversion (AMI)	130
9.6 High Density Bipolar 3 (HDB3)	130
9.7 Manchester	132
9.8 weitere Codes	134
10 Fehlererkennung	137
10.1 Kanalmodell	138
10.2 Grundlagen	139
10.3 Repetition Code	141
10.4 Parity-Bit	142
10.5 Fehlerhypothese	144
10.6 Cycle Redundancy Check	145
11 Topologie	151
11.1 Netzwerk Topologie	152
11.2 Anisotropie / Richtungsabhängigkeit	158
11.3 Netzwerkausdehnung	162
11.4 Fallbeispiele	162
12 Dataframe	165
12.1 Datenfluss	166
12.2 Dataframe	168
12.3 Adressierung	170
12.4 Dataframe Beispiel Implementationen	172
12.5 Bit-Stuffing	174
13 Fehlerkorrektur	177
13.1 Grundlagen	178
13.2 Hamming Code	179
13.3 BCH-Code	187
13.4 Reed-Solomon Code	189
13.5 Trellis / Viterbi	190
13.6 Turbo-Code	194
14 Zugriffsverfahren	195
14.1 Master-Slave Prinzip	197
14.2 Polling	197

14.3 Token-Passing	198
14.4 Resource reservation	199
14.5 CSMA/CD	199
14.6 CSMA/CA	203
15 Multiplexing	205
15.1 Frequency Division Multiplexing (FDM)	206
15.2 Time Division Multiplexing (TDM)	207
15.3 Code Division Multiplexing (CDM)	208
15.4 Space Division Multiplexing (SDM)	212
15.5 Wavelength Division Multiplexing (WDM)	214
15.6 Multiplexing	215
16 Flusssteuerung	217
16.1 ACK / NACK	218
16.2 RTS / CTS	220
16.3 Feedback Error Control (ARQ)	221
16.4 Verbindungsprotokolle	222
17 OSI Layer 3 : Network Layer	225
17.1 Routing Schemata	226
17.2 Vermittlung von Daten	228
17.3 Verbindung von Netzen	233
17.4 Internet Protocol (IP)	236
18 OSI Layer 4 : Transport Layer	245
18.1 Funktionen des Layers 4 (Transport Layer)	246
18.2 Transport Layer Services	247
18.3 Layer 4 Protokolle bei Ethernet	249
19 Gebäudeinstallation / Automation	257
19.1 Standardisierung	258
19.2 Zonen der Strukturellen Verkabelung	259
19.3 Verteilung nach ISO/IEC 11801	259
20 Epilog	261
Anhang	263
A Datenblätter	263
B Aufgaben-Verzeichnis	269

Literaturverzeichnis

Online

- [O1] *Appendix A : BCH-Code Table.* URL: <http://link.springer.com/content/pdf/bbm:978-1-4899-2174-1/1.pdf>.
- [O2] *Bundesamt für Kommunikation BAKOM : Nationaler Frequenzzuweisungsplan NaFZ.* 2016. URL: <https://www.bakom.admin.ch/bakom/de/home/frequenzen-antennen/nationaler-frequenzzuweisungsplan.html>.
- [O3] *Si86xx : Digitaler Isolator.* URL: <http://www.silabs.com/documents/public/datasheets/Si860x.pdf>.
- [O4] *USB in a Nutshell.* URL: <http://www.beyondlogic.org/usbnutshell/usb1.shtml>.

Wikipedia

- [W1] *Wikipedia : Admittanz.* URL: <https://de.wikipedia.org/wiki/Admittanz>.
- [W2] *Wikipedia : Antennentechnik.* URL: <https://de.wikipedia.org/wiki/Antennentechnik>.
- [W3] *Wikipedia : äquivalente isotrope Strahlungsleistung.* URL: https://de.wikipedia.org/wiki/%C3%84quivalente_isotrope_Strahlungsleistung.
- [W4] *Wikipedia : Bandbreite.* URL: <https://de.wikipedia.org/wiki/Bandbreite>.
- [W5] *Wikipedia : Baudrate.* URL: <https://de.wikipedia.org/wiki/Symbolrate>.
- [W6] *Wikipedia : BCH-Code.* URL: <https://de.wikipedia.org/wiki/BCH-Code>.
- [W7] *Wikipedia : Bitrate.* URL: <https://de.wikipedia.org/wiki/Bitrate>.
- [W8] *Wikipedia : Bitstrom.* URL: <https://de.wikipedia.org/wiki/Bitstrom>.
- [W9] *Wikipedia : Bitwertigkeit.* URL: <https://de.wikipedia.org/wiki/Bitwertigkeit>.
- [W10] *Wikipedia : Bluetooth 5.* URL: https://en.wikipedia.org/wiki/Bluetooth#Bluetooth_5.
- [W11] *Wikipedia : Cat-M1.* URL: https://en.wikipedia.org/wiki/Narrowband_IoT.
- [W12] *Wikipedia : Cycle Redundancy Check : CRC.* URL: https://de.wikipedia.org/wiki/Zyklische_Redundanzpr%C3%BCfung.
- [W13] *Wikipedia : Definition der Informationstheorie.* URL: <https://de.wikipedia.org/wiki/Informationstheorie>.
- [W14] *Wikipedia : Definition der Kommunikationstechnik.* URL: <https://de.wikipedia.org/wiki/Kommunikationstechnik>.
- [W15] *Wikipedia : Die / Wafer (Halbleitertechnik).* URL: [https://de.wikipedia.org/wiki/Die_\(Halbleitertechnik\)](https://de.wikipedia.org/wiki/Die_(Halbleitertechnik)).
- [W16] *Wikipedia : Digitale Revolution.* URL: https://de.wikipedia.org/wiki/Digitale_Revolution.
- [W17] *Wikipedia : elektrischer Energie.* URL: https://de.wikipedia.org/wiki/Elektrische_Energie.
- [W18] *Wikipedia : elektrischer Leistung.* URL: https://de.wikipedia.org/wiki/Elektrische_Leistung.
- [W19] *Wikipedia : elektrischer Strom.* URL: https://de.wikipedia.org/wiki/Elektrischer_Strom.

- [W20] Wikipedia : elektrischer Widerstand. URL: https://de.wikipedia.org/wiki/Elektrischer_Widerstand.
- [W21] Wikipedia : Elektromagnetische Welle. URL: https://de.wikipedia.org/wiki/Elektromagnetische_Welle.
- [W22] Wikipedia : Endianess. URL: <https://de.wikipedia.org/wiki/Byte-Reihenfolge>.
- [W23] Wikipedia : Float. URL: <https://de.wikipedia.org/wiki/Gleitkommazahl>.
- [W24] Wikipedia : Galois-Felder : Endliche Felder. URL: https://de.wikipedia.org/wiki/Endlicher_K%C3%B6rper.
- [W25] Wikipedia : Grenzfrequenz. URL: <https://de.wikipedia.org/wiki/Grenzfrequenz>.
- [W26] Wikipedia : Hamming-Code. URL: <https://de.wikipedia.org/wiki/Hamming-Code>.
- [W27] Wikipedia : Impedanz. URL: <https://de.wikipedia.org/wiki/Impedanz>.
- [W28] Wikipedia : Induktivitat. URL: <https://de.wikipedia.org/wiki/Induktivit%C3%A4t>.
- [W29] Wikipedia : Industrialisierung. URL: <https://de.wikipedia.org/wiki/Industrialisierung>.
- [W30] Wikipedia : Industrie 4.0. URL: https://de.wikipedia.org/wiki/Industrie_4.0.
- [W31] Wikipedia : Information. URL: <https://de.wikipedia.org/wiki/Information>.
- [W32] Wikipedia : Integer. URL: [https://de.wikipedia.org/wiki/Integer_\(Datentyp\)](https://de.wikipedia.org/wiki/Integer_(Datentyp)).
- [W33] Wikipedia : Internet der Dinge. URL: https://de.wikipedia.org/wiki/Internet_der_Dinge.
- [W34] Wikipedia : Ionosphre. URL: <https://de.wikipedia.org/wiki/Ionosph%C3%A4re>.
- [W35] Wikipedia : Kapazitt. URL: https://de.wikipedia.org/wiki/Elektrische_Kapazit%C3%A4t.
- [W36] Wikipedia : Latenz. URL: [https://de.wikipedia.org/wiki/Verz%C3%B6gerung_\(Telekommunikation\)](https://de.wikipedia.org/wiki/Verz%C3%B6gerung_(Telekommunikation)).
- [W37] Wikipedia : Leitungscode. URL: <https://de.wikipedia.org/wiki/Leitungscode>.
- [W38] Wikipedia : Lichtgeschwindigkeit. URL: <https://de.wikipedia.org/wiki/Lichtgeschwindigkeit>.
- [W39] Wikipedia : Lichtwellenleiter. URL: <https://de.wikipedia.org/wiki/Lichtwellenleiter>.
- [W40] Wikipedia : Logikpegel. URL: <https://de.wikipedia.org/wiki/Logikpegel>.
- [W41] Wikipedia : LoRa. URL: <https://en.wikipedia.org/wiki/LoRa>.
- [W42] Wikipedia : OSI Modell : Application Layer. URL: https://en.wikipedia.org/wiki/Application_layer.
- [W43] Wikipedia : OSI Modell : Data Link Layer. URL: https://en.wikipedia.org/wiki/Data_link_layer.
- [W44] Wikipedia : OSI Modell : Network Layer. URL: https://en.wikipedia.org/wiki/Network_layer.
- [W45] Wikipedia : OSI Modell : Physical Layer. URL: https://en.wikipedia.org/wiki/Physical_layer.
- [W46] Wikipedia : OSI Modell : Presentation Layer. URL: https://en.wikipedia.org/wiki/Presentation_layer.
- [W47] Wikipedia : OSI Modell : Session Layer. URL: https://en.wikipedia.org/wiki/Session_layer.
- [W48] Wikipedia : OSI Modell : Transport Layer. URL: https://en.wikipedia.org/wiki/Transport_layer.
- [W49] Wikipedia : OSI-Modell. URL: <https://de.wikipedia.org/wiki/OSI-Modell>.
- [W50] Wikipedia : Parallele Schnittstelle IEEE1284. URL: https://de.wikipedia.org/wiki/Parallele_Schnittstelle.
- [W51] Wikipedia : Parity-Bit. URL: <https://de.wikipedia.org/wiki/Parit%C3%A4tsbit>.
- [W52] Wikipedia : Permeabilitt. URL: https://de.wikipedia.org/wiki/Magnetische_Permeabilit%C3%A4t.

- [W53] Wikipedia : Permitivität. URL: <https://de.wikipedia.org/wiki/Permittivit%C3%A4t>.
- [W54] Wikipedia : Phase-Shift Keying. URL: https://en.wikipedia.org/wiki/Phase-shift_keying.
- [W55] Wikipedia : Polymer optische Faser. URL: https://de.wikipedia.org/wiki/Polymere_optische_Faser.
- [W56] Wikipedia : Quadratur Amplituden Modulation. URL: <https://de.wikipedia.org/wiki/Quadraturamplitudenmodulation>.
- [W57] Wikipedia : radio-frequency identification. URL: <https://de.wikipedia.org/wiki/RFID>.
- [W58] Wikipedia : Redundanz (Kommunikationstheorie). URL: [https://de.wikipedia.org/wiki/Redundanz_\(Kommunikationstheorie\)](https://de.wikipedia.org/wiki/Redundanz_(Kommunikationstheorie)).
- [W59] Wikipedia : Redundanz (Technik). URL: [https://de.wikipedia.org/wiki/Redundanz_\(Technik\)](https://de.wikipedia.org/wiki/Redundanz_(Technik)).
- [W60] Wikipedia : Reed-Solomon-Code. URL: <https://de.wikipedia.org/wiki/Reed-Solomon-Code>.
- [W61] Wikipedia : Reflexionsfaktor. URL: <https://de.wikipedia.org/wiki/Reflexionsfaktor>.
- [W62] Wikipedia : RS-232. URL: <https://de.wikipedia.org/wiki/RS-232>.
- [W63] Wikipedia : Sigfox. URL: <https://de.wikipedia.org/wiki/Sigfox>.
- [W64] Wikipedia : Skin-Effect. URL: https://en.wikipedia.org/wiki/Skin_effect.
- [W65] Wikipedia : spezifischer elektrischer Widerstand. URL: https://de.wikipedia.org/wiki/Spezifischer_Widerstand.
- [W66] Wikipedia : Symbol. URL: [https://de.wikipedia.org/wiki/Symbol_\(Nachrichtentechnik\)](https://de.wikipedia.org/wiki/Symbol_(Nachrichtentechnik)).
- [W67] Wikipedia : Transformator. URL: <https://de.wikipedia.org/wiki/Transformator>.
- [W68] Wikipedia : Trellis-Code. URL: <https://de.wikipedia.org/wiki/Trellis-Code>.
- [W69] Wikipedia : Turbo-Code. URL: <https://de.wikipedia.org/wiki/Turbo-Code>.
- [W70] Wikipedia : Universal Asynchronous Receiver Transmitter. URL: https://de.wikipedia.org/wiki/Universal_Asynchronous_Receiver_Transmitter.
- [W71] Wikipedia : Universal Serial Bus. URL: https://de.wikipedia.org/wiki/Universal_Serial_Bus.
- [W72] Wikipedia : Viterbi-Algorithmus. URL: <https://de.wikipedia.org/wiki/Viterbi-Algorithmus>.
- [W73] Wikipedia : Volt. URL: <https://de.wikipedia.org/wiki/Volt>.
- [W74] Wikipedia : Weightless. URL: [https://en.wikipedia.org/wiki/Weightless_\(wireless_communications\)](https://en.wikipedia.org/wiki/Weightless_(wireless_communications)).
- [W75] Wikipedia : zweite Industrielle-Revolution. URL: https://de.wikipedia.org/wiki/Zweite_industrielle_Revolution.

Glossar

Allgemein		
Anisotropie	griechisch für Drehung, Richtung. In unserem Zusammenhang die Richtung einer Verbindung (Duplex / Simplex)	
ANSI	American National Standards Institute	
CCITT / ITU-T	Comite Consultatif International Telegraphique et Telephonique. Ist in der ITU-T aufgegangen	
de-facto / IndustrieStandard	Standard, welcher durch seine Verbreitung oder durch Verknüpfung an ein spezielles Verfahren untrennbar verbunden ist	
dissipiert	Technische Bezeichnung für den Umwandlungsprozess von Energie. zB. Umwandlung von elektrischer Energie in Wärme	
false-positiv	Etwas als richtig bewerten, obwohl es falsch ist	
gender	Englische Bezeichnung für Geschlecht. Wird im Zusammenhang mit der Stecker «Polarität» verwendet	
IEEE	Institute of Electrical and Electronic Engineers	
Ionosphäre	Die Ionosphäre ist jener Teil der Atmosphäre der große Mengen von Ionen und freien Elektronen enthält.	[W34]
ISO	Internationale Standardization Organisation	
ITU	International Telecommunications Union	
offener Standard	Standard, welcher Allgemein zugänglich oder auch also Lizenzierbar ist	
Physical Interface	Stecker, Pinbelegung	
proprietärer Standard	Standard, welcher nur Firmenintern und nicht offen ist	
Communication		
B_d	Symbolrate / Baud / [1/s]	[W5]
R_b	Bitrate / [bit/s]	[W7]
T_s	Symboldauer / [s]	
BCH	Ein fehlerkorrigierenden Code, die Berechnung ähnlich der <i>Cycle Redundancy Check</i> Berechnung	
Bitstuffing	Wird angewandt um ein Start-/Stop-Symbol in den Sendedaten zu maskieren	
Block-Code	Ein fehlerkorrigierenden Code, welcher immer auf einen ganzen Block Daten angewandt wird	
Bodenwelle / direkte Welle	Eine EM-Welle, welche über den Boden gleitet und nicht in den Raum strahlt	
client	engl. für Kunde / in der Kommunikationstechnik auch für einen Kommunikationsteilnehmer verwendet	
Convolution-Code	Ein fehlerkorrigierenden Code, welcher auf einen kontinuierliche Datenstrom angewandt wird	
CRC	Cycle Redundancy Check. Wird zur Fehlererkennung benutzt	
Crosstalk	Nennt man den Störeinfluss einer Leitung auf eine zweite Leitung bei der Datenübertragung	
Dataframe	Ein OSI-Layer 2 Frame, mit zB. Start-/Stop-Symbol	
Datagram	Ein OSI-Layer 3 Datagram, zB. ein IP-Packet	
Duplex	Nachrichten können gleichzeitig gesendet und empfangen werden	
EM-Welle	Elekromagnetische Welle : zB. Licht oder Funk	[W21]
Halbduplex	Nachrichten können abwechselnd gesendet und empfangen werden	
Hamming-Code	Ein einfacher fehlerkorrigierenden Code	
Header	Wird auch als «Protokollinformation» bezeichnet. Sie enthält selbst keine Daten, aber Zusatzinformationen, wie Absender- und Empfänger-Adresse etc.	
Hub (Netzwerk)	Client im Zentrum einer Stern-Topologie	
Interface	engl. für Schnittstelle	
ISM-Band	industrial, scientific and medical (ISM) Frequenzband. Für den Betrieb eines Gerätes im ISM Band muss das Frequenzband nicht lizenziert werden (royalty free)	
Layer	engl. für Abstraktionsschicht	
Leitungscode	Leitungscode : Vorschrift wie ein «Symbol» übertragen wird. (Spannungsspeigel/Schaltzeiten etc.)	
LWL	Lichtwellen-Leiter. Optische Faser, meist Glasfaser	[W39]
MAC-Adresse	engl. für «Media Access Control Address». Addressierung für LAN IEEE802	

Netzwerk	Zusammenschluss mehrerer Kommunikationsteilnehmer	
node (Netzwerk)	Client in einer Ast-Verteilung in einer Baum-Topologie	
parity-bit	Binäre Quersumme eines Datenworts. Wird zur Fehlererkennung benutzt	
POF	Polymer optische Faser : «Plastik-Glasfaser»	[W55]
point-to-point	engl. für Punkt-zu-Punkt Verbindung	
Protokoll	Bezeichnet eine definierte Verfahren, wie zwischen zwei (entfernte) Schichten auf dem gleichen OSI-Layer miteinander kommunizieren.	
Raumwelle	EM-Welle, welche in der Raum abgestrahlt wird	
root / Wurzel (Netzwerk)	Der unterste Client einer Baum-Topologie	
Service Access Point (SAP)	Jede ISO/OSI Schicht stellt der darüberliegenden Schicht einen Dienst zur Verfügung. Dies wird der SAP genannt.	
shared (Voll-) Duplex	Duplex Verbindung auf einem Shared-Medium	
shared Halb-Duplex	Halb-Duplex Verbindung auf einem Shared-Medium	
shared Simplex	Simplex Verbindung auf einem Shared-Medium	
Shared-Medium	Bezeichnet ein Verbindungs-Medium an welchem mehrere Teilnehmer verbunden sind	
Simplex	Nachrichten können nur in eine Richtung gesendet werden	
Symbol	Das Datenwort, welches pro Taktflanke übermittelt wird.	[W66]
Trellis / Viterbi	Ein komplexer fehlerkorrigierenden Code, welcher sich «erholen» kann	

Digitaltechnik

V_+ / V_{cc} / V_{dd}	Positiver Anschluss der Versorgungsspannung von Digitalen Schaltungen	
V_- / V_{ee} / V_{ss}	Negativer Anschluss der Versorgungsspannung von Digitalen Schaltungen	
Glitch	engl. für ein kurzes nicht erwünschtes Signal einer Digitalschaltung	
latch / latching	engl. für Verschluss / Einrasten. «to latch a register» : Ein Register laden	
Propagation Delay	engl. für Signal-Durchlaufzeit	
Threshold	engl. für Schwellwert	

Elektrotechnik

ϵ_0	Dielektrizitätskonstante / $\frac{F}{m} = \frac{As}{Vm}$	[W53]
ϵ_r	Dielektrizitäts-Zahl / 1	[W53]
Γ	Reflexionsfaktor / [1] / [1]	[W61]
μ_0	Permeabilitäts-Konstante / $\frac{H}{m} = \frac{Vs}{Am}$	[W52]
μ_r	Permeabilitäts-Zahl / 1	[W52]
ρ	spez. elektrische Widerstand / $\Omega \frac{m^2}{m}$	[W65]
B	Bandbreite / [Herz] / [Hz]	[W4]
C	Kapazität / [Farad] / [F]	[W35]
c	Die Lichtgeschwindigkeit $c \approx 300000 \frac{km}{s}$	[W38]
E	Energie / Arbeit / [Watt · sec] / [E]	[W17]
f_g	-3 dB Grenzfrequenz / [Herz] / [Hz]	[W25]
I	Strom / [Ampère] / [A]	[W19]
L	Induktivität / [Henry] / [H]	[W28]
P	Leistung [Watt] / [W]	[W18]
R	Widerstand / [Ohm] / [Ω]	[W20]
U	Spannung / [Volt] / [V]	[W73]
Y	komplexe Leitwert / [Siemens] / [S]	[W1]
Z	komplexe Impedanz / [Ohm] / [Ω]	[W27]
EIRP	äquivalente isotrope Strahlungsleistung / [Watt] / W	[W3]
Thermisches Rauschen	Spannungsrauschen, hervorgerufen durch die zufällige Verschiebung von Elektronen in einem Metallgitter	

Informatik

Bitwertigkeit	Die Reihenfolge wie die einzelnen Bit in einem Byte gewertet werden.	[W9]
COM / RS-232	Serielle Schnittstelle aus den 1960er Jahren	[W62]
Endian	Big- / Little-Endian, gibt an in welcher Reihenfolge die Daten übermittelt oder im Speicher abgelegt werden.	[W22]
Float	Bezeichnung für eine Gleitkomma-Zahl, aber auch für einen Datentyp welcher eine Gleitkomma-Zahl repräsentiert	[W23]
GAN	Global Area Network	

Integer	Bezeichnung für eine Ganzzahlige-Zahl, aber auch für einen Datentyp welcher eine Ganze Zahl repräsentiert, lat. numeri integri	[W32]
LAN	Local Area Network. Synonym für eine Ethernet basierte Schnittstelle	
LPT / Centronics	Parallele Schnittstelle aus den 1970er Jahren	[W50]
MAN	Metropolitan Area Network	
PAN	Personal Area Network	
UART	gleich wie RS-232 mit anderen Signalpegeln	[W70]
USB	Universal Serial Bus	[W71]
WAN	Wide Area Network	

OSI-Modell

Application Layer	OSI Layer 7	[W42]
Data Link Layer	OSI Layer 2	[W43]
Network Layer	OSI Layer 3	[W44]
OSI-Modell	Standardisierte Form eines Kommunikationssystems	[W49]
Physical Layer	OSI Layer 1	[W45]
Presentation Layer	OSI Layer 6	[W46]
Session Layer	OSI Layer 5	[W47]
Transport Layer	OSI Layer 4	[W48]

Vorwort

Ich habe dieses Skript während meiner Zeit als Dozent an der **Höheren Fachschule Uster** im Fach **Grundlagen Kommunikation** für den Studiengang **dipl. Techniker HF Automation** geschrieben.
Das Skript war nicht Teil der Entlohnung und wurde in meiner Freizeit erstellt.

Es sind vermutlich noch einige Rechtschreib-Fehler enthalten,
wer möchte kann mir diese per E-Mail melden.
alexander.ott.ottale@gmail.com

Lizenz

Dieses Dokument wurde unter der Creative Commons Lizenz **CC-BY-SA** veröffentlicht.

- **CC** : Creative Commons Lizenz
- **BY** : Nennung des Autors
- **SA** : Share-Alike

Das heisst, das eine auf diesem Dokument aufbauende Arbeit den ursprünglichen Autor nennen muss und das Erzeugnis ebenfalls unter gleiche Lizenz veröffentlicht werden muss.

weiter Informationen unter:

- creativecommons.org
- creativecommons.ch

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License.
To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/4.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.



Einleitung

Die Erfindung der Dampfmaschine gilt als eine der wichtigsten Erfindungen der (ersten) Industriellen Revolution [W29]. Die Nutzung der Elektrizität war der Beginn der zweiten [W75] und Henry Ford läutete mit der Einführung des Fliessbandes die dritte Industrielle Revolution ein.

Laut Wikipedia wird die Digitale Revolution [W16] auf ca. 2002 datiert. Man nimmt an, dass zu dieser Zeit mehr Daten in digitaler als in analoger Form gespeichert wurden.

Wegbereiter hierfür war die Erfindung des Transistors und deren Integration zu komplexen Schaltkreisen, aber es sei auch die Erfindung des Internets genannt. Für manche gilt die Erfindung des Internets als Ursprung für die vierte Industrielle Revolution. Von der Deutschen Bundesregierung auch «Industrie 4.0» [W30] genannt.

Meiner persönlichen Meinung nach, sind wir noch nicht an einer vierten Industriellen Revolution angegangt. Aber die Vernetzung jeglicher Gegenstände und Gerätschaften schreitet unermüdlich voran.

Wir leben heute in der Zeit des «Internet of Things» [W33] (IoT, Internet der Dinge). Geht es nach der Industrie und Politik, wird schon bald jedes noch so kleine Gerät an das Internet angeschlossen sein. Zum Beispiel erfährt die Kaffeemaschine von der Smartwatch wann wir aufstehen und kann schon mal den Kaffee brühen. Die Kaffeekapsel kennt seine perfekte Brühtemperatur.

Das Auto ist vom Terminplaner des Smartphones informiert, dass es heute zum Arzt und nicht ins Geschäft geht. Oder aber in der Produktion. Jedes Werkstück kennt seine Bearbeitungsschritte. Der Bohrer kennt seine optimale Drehzahl für die verschiedenen zu bearbeitenden Materialien und sein Betriebsstundenzähler informiert das «Fussvolk» wann denn der Bohrer ausgewechselt werden möchte.

Diese Szenarien sind (leider) nicht mehr all zu fern.

Ich hoffe sie werden sich nicht genau so entwickeln, aber eines ist klar: Der *dipl. Techniker/in HF Automation* von heute wird sich mehr und mehr in der Welt der Informatik und Kommunikationstechnik auskennen müssen.

Damit Sie sich in Zukunft auch in diesem Gebiet zurechtfinden, möchte ich mit Ihnen die Grundlagen der Kommunikationstechnik erarbeiten. Wir werden erkennen, dass in der Welt der Kommunikationstechnik viele Standards eingehalten werden müssen und viele Schnittstellen dennoch nur rudimentär spezifiziert sind um möglichst flexibel einsetzbar zu sein. Außerdem werden wir sehen, dass *fehlerbehaftete Übertragungskanäle* mit einem *Protokoll* «gehärtet» werden können, so dass sogar systemkritische Informationen darüber verteilt werden können.

Zu guter Letzt werden wir die Grundlagen zu den nachfolgenden Kursen *Netzwerk Grundlagen* und *Feldbusse* erarbeiten.

1

Grundlagen der Kommunikation

Inhalt des Kapitel

1.1	Kommunikation	2
1.1.1	Einfaches Modell	2
1.1.2	Sender	2
1.1.3	Empfänger	2
1.1.4	Kanal	3
1.1.5	einfaches Sender- / Empfänger-System	3
1.1.6	Kommunikationstechnik	3
1.2	Informationstheorie	4
1.2.1	Information	5
1.2.2	Redundanz	5
1.2.3	Entropie	5
1.2.4	Verwaltungs-Kosten / Overhead	6
1.2.5	Redundanz in der japanischen Sprache	7

Zitat:

Any sufficiently advanced technology is indistinguishable from magic

Jede ausreichend fortgeschrittene Technologie ist nicht von Magie zu unterscheiden

Arthur C. Clarke : Schriftsteller

Das Ziel dieses Moduls ist es ihnen die Grundlagen der Kommunikation beizubringen.
Es stellt sich somit die Frage: Was ist Kommunikation und wie können wir diese als ein technisches Problem modellieren?

1.1 Kommunikation

Als Kommunikation bezeichnen wir ganz Allgemein den Vorgang, wie wir Informationen von einem Teilnehmern zu einem zweiten Teilnehmer (oder einer ganzen Gruppe) übermitteln.

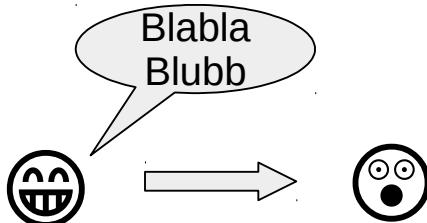


Abbildung 1.1: Kommunikation

1.1.1 Einfaches Modell

Wir können nun ein sehr einfaches Modell der Kommunikation erstellen:

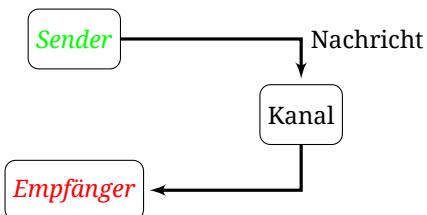


Abbildung 1.2: Einfaches Kommunikationsmodell

Wir erkennen in Abbildung 1.2 einen Sender, welcher eine Nachricht über einen Kanal zu einem Empfänger schickt.

Wir werden im weiteren Verlauf dieses Kurses die einzelnen Blöcke genauer untersuchen und unser Modell erweitern, so dass wir die Grundlagen erarbeiten und die Funktion der heutigen Kommunikations-Systeme verstehen.

1.1.2 Sender

Der **Sender** beinhaltet alles was es braucht um eine «Information» von einem Ort in den Kanal zu transferieren.

Stellen wir uns das ganze einmal folgendermassen vor:

Wir haben eine Vase, welche wir in einem Paket an eine Person zusenden wollen. Der Inhalt des Pakets (die Vase) ist unsere «Information». Wir mussten vorgängig unsere «Information» einpacken, das Adressfeld ausfüllen und das Packet auf die Post bringen. (ev. noch eine Marke aufkleben)

1.1.3 Empfänger

Analog dazu enthält der **Empfänger** alles um die «Information» aus dem Kanal zu extrahieren und die «Information» wieder in die Ursprüngliche Form zu bringen und schlussendlich am Ziel (Der Senke) abzugeben.

Bleiben wir beim Paket-Beispiel:

Das Paket wird an der Haustüre abgeliefert. Wir müssen das Paket in Empfang nehmen, eventuell für den Erhalt unterschreiben. Danach können wir das Paket ins Haus nehmen, auspacken und den Inhalt geniessen.

1.1.4 Kanal

Der Kanal ist das Transportmedium für unsere «Information». Wir modellieren in der Regel alle Stör-einflüsse auf den Kanal.

Um beim Paket-Beispiel zu bleiben, sie haben es vielleicht bereits geahnt: Der Kanal ist die Post. Die Post liefert uns (sofern richtig Frankiert und Adressiert) das Paket von der Abgabestelle bis zum Ziel-Ort. Auch die Post ist nicht perfekt. Ein Paket kann falsch zugestellt werden, falls die Adresse darauf falsch ist, oder weigert sich bei einer falschen Frankierung gleich ganz das Paket zu übermitteln. Wir haben seltener Fälle von höherer Gewalt, bei dem das Paket verloren geht. Wir werden uns im Kurs intensiv mit dem Kanal befassen. In der Regel sind dies Kupferkabel oder Glas-faserkabel, aber auch Funk.

1.1.5 einfaches Sender- / Empfänger-System

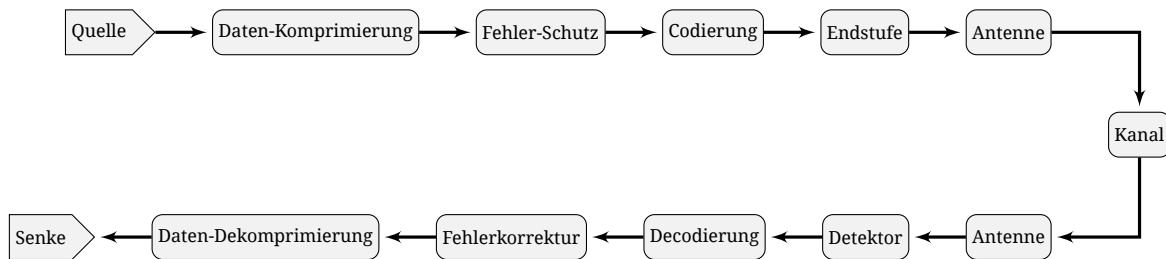


Abbildung 1.3: Beispiel eines Sender- / Empfänger-Systems

In Abbildung 1.3 ist ein einfaches Beispiel eines Sender- / Empfänger-Systems abgebildet. Wir sehen zum einen, dass der Sende- und Empfangs-Teil symmetrische aufgebaut ist. Ist beim Sendeteil eine Daten-Komprimierung implementiert, muss auch der Empfangsteil (in der Regel an der selben Stelle) die inverse Funktion, also eine Daten-Dekomprimierung implementieren. Das gleiche gilt natürlich auch für die Fehlerkorrektur, Daten-Codierung etc.

Der letzte Block (in diesem Beispiel die Antenne) sorgt für die Anbindung an den Kanal.

1.1.6 Kommunikationstechnik

Info

Als Kommunikationstechnik (oder: Kommunikationstechnologie) bezeichnet man zusammenfassend Techniken für die technisch gestützte Kommunikation. Für die Telekommunikation, z.B. Mobilkommunikation, Satellitenkom-munikation und Fernsprechen sind dies die Nachrichtentechnik, Funktechnik, Vermittlungstechnik, Übertragungstechnik, Hochfrequenztechnik, Mikroelektronik, Technische Informatik und Drucktechnik.

Info 1: Definition der Kommunikationstechnik [W14]

Wie wir in Info (1) lesen können, beschreibt die Kommunikationstechnik den technischen Aspekt der Kommunikation.

Aufgabe 1.1 : verbale Kommunikation

Frage

Wenden Sie das Modell aus *Abbildung 1.2* auf die menschliche verbale Kommunikation zweier Personen (**A / B**) an. (Sprache)

- Was ist der Sender? Wie wird die Nachricht übertragen?
- Welches physikalische Medium ist unser Kanal? Wie wird die Nachricht übertragen?
- Was ist unser Empfänger? Wie verarbeitet die Person B die Nachricht?
- Benennen Sie Störeinflüsse welche die Kommunikation beeinflussen können.

Antwort

- Das Gehirn von Person **A** ist unser Sender. Es erzeugt elektrische Impulse, welche an die Muskeln im Gesicht, Kehlkopf und Atmungssystem gesendet wird. Diese wiederum erzeugen Schallwellen welche an die Umgebungsluft abgegeben werden.
- Das physikalische Medium der verbalen Kommunikation ist natürlich die Umgebungsluft. Die Schallwellen breiten sich durch den Raum aus und werden vom Hörorgan von Person **B** empfangen.
- Der Empfänger ist schlussendlich das Gehirn von Person **B**. Damit diese die Schallwellen verarbeiten kann, braucht es ein System, welches die Schallwellen in elektrische Impulse umwandelt. Dieses System besteht aus Ohrmuschel, Trommelfell, Steigbügel, Amboss, Hammer und Hörschnecke. Diese wandeln die Schallwellen wieder in elektrische Impulse um, welche an das Gehirn von Person **B** weitergeleitet werden
- Störeinflüsse könnten zum Beispiel andere Personen sein, welche gleichzeitig reden, aber auch Lärm im Allgemeinen. Eher exotisch ist das Fehlen des Kanal-Mediums, sprich der Umgebungsluft. Dies tritt eigentlich nur außerhalb der Erdatmosphäre auf.

1.2 Informationstheorie

Wir werden ebenfalls einen Einstieg in die Informations-Theorie wagen.

Hierzu müssen wir zunächst die Grundlegenden Begriffe kurz kennen lernen.

Info

Die Informationstheorie ist eine mathematische Theorie aus dem Bereich der Wahrscheinlichkeitstheorie und Statistik, die auf Claude Elwood Shannon zurückgeht. Sie beschäftigt sich mit Begriffen wie Information, Entropie, Informationsübertragung, Datenkompression, Kodierung und verwandten Themen.

Info 2: Definition der Informationstheorie [W13]

Damit wir uns ein etwas handfestes Vorstellen können: Wenn wir etwas versenden, zum Beispiel einen Brief, so enthält der Brief einige Dinge welche unser Empfänger nicht benötigt. Das ist das Briefpapier, der Briefumschlag.

Aber auch auf dem Brief sind einige Dinge, welche der Empfänger nicht wirklich benötigt. Beispielsweise die Anrede-Floskel: «Sehr geehrter...». Diese Floskel enthält nur wenig Wissenswertes, es enthält somit wenig Information. Der Text im Brief ist die Information, welche unser Empfänger interessiert.

Info

Die Gesamtheit des Briefes, welches wir versenden, resp. die gesamten Daten welche wir im späteren Verlauf des Kurses versenden, nennen wir von nun an die «Nachricht (engl. message)»

Info 3: Definition der Nachricht

Eine Nachricht besteht im Grunde aus 3 Teilen: Der **Information**, der **Redundanz** und dem **Overhead**.

1.2.1 Information

Wir haben den Begriff der **Information** bereits gehört. Aber was bedeutet Information im Detail? Das ist tatsächlich gar nicht so einfach zu erklären. (Siehe auch : [W31]) **Information**, aus dem lateinischen : in-formare bedeutet «formen, bilden, unterrichten, gestalten» und beschreibt eine *Teilmenge an Wissen*. Für uns Techiker bedeutet es zum Beispiel : Der Temperaturwert eines Temperatursensors, oder die Umdrehungszahl eines Motors. Information ist der Teil einer Nachricht, welcher uns neues, interessantes, wichtiges Wissen übermittelt. Bleiben wir vorläufig beim Brief Beispiel: Im Beispiel ist der Text in dem Brief unsere Information. Die Text ist *das Gut*, welches wir von einem Ort an einen anderen Senden möchten. Für den Empfänger ist nur dieser Text von Interesse. Weder die Verpackung, das Briefpapier, noch das Versenden an sich, weder Anrede noch das «Adieu Merci...» interessieren ihn.

1.2.2 Redundanz

Wie bereits erwähnt, enthält eine Nachricht neben der Information auch die sog. «Redundanz» [W58] / [W59]. **Redundanz** beschreibt uns den Teil einer Nachricht, welcher Mehrfach vorhanden ist- Wenn wir also eine Passage in einem Text steichen können und damit keine Information vernichten, so ist dieser Teil redundant. In der Technik bezeichnet man auch eine Absicherung («Hängematten-Lösung») als Redundanz. Nehmen wir an, sie sind in einem Seilpark. Um sich an einer Route zu sichern, haben Sie zwei Karabinerhaken. In der Route sind Sie somit immer mit zwei Karabinern gesichert und falls ein Karabiner eine Fehlfunktion aufweist, sind sie immer noch über den zweiten Karabiner gesichert. Mithilfe der Redundanz in einem Text, können wir einen Text immer noch verstehen, auch wenn dieser grammatischen oder inhaltlichen Fehler enthält.

1.2.3 Entropie

Claude Shannon, ein Urgestein der modernen Informatik hat in seinem Werk «The Mathematical Theory of Communication (1949)» den Begriff der Entropie erfunden.

Die **Entropie** eines «Datenhaufens» (oder auch eines Textes) gibt uns seine minimale benötigte Größe in Bit's an.

Was bedeutet dies in der Praxis: Wenn wir eine Nachricht haben und die Entropie bestimmen, gibt uns die Entropie an, wieviel Information in der Nachricht steckt und wie viel Redundanz.

Besitzt der Datenhaufen Redundanz, können wir diesen mit einem komprimieralgorithmus komprimieren, bis dieser keine Redundanz mehr aufweist. Der resultierende Datenhaufen besteht nun nur noch aus Information.

Durch nochmaliges komprimieren lassen sich die Daten nicht weiter reduzieren.

Kein Angst, Sie werden in meinem Kurs nie die Entropie berechnen müssen.

Zitat:

Information: the negative reciprocal value of probability

Information : Der negativ reziproke Wert der Wahrscheinlichkeit

$$E = - \sum p(x) \cdot \log_2(p(x))$$

E : Entropie

p : Symbol Wahrscheinlichkeit

Claude Shannon : Mathematiker und Elektroingenieur

1.2.4 Verwaltungs-Kosten / Overhead

In einer Nachricht ist neben der Information und der Redundanz meist auch ein gewisser Anteil an **Overhead** enthalten. Dieser Overhead besteht aus sog. *Protokoll-Daten*.

Damit wir uns etwas damit vorstellen können, nehmen wir wieder das Brief-Beispiel zur Hand:

Auf unserem Brief müssen wir den Empfänger und den Absender angeben. Damit informieren wir die Post wohin Sie den Brief liefern müssen und an wen Sie den Brief zurücksenden müssen falls der Empfänger nicht erreichbar ist.

Diese Angaben sind unsere Verwaltungs-Kosten (engl. Overhead). Sie enthalten keinerlei Information und ist nicht redundant zu den Daten.

Wir brauchen diese Angaben, damit unser Kanal (also die Post) die Nachricht an den richtigen Ziel-Ort weiterleitet.

Aber auch das Brief-Couvert ist Overhead, da wir es lediglich brauchen, damit die Post damit eine standardisierte Sortiermaschine füttern kann.

Aufgabe 1.2 : Redundanz in der deutschen Sprache

Frage

Lesen Sie den folgenden Text:

Beim Lesen liesett usner Geirhn Eicsraunltehs. Für das Vehrsteen eenis Txetes kenönn die Bathesbcun der elinnzeen Wötter in bgieiebelr Roilfenehge aodgeernnt sien. Das ezinig Whigtice ist, dsas der estre und ltezte Bahbcutse kkerrot snid. Der Rest knan zufilläg gicemsht sein, und tztdeorm knan man den Text ncoh fast ohne Poelbmre leesn. Kalr ist, dsas bei zemdenuhner Lgnäe der Wöetr ein kteerorks Vrehseetn iemmr srwegiicher wird.

Was meinen Sie, wieso können Sie den Text lesen, obwohl so viele Fehler vorhanden sind?

Antwort

- Die Wörter müssen mit den selben Start- und End-Buchstaben geschrieben sein
- Die Reihefolge der anderen Buchstaben ist nicht relevant

daraus folgt:

In der deutschen Sprache haben wir viel «**Redundanz**», so dass wir Fehler erkennen und korrigieren können.

Die Äusseren Buchstaben enthalten mehr «**Information**» als die anderen.

1.2.5 Redundanz in der japanischen Sprache

Vergleichen wir dies einmal mit einer Sprache, welche extrem wenig Redundanz in seiner Symbolik aufweist.

Die Japanische Schrift besteht aus 3 Teilen. Zum einen aus Schriftzeichen sog. Kanji's, aus einer Silbenschrift dem Hiragana und einer zweiten Silbenschrift dem Katakana.

Für die Kanji's gibt es je nach Kontext sogar mehrere Aussprachen. Aber die Schriftbilder ähneln sich stark. In einem Japanischen Satz brauchen Sie Kanji, Hiragana und Katakana um den Satz zu verstehen!

Kanji	Bedeutung
大きい	gross
犬	Hund
持つ	halten, haben
待つ	warten
問	Problem
間	Zwischenraum
口	Mund
日	Tag
白	Weiss
目	Auge
且	außerdem; ferner
自	von …; ab …

Tabelle 1.1: Japanische Kanji's und ihre deutsche Wort Übersetzung

In *Tabelle 1.1* sehen Sie eine kleine Auswahl an Kanji's welche sich sehr ähnlich sehen. Anders als in der deutschen Sprache ist die Redundanz der Schrift im japanischen extrem gering!

Wir beenden nun den Ausflug in die Informationstheorie.

Zusammenfassung

Wie Sie nun gesehen haben, besteht die Kommunikationstechnik nicht nur aus Leitungen, Signalpegeln und Datenraten. Sondern auch eher abstrakte Begriffe wie Redundanz und Information. Die ganzen Themen zu Verbinden wird ebenfalls Umfang dieses Kurses sein.

Wir werden die Begriffe Information, Redundanz und Overhead zu einem späteren Zeitpunkt in *Kapitel 10 : Fehlererkennung* und *Kapitel 13 : Fehlerkorrektur* wieder aufgreifen.

Wichtig ist mir, dass Sie wissen, dass eine Nachricht immer aus **Information, Redundanz und Overhead** besteht.

2

Grundlagen Repetition

Inhalt des Kapitel

2.1 Elektrotechnik Grundlagen	10
2.1.1 Ohmsches Gesetz	10
2.1.2 Widerstand	10
2.1.3 Kapazität	11
2.1.4 Induktivität	12
diverse Materialeigenschaften	13
2.1.5 Die Natur des Dielektrikums und der Permeabilität	15
Dielektrikum	15
Permeabilität	15
2.2 Digitaltechnik Grundlagen	16
2.2.1 Digital vs. Analog	16
Das Rauschen	17
2.2.2 Logiklevel	20
2.2.3 Gatter	21
NOT - Inverter	21
Buffer	21
NAND - «Nicht-Und»	22
AND - Und	22
NOR - «Nicht-Oder»	23
OR - «Oder»	23
XOR - «Exklusiv-Oder»	24
XNOR - «Exklusiv-Nicht-Oder»	24
2.2.4 Sequenzielle Logik / FlipFlop	26
Glitch	26
D-FlipFlop	27
Register	29
Schiebe-Register	30
2.3 Zahlensysteme	30
2.3.1 Binäre- / Logische Einheiten	30
2.3.2 Hexadezimalsystem	31
Binär- / Hexadecimal-Umrechnung	31
Bits and Bytes	33
Binäre Präfixe	33
2.4 Schaltungstechnik	34
2.4.1 Treiber Schaltungen	34
Pull-Up / Pull-Down	34
2.4.2 Detektor Schaltungen	35
2.4.3 Signal Directivity	35
2.5 Mathematik	35
2.5.1 dBm - Watt umrechnen	35

Zum Inhalt

Kurze Repetition der Elektro- und Digital-Technik Grundlagen

2.1 Elektrotechnik Grundlagen

Wir wiederholen nun die Elektrotechnischen Grundlagen, welche wir in diesem Kurs benötigen.

Eigentlich wird dieses Wissen bereits vorausgesetzt!

Wir brauchen die elektrotechnischen Grundlagen um **reale Leitungssysteme** besser zu verstehen.

2.1.1 Ohmsches Gesetz

Wie sie bestimmt noch wissen, ist das Ohmsche Gesetz wie folgt definiert:

$$U = R \cdot I \quad (2.1)$$

Grösse	Formelzeichen	Einheit
Spannung	U	V
Widerstand	R	Ω
Strom	I	A

Tabelle 2.1: Formel-Zeichen / -Einheiten : Ohmsches Gesetz

2.1.2 Widerstand

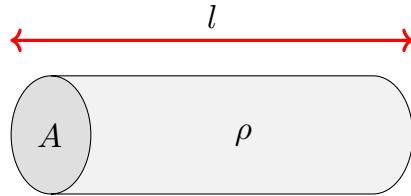


Abbildung 2.1: sehr einfaches Modell eines Ohmschen Widerstandes

Der Ohmsche Widerstand ist rein durch die physikalischen Abmessungen (Länge l , Fläche A) und des Materials (spezifische elektrische Widerstand ρ) abhängig.

$$R = \rho \cdot \frac{l}{A} \quad (2.2)$$

Grösse	Formelzeichen	Einheit
Widerstand	R	Ω
Länge	l	m
Fläche	A	m^2
spez. elektrische Widerstand	ρ	$\Omega \frac{m^2}{m}$

Tabelle 2.2: Formel-Zeichen / -Einheiten : Ohmscher Widerstand

2.1.3 Kapazität

Wir betrachten den Kondensator nur mit einem Spezial-Fall : Dem einfachen Plattenkondensator. Dieser vernachlässigt die Rand-Effekte eines Reelen-Kondensators, aber genügt für unsere Betrachtungen.

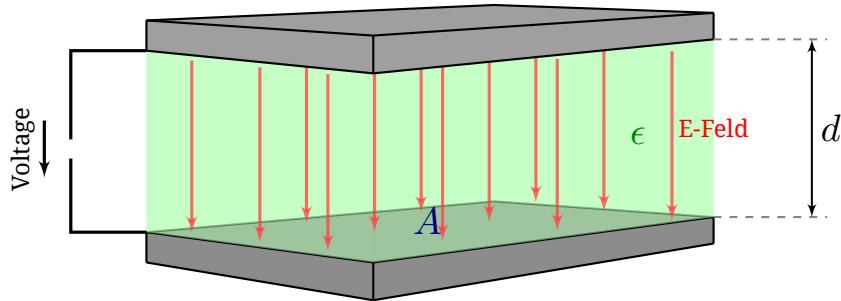


Abbildung 2.2: einfacher Plattenkondensator (Rand-Effekte vernachlässigt)

Die Kapazität ist folgendermassen definiert:

$$C \approx \epsilon \cdot \frac{A}{d} \quad (2.3)$$

Ein Plattenkondensator ist durch seine Mechanischen Abmessungen (d Abstand der Platten und A Plattenfläche) und dem dazwischen liegenden Material ϵ bestimmt. Das ϵ ist dabei eine Materialkonstante für das Dielektrikum des Kondensators und setzt sich durch eine Universalkonstante ϵ_0 und einem Material-Anteil ϵ_r zusammen.

$$\begin{aligned} \epsilon &= \epsilon_0 \cdot \epsilon_r \\ \epsilon_0 &= \frac{1}{\mu_0 \cdot c^2} = 8.854 \cdot 10^{-12} \frac{As}{Vm} \end{aligned} \quad (2.4)$$

Grösse	Formelzeichen	Einheit
Kapazität	C	F
Länge	l	m
Fläche	A	m^2
Dielektrizität / Permittivität	ϵ	$\frac{F}{m} = \frac{As}{Vm}$
Dielektrizitäts-Konstante	ϵ_0	$\frac{F}{m} = \frac{As}{Vm}$
Dielektrizitäts-Zahl	ϵ_r	1
Permeabilitäts-Konstante	μ_0	$\frac{H}{m} = \frac{Vs}{Am}$
Lichtgeschwindigkeit	c	$\frac{m}{s}$

Tabelle 2.3: Formel-Zeichen / -Einheiten : Kapazität

Info

Einfach zu merken : Luft besitzt eine Dielektrizitäts-Zahl von $\epsilon_r = 1$.

Info 4: Dielektrizitäts-Zahl bei Luft

2.1.4 Induktivität

Wir betrachten auch die Induktivität mit einem Spezial-Fall: Dem einfachen Parallelen-Leiter. Dieser vernachlässigt die «Rückseite» der Leiter, aber auch diese Betrachtung genügt für unseren Kurs

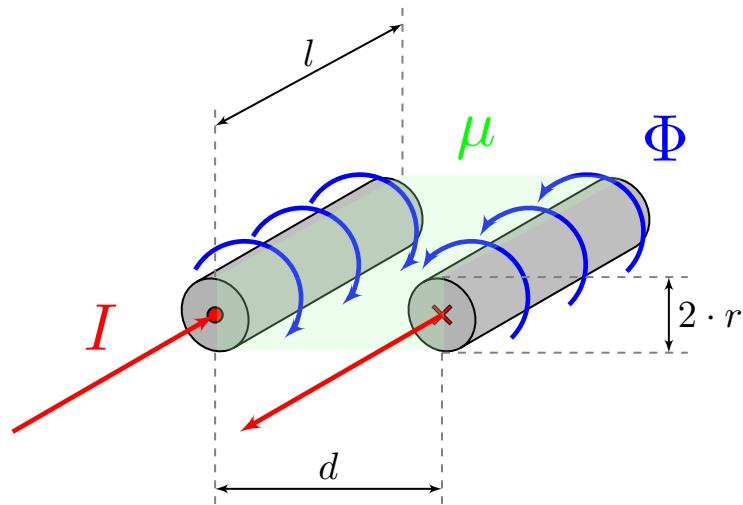


Abbildung 2.3: einfache Induktivität (parallele Leiter)

Die Induktivität eines parallelen Leitungssystems ist folgendermassen definiert:

$$L \approx \frac{\mu}{\pi} \cdot \operatorname{arccosh}\left(\frac{d}{2r}\right) \cdot l \quad (2.5)$$

Die Induktivität ist durch Ihre Mechanischen Abmessungen (d Abstand der Leiter, Dicke der Leiter $2 \cdot r$, der Leitungslänge l) und dem dazwischen liegenden Material μ definiert. Das μ ist dabei eine Materialkonstante für die Permeabilität. (Das dazwischenliegende Material) Das μ setzt sich durch eine Universalkonstante μ_0 und einem Material-Anteil μ_r zusammen.

$$\begin{aligned} \mu &= \mu_0 \cdot \mu_r \\ \mu_0 &= 4\pi \cdot 10^{-7} \frac{Vs}{Am} \end{aligned} \quad (2.6)$$

Für diese Formel gibt es eine weitere Einschränkung. Wie dargestellt in Abbildung 2.3 handelt es sich bei diesem Leiterpaar um Hin und Rück-Leiter. Aus Formel (2.5) sehen wir, dass wir mit einem grossen Abstand d der Leiter die max. Induktivität haben und mit abnehmenden Abstand d die Induktivität immer weiter abnimmt.

Dies ist auf den ersten Blick ungewöhnlich, wir werden dies aber noch in Kapitel 5.5.3 : Induktive Paar-Kopplung genauer besprechen.

An dieser Stelle sei gesagt : Die max. Induktivität entspricht der Eigen-Induktivität des Leiters und wird durch sein Counterpart mit abnehmenden Abstand d immer weiter reduziert.

Grösse	Formelzeichen	Einheit
Induktivität	L	H
Abstand	d	m
Länge	l	m
Dicke	r	m
Permeabilität	μ	$\frac{H}{m} = \frac{Vs}{Am}$
Permeabilitäts-Konstante	μ_0	$\frac{H}{m} = \frac{Vs}{Am}$
Permeabilitäts-Zahl	μ_r	1

Tabelle 2.4: Formel-Zeichen / -Einheiten : Induktivität

Info

Einfach zu merken : Luft besitzt eine Permeabilität-Zahl von $\mu_r = 1$.
Alles was magnetisch ist, besitzt eine Permeabilitäts-Zahl grösser $\mu_r > 1$

Info 5: Permeabilitäts-Zahl bei Luft / magn. Material

diverse Materialeigenschaften

Material	ρ	ϵ_r	μ_r	Preis / Tonne (9.02.2016)
Vakuum	—	1	1	—
Luft	—	1	1.0000004	—
Kupfer	$17 \cdot 10^{-3} \Omega \frac{mm^2}{m}$	—	0.999991	4507\$
Silber	$16.3 \cdot 10^{-3} \Omega \frac{mm^2}{m}$	—	0.99998	490675\$
Gold	$21.9 \cdot 10^{-3} \Omega \frac{mm^2}{m}$	—	≈ 1	38263665\$
Aluminium	$26.5 \cdot 10^{-3} \Omega \frac{mm^2}{m}$	9.8	1.00002	1484\$
Eisen	$100 \cdot 10^{-3} \Omega \frac{mm^2}{m}$	—	≈ 2000	—
Silizium	$1000 \cdot 10^{-3} \Omega \frac{mm^2}{m}$	11.68	≈ 1	—
PTFE / Teflon	—	2.1	≈ 1	—
XLPE / Polyethylene	—	2.25	≈ 1	—

Tabelle 2.5: div. Materialeigenschaften

Aufgabe 2.3 : Leitungsmaterial**Frage**

Welches Metall leitet den Strom am effizientesten (mit kleinstem Widerstand)?

Antwort

Silber, es besitzt einen spezifischen Widerstand von $\rho = 16.3 \cdot 10^{-3} \Omega \frac{mm^2}{m}$ und damit die beste Leitfähigkeit.

Aufgabe 2.4 : Widerstandsbaufom**Frage**

Sie kennen die Gleichung wie man den Widerstand aus seinen mechanischen Abmessungen berechnet.

Sie haben einen bestehenden Widerstand, wie muss man die mech. Größen verändern, damit wir den Widerstand erhöhen können?

Antwort

Wir können zum einen den Leiterquerschnitt A verkleinern, oder die Länge des Materials l vergrössern

Aufgabe 2.5 : Kapazität

Frage

Sie haben zwei Kondensatoren, beide haben die gleichen mechanischen Abmessungen. Bei Kondensator A ist als Dielektrikum Teflon verwendet worden und beim Kondensator B wurde Luft verwendet.

Welche Kondensator weist die höhere Kapazität auf?

Antwort

natürlich Kondensator A, da die Dielektrizitäts-Zahl von PTFE $\epsilon_r_{PTFE} = 2.1$ grösser als die von Luft $\epsilon_r_{Luft} = 1$

Aufgabe 2.6 : Kapazität erhöhen

Frage

Sie haben einen Kondensator und sie können den Abstand zwischen den Kondensatoren-Platten verändern.

Wie müssen Sie den Abstand verändern, damit wir die Kapazität erhöhen können?

Antwort

Je näher die Platten nebeneinander liegen, desto höher ist die Kapazität. Also d verringern.

Aufgabe 2.7 : Induktivität

Frage

Wir haben eine Induktivität mit Eisen als Permeabilität.

Ist die Induktivität grösser oder kleiner als wenn die Permeabilität mit Aluminium realisiert wird?

Antwort

Die Induktivität mit Eisen als Permeabilität ist natürlich grösser. Sehr viel grösser.

Aufgabe 2.8 : mech. Eigenschaften einer Induktivität

Frage

Was passiert wenn wir bei einer Induktivität den Abstand zwischen den Leitungen verringern?
(Wir wickeln die Induktivität dichter)

Antwort

Mit kleinerem Abstand der Leitungen nimmt die Induktivität ab.

2.1.5 Die Natur des Dielektrikums und der Permeabilität

Betrachten wir noch kurz die Natur hinter dem Dielektrikum und der Permeabilität.

Dielektrikum

Alles zwischen zwei geladenen Leitern ist ein Dielektrikum. Dabei wird das Dielektrikum polarisiert und damit die elektrischen Dipole im Material nach dem elektrischen Feld ausgerichtet. Je leichter die Dipole ausgerichtet werden können, desto besser wird das elektrische Feld durch das Dielektrikum geleitet. Wir können das ganze mit dem elektrischen Strom vergleichen und sagen : Das Dielektrikum entspricht gewissermassen der elektrischen Leitfähigkeit im Ohmschen Gesetz.

Nice to Know

Kondensatoren gibt es in vielen Formen und werden mit unterschiedlichen Dielektrikum hergestellt.
 Die Folienkondensatoren eignen sich dabei gut für Entstörfilter von Motoren oder Starkstrom-Schaltungen, da das Dielektrikum sog. selbstheilend ist.
 Für die Spannungsversorgung werden häufig Elektrolyt-Kondensatoren eingesetzt, welche hohe Kapazitäten bei kleiner Bauform ermöglichen. Aber diese haben ein Plus und Minus Anschluss, sind also Polarisiert und sollten richtig angeschlossen werden (sonst Booom...)
 Häufig werden aber auch Keramik-Kondensatoren eingesetzt. Dabei gibt es sog. X-Materialien (zB. X5R / X7R etc.) Um es kurz zu machen : verwenden Sie besser X7R anstatt X5R Material.
 Und falls Sie Filter bauen möchten, verwenden Sie unbedingt sog. C0G / NP0 Keramik Kondensatoren. (Aber nur mit kleinen Kapazitäten erhältlich)

Info 6: Dielektrikum in der Praxis

Permeabilität

Alles zwischen zwei stromdurchflossenen Leiter wird Permeabilität genannt. Dabei wird die Permeabilität magn. polarisiert und damit die magnetischen Dipole im Material nach dem magnetischen Feld ausgerichtet.

Das dürfte Ihnen nun schon sehr bekannt vor kommen.

Und ja, je leichter die Dipole ausgerichtet werden können, desto besser wird das magnetische Feld durch die Permeabilität geleitet.

Wir können das ganze mit dem elektrischen Strom vergleichen und sagen : Die Permeabilität entspricht gewissermassen der elektrischen Leitfähigkeit im Ohmschen Gesetz.

Nice to Know

Zwar ist mit Eisen als Permeabilität die Induktivität sehr viel grösser als bei einer Luftspule, aber falls Sie Filter bauen wollen, sollten Sie Luftspulen nehmen. Eine Eisenspule (oder auch Ferrit Spule genannt) weist eine Sättigung auf (es hat eine begrenzte Anzahl Dipole, sind alle umgeladen wir das $\mu_r = 1$). Ebenfalls ist das Umladen der kleinen Dipole im Eisen stark verlustbehaftet. Damit weist der Filter eine höhere Einfügedämpfung auf.

Info 7: Permeabilität in der Praxis

2.2 Digitaltechnik Grundlagen

Neben den Grundlagen der Elektrotechnik brauchen wir auch wieder die Grundlagen der Digitaltechnik. In der heutigen Kommunikationstechnik werden die Daten als «Digitale-Symbole» übertragen. Aber wieso?

2.2.1 Digital vs. Analog

Aufgabe 2.9 : Digital vs. Analog

Frage

Wieso, denken Sie, werden die Daten digital und nicht analog übertragen?

Antwort

Sie denken, weil digital schneller ist als analog? Falsch gedacht, mit der Analogtechnik erreichen Sie höhere Taktraten!

Auf einen Silizium-Die [W15] werden gerne Frequenzen im THz Bereich erreicht. Mit den heutigen CPU's sind aber nur ca. $4GHz$ (unter Extrembedingungen bis zu $10GHz$) möglich.

Sie denken, weil digital genauer ist als analog? Falsch gedacht, während Sie bei der Digitaltechnik immer mit diskreten Werten arbeiten, kann ein Analogsignal alle Werte dazwischen abdecken.

Ja wieso denn nun? Digitale Signale sind nichts anders als analoge Signale, für die nur gewisse Spannungspegel zulässig sind. Alle Zwischenwerte werden auf- resp. abgerundet.

Was nützt uns das ganze : Wir werden sehen, dass digitale Signale unter gewissen Umständen, viel störungsresistenter sind als analoge Signale. Damit lassen sich Übertragungsfehler vermeiden und damit höhere Datenraten übertragen als mit einem vergleichbaren analogen System.

TLDR: Digital-Schaltungen sind störungsresistenter

Wie kam man von der Analogtechnik zur Digitaltechnik?

Sehen wir uns *Abbildung 2.4* genauer an.

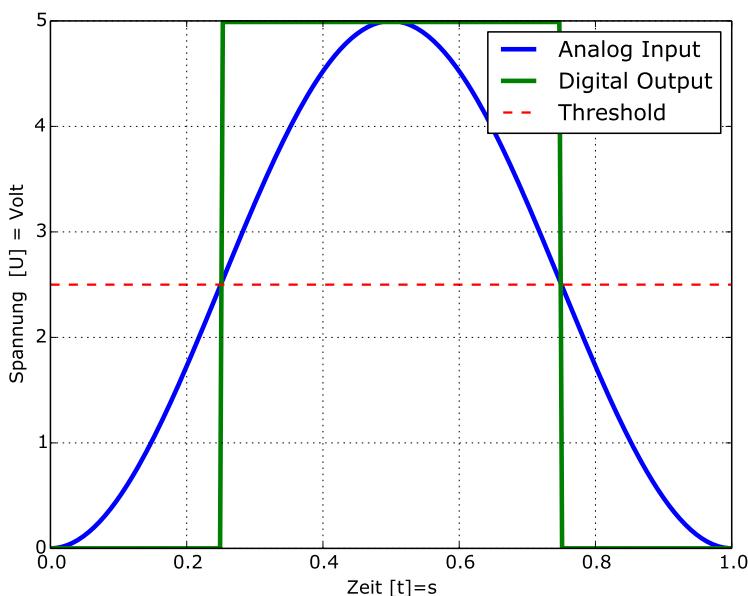


Abbildung 2.4: Analog Digital Wandlung

Wir sehen ein analoges Eingangssignal (**blau**). Das digitale Ausgangssignal (**grün**) ist zu Begin bei 0V. Sobald unser Eingangssignal über den Schwellwert (engl. threshold **rot**) gelangt, wird der digitale Ausgang auf 5V ansteigen. Sinkt das Eingangssignal wieder unter den Schwellwert, liegt am digitalen Ausgang wieder 0V an. Wir haben unser analoges «kontinuierliches» Signal zu einem «diskreten» Signal gewandelt, welches nur zwei Zustände kennt, nämlich 5V und 0V.

Das Rauschen

Das alles funktioniert sehr gut, bis wir an einem realen System arbeiten. Die analoge Welt kennt nun einen physikalischen Effekt, genannt «das Rauschen». Wie sie vielleicht wissen, bestehen wir selbst und alles um uns herum aus einzelnen Atomen. Diese «kleben» als Moleküle zusammen und formen unsere Umwelt. Das Atom selbst besteht aus einem Atom-Kern aus Protonen und Neutronen, die Orbitale bestehen aus Elektronen. In einem metallenen Leiter, sind die äussersten Elektronen eines Metall-Atom nur sehr schwach an das Atom gebunden. Diese Eigenschaft sorgt dafür, dass Metall ein guter Leiter ist. Aber es sorgt auch dafür, dass sich ein Teil der Elektronen auch zufällig von Atom löst und herumschwirren.

Aufgabe 2.10 : Verschiebung von Ladung

Frage

Sie haben nun gehört, dass das sog. Rauschen eigentlich das herumschwirren der Valenzelektronen der Metall-Atome darstellt.

Anders ausgedrückt, es werden kleine Ladungseinheiten (Elektronen) verschoben. Was Resultiert nun aus diesem Zusammenhang?
Denken Sie dabei an die Kapazität!

Antwort

Denken wir an die Kapazität : Was passiert wenn wir Ladungen trennen? Richtig, es entsteht eine Spannung!

Die Bewegung der Elektronen führt also zu kleinen zufälligen Spannungsänderungen. Wir nennen diese Spannungsänderungen «Rauschen»
<https://youtu.be/Eiy40WqrrZo?t=203>

Nice to Know

Das besprochene Rauschphänomen nennt sich **Thermisches Rauschen**. Leider gibt es eine viele verschiedene Arten von Rauschen:

1/f-Rauschen Rauschen, welches bei kleinen Frequenzen auftritt und mit steigender Frequenz abnimmt

Schrottrauschen Rauschen, welches von einem Halbleiterprozess generiert wird. zB. Dioden-PN Übergang

Stromrauschen Rauschen, welches durch sich unterschiedlich schnell bewegende Ladungsträger (Elektronen) generiert wird

Es gibt noch weitere Rauscharten. Sie alle bringen aber das gleiche Problem mit: Man bringt es nicht weg!

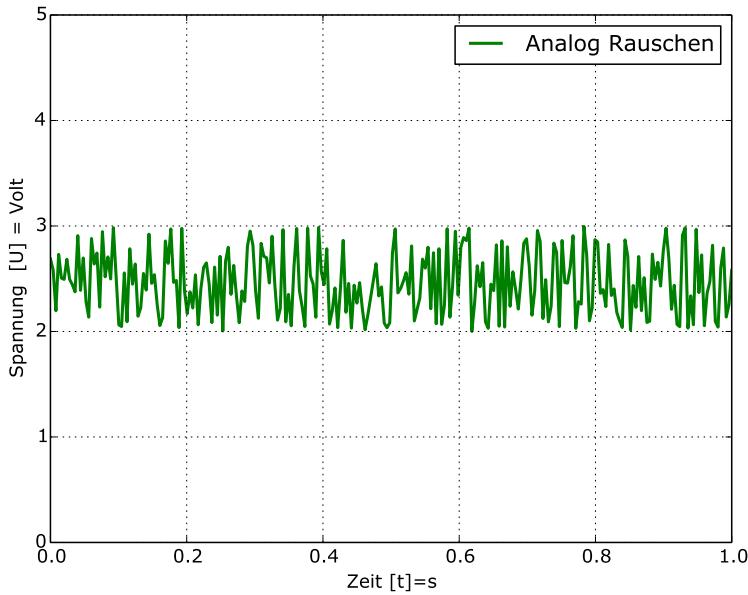


Abbildung 2.5: Analoges (Thermisches) Rauschen

In Abbildung 2.5 ist typisches (thermisches) Rauschen abgebildet.

Was hat das nun mit Analog vs. Digital zu tun?

Wie bereits erwähnt, in der realen Welt haben wir nicht so toll saubere Signale wie in Abbildung 2.4, sondern eine Überlagerung von unserem idealen Signal und dem analogen Rauschen.

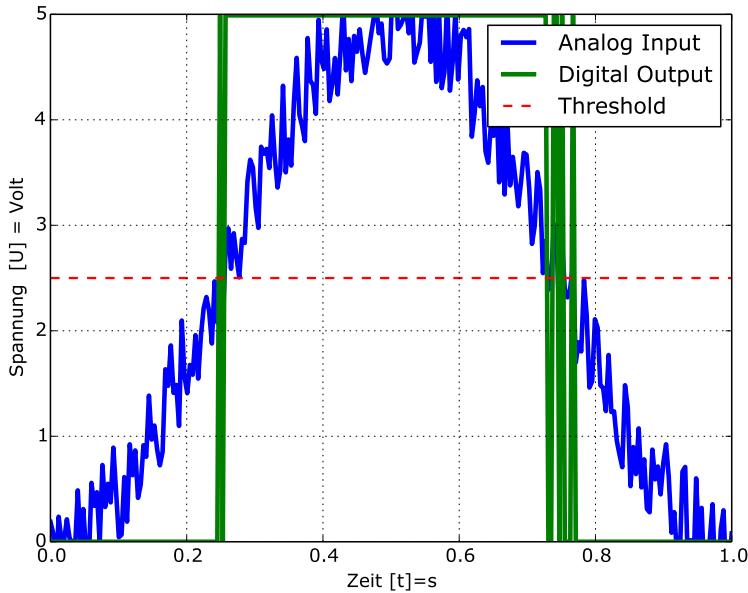


Abbildung 2.6: Mit Rauschen überlagertes Signal

Wir sehen in Abbildung 2.6 wie unser Eingangssignal überlagert mit Rauschen aussieht. Ausserdem sehen wir das digitale Ausgangssignal, welches um den Schwellwert herum sehr ungewöhnlich, beinahe eratisch (zufällig) verhält.

Wir sehen dass der Ausgang mehrmals «toggelt». Wir erkennen, dass es gefährlich ist eine digitale Schaltung mit einer Spannung, nahe an der Schwellspannung zu betreiben.

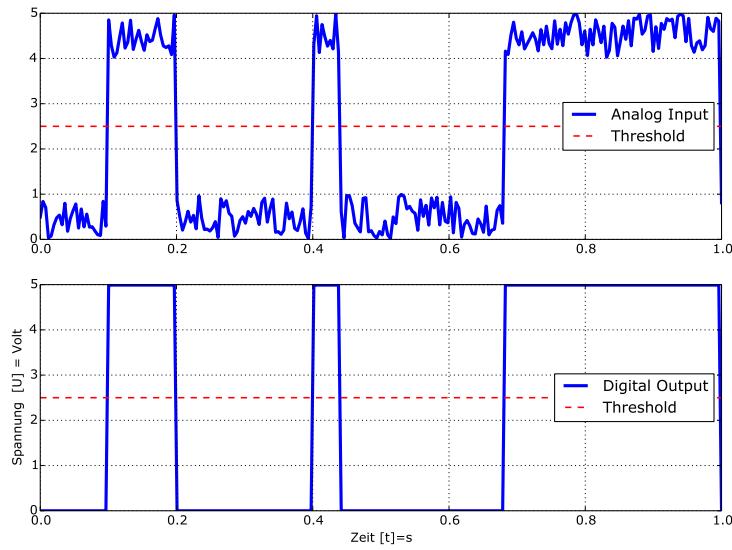


Abbildung 2.7: Restauriertes Signal

Wir sehen in Abbildung 2.7 oben ein rechteckförmiges Signal, welches mit Rauschen überlagert ist. Das könnte ein digitales Funksignal sein, welches bei der Übertragung stark verrauscht wurde.

Wir geben dieses Signal wieder auf eine digitale Schaltung und beobachten den Ausgang : Und siehe da, kein Rauschen mehr.

Das ganze ist nicht ganz korrekt: Was wir hier nicht gut sehen, ist dass sich die Flanken mit den Rauschen auch hin und herbewegen. Wir können das verrauschte Signal nicht vollkommen restaurieren, falls noch mehr Rauschen überlagert wird. Irgendwann können wir auch in der Mitte des Signals nicht mehr klar sagen, ob das empfangene Signal eine «1» oder eine «0» war.

Unsere heutigen Kommunikationssysteme müssen immer von der digitalen Welt in die analoge Welt und zurück wechseln. Hierbei können solche Effekte auftreten und wir beobachten am Empfänger fehlerhafte Daten.

Aber wir werden im Kurs lernen, wie wir trotz diesen Fehlern immer noch funktionierende Systeme betreiben können.

2.2.2 Logiklevel

In Abbildung 2.4 / 2.6 / 2.7 war ich nicht ganz ehrlich. Der Threshold ist nicht wie eingezeichnet in der Mitte der Versorgungsspannung, sondern weist eine untere und eine obere Schwelle auf.

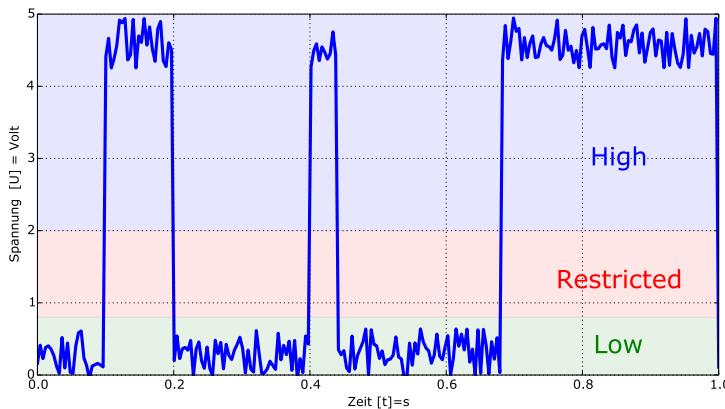


Abbildung 2.8: Digitales Signal unterteilt nach dem TTL-Standard [W40]

In Abbildung 2.8 ist nun ein Signal nach dem sog. TTL-Standard abgebildet. Ist unser Eingangssignal grösser als $V_{IH} + 2\text{ V}$, so wird der Ausgang auf seinen maximalen Spannungspiegel gezogen. Ist das Eingangssignal kleiner als $V_{IL} = +0.8\text{ V}$, so wird der Ausgang nach 0 V gezogen.

Der Bereich zwischen 2 V und 0.8 V ist ein verbotener Bereich und darf nicht am Eingang anliegen.

Es handelt sich dabei um den Bereich, bei welchem die TTL Logik leicht zu leiten beginnt. Der Eingangs-Transistor ist nicht vollständig durchgesteuert.

Der Ausgang des Logikgatters könnte in diesem Bereich «toggeln».

Es gibt nun verschiedene Standards, welche die Schwellen vorschreiben. Diese sind unter anderem von der eingesetzten Transistor-Technologie abhängig.

Standard	Eingang		Ausgang	
	V_{IL}	V_{IH}	V_{OL}	V_{OH}
TTL	$\leq 0.8\text{ V}$	$\geq 2.0\text{ V}$	$\leq 0.4\text{ V}$	$\geq 2.4\text{ V}$
CMOS	$\leq 1.5\text{ V}$	$\geq 3.5\text{ V}$	$\leq 0.5\text{ V}$	$\geq 4.44\text{ V}$
LVTTL 3.3 V	$\leq 0.8\text{ V}$	$\geq 2.0\text{ V}$	$\leq 0.4\text{ V}$	$\geq 2.4\text{ V}$
CMOS 2.5 V	$\leq 0.7\text{ V}$	$\geq 1.7\text{ V}$	$\leq 0.2\text{ V}$	$\geq 2.3\text{ V}$
CMOS 3.3 V	$\leq 0.7\text{ V}$	$\geq 1.17\text{ V}$	$\leq 0.45\text{ V}$	$\geq 1.2\text{ V}$

Tabelle 2.6: div. Logik-Standards

In der Digitaltechnik wird ein High-Pegel als logische **1** bezeichnet und ein Low-Pegel als logische **0** bezeichnet.

2.2.3 Gatter

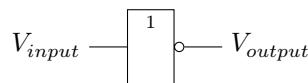
In der Digitaltechnik gibt es 8 Logische-Gatter (engl. logical gate) welche die Grundelemente der sog. Booleschen-Funktion implementieren.

Das heisst in Deutsch : Mit den Logischen-Gattern wurde die theoretische binäre Mathematik, welche George Bool anno 1847 beschrieben hatte, in der «Praxis» umgesetzt.

Nice to Know

Uns interessiert im speziellen das XOR-Gatter, da dieses häufig bei sog. Fehlerschutzcodierungen und der Kryptographie eingesetzt wird.

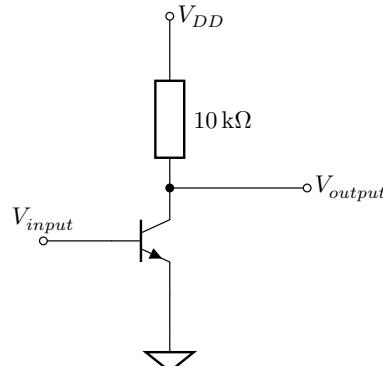
NOT - Inverter



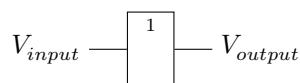
V_{input}	V_{output}
0	1
1	0

Abbildung 2.9: NOT Gate («Nicht» Gatter)

Inverter als Transistororschaltung

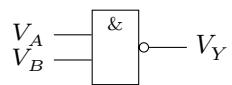


Buffer



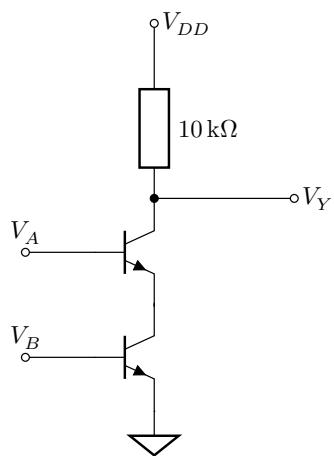
V_{input}	V_{output}
0	0
1	1

Abbildung 2.10: Buffer Gate («Buffer»)

NAND - «Nicht-Und»

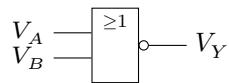
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Abbildung 2.11: NAND Gate («Nicht-Und» Gatter)

NAND als Transistorschaltung**AND - Und**

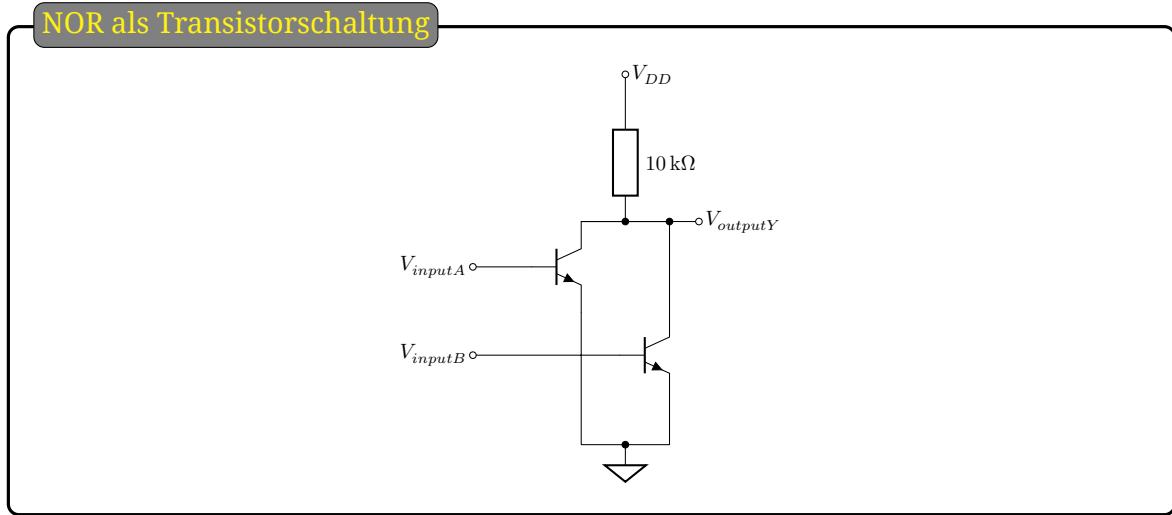
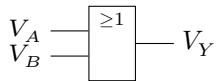
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Abbildung 2.12: AND Gate («Und» Gatter)

NOR - «Nicht-Oder»

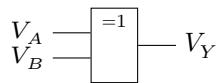
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Abbildung 2.13: NOR Gate («Nicht-Oder» Gatter)

**OR - «Oder»**

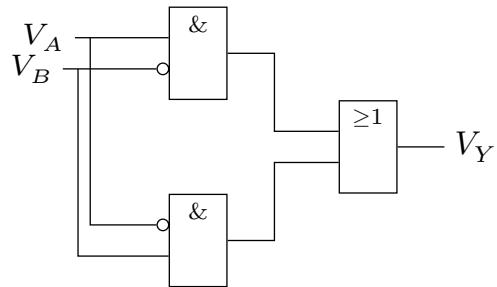
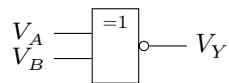
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Abbildung 2.14: OR Gate («Oder» Gatter)

XOR - «Exklusiv-Oder»

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Abbildung 2.15: XOR Gate («Exklusiv-Oder» Gatter)

XOR als Transistororschaltung**XNOR - «Exklusiv-Nicht-Oder»**

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

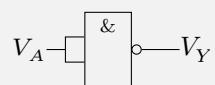
Abbildung 2.16: XNOR Gate («Exklusiv-Nicht-Oder» Gatter)

Aufgabe 2.11 : NAND-Gate umbauen

Frage

Bauen Sie aus einem NAND-Gate einen Inverter!

Antwort

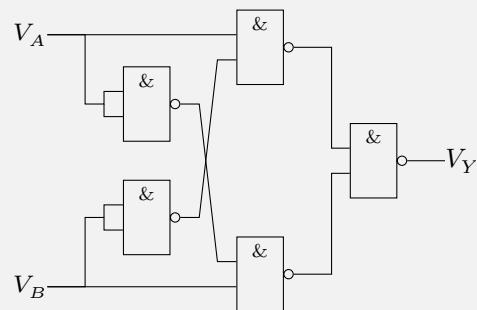


Aufgabe 2.12 : NAND-Gate umbauen (2)

Frage

Bauen Sie aus einem NAND-Gate ein EXOR!

Antwort



Nice to Know

Wenn Sie eine kleine Digitalschaltung entwerfen, können Sie nicht einfach 1 AND, 2 OR und ein EXOR Gatter kaufen. Sie können nur 8 mal das gleiche Gatter in einem Gehäuse kaufen. Was machen Sie also mit dem Rest?

Wenn Sie mit einem Gatter die anderen Gatter nachbauen können (Wofür sich das NAND Gatter gut eignet) dann müssen Sie nur mehrere Gatter des gleichen Typs kaufen, der Rest ist dann einfach.

2.2.4 Sequenzielle Logik / FlipFlop

Glitch

Nachdem immer grössere Digital-Schaltungen entworfen wurden, kam man an einen Punkt an dem die Schaltungen nicht mehr genau das getan haben, wofür man sie entworfen hatte. Man bekam «toggelnde» Ausgangssignale, zufällige zusätzliche **0/1** Übergänge.

Aber woher kam dieses Verhalten? Und wie kriegt man dieses in den Griff?

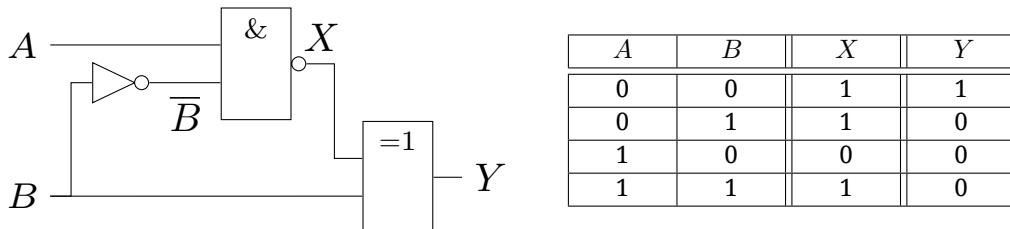


Abbildung 2.17: Schema und Wahrheitstabelle : Glitches (1)

Wir sehen in Abbildung 2.17 irgendeine kleine digitale Schaltung. Wir sehen die Implementation mit Gattern und das Wahrheitsdiagramm. Wir sehen vor uns eine Art NAND-Gatter, halt einfach sehr kompliziert aufgebaut.

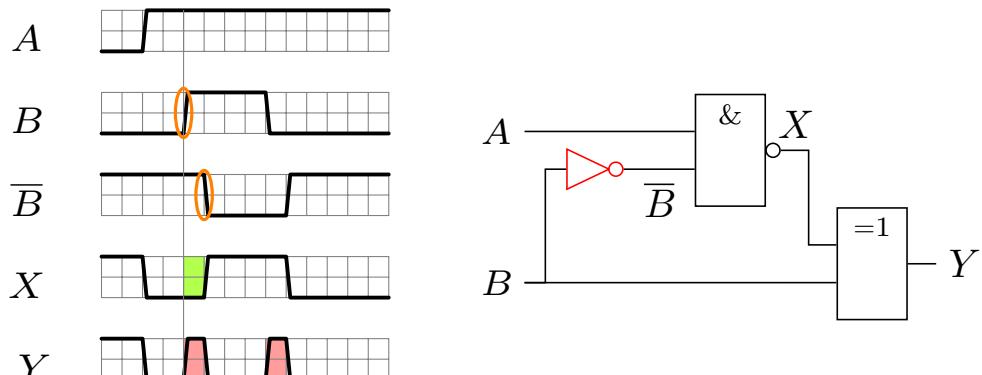


Abbildung 2.18: Timing Diagram : Glitches (2)

Wir sehen in Abbildung 2.18 Nun das «Timing»-Diagramm. Wir sehen die einzelnen Spannung / resp. digitale Signale abgebildet. (Nicht eingezeichnet ist der Zeitpfeil, links ist das älteste Signal, rechts das neuste) Wir sehen, dass die einzelnen Digitalen Schaltungen immer eine kleine Zeitverzögerung unterliegen. Wir nehmen für unser Beispiel an, dass der **rote** Inverter viel langsamer ist, als die anderen Gatter. Somit kommt es vor, dass die einzelnen Signale nicht gleich schnell durch unsere Schaltung hindurch «propagieren».

Und das führt dazu, dass wir am Ausgang der einzelnen Gatter Mischsignale aus schnellen und langsamem Signalen erzeugt werden.

Sehen wir uns das noch gleich an diesem Beispiel genauer an: Wir sehen das Signal B und das inverse Signal \bar{B} . Dazwischen eine kleine Zeitverzögerung.

Das führt dazu, dass der Ausgang X auch ein wenig verzögert die richtige NAND Operation ausgiebt. (Siehe **grüne** Fläche)

Schlimm wird es erst, wenn wir das verzögerte Signal X mit dem nicht verzögerten Signal B im zweiten EXOR Gatter verrechnen.

Wir sehen nun bei Signal Y ein kurzen Bereich (rote Fläche), bei dem das Signal nicht mit der Wahrheitstabelle übereinstimmt!

Dieses aus den verzögerten «falsche» Signal nennt man einen Glitch.(Siehe **rote** Fläche)

Es stellt sich nun die Frage, wie wir dieses Verhalten unterdrücken können.

Nice to Know

Wir haben nun gesehen wie «Glitches» entstehen.
 Merken Sie sich eines : Glitches gehören zu den übelsten Fehlern / Störungen welche wir in einem Digital-System haben können. Sie sind schwer zu debuggen und sorgen dafür das unsere Schaltung nicht so funktioniert wie erwartet.

D-FlipFlop

Wie können wir nun die Digitalschaltung aus Abbildung 2.18 verbessern?

Hierzu müssen wir zunächst ein weiteres Bauteil der Digitaltechnik kennen lernen.

Es nennt sich D-FlipFlop und gehört zur Klasse der Sequenziellen-Logik.

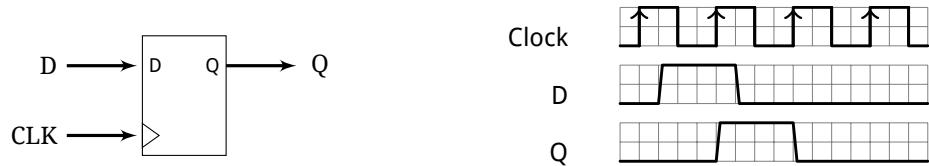


Abbildung 2.19: D-FlipFlop

Stellen Sie sich das D-FlipFlop wie ein Buffer vor, welchen wir über das sog. Takt-Signal (engl. Clock-Signal) steuern.

Mit einer positiven Flanke wird das Eingangssignal D übernommen und am Ausgang Q ausgegeben. Alle Signal-Änderungen welche zwischen den pos. Takt-Flanken passieren, werden ignoriert.

Aufgabe 2.13 : De-Glitch

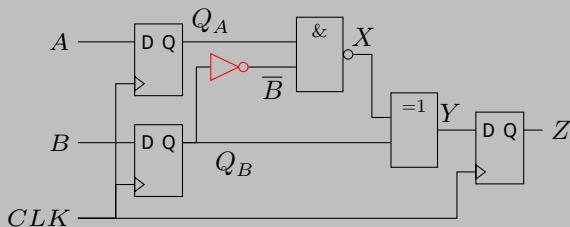
Frage

Wir verwenden die Schaltung aus Abbildung 2.18 und bauen für alle Eingänge und Ausgänge jeweils ein D-FlipFlop ein.

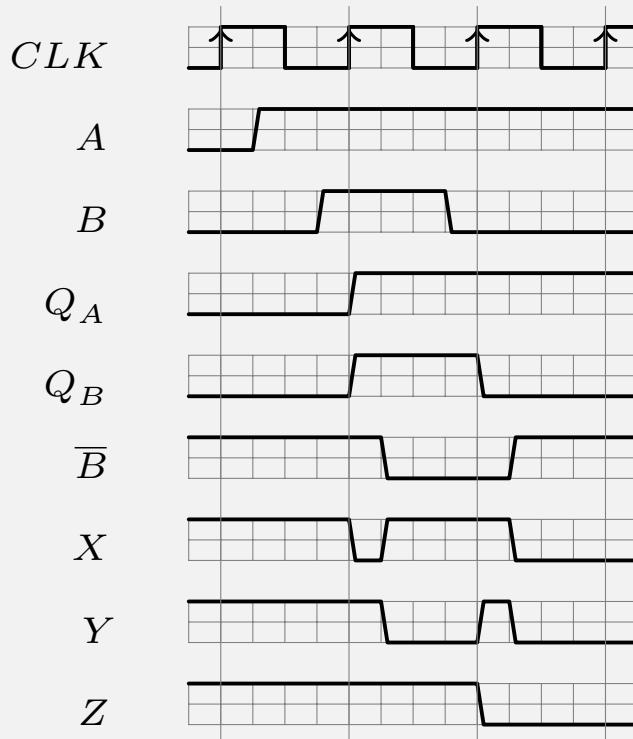
Bestimmen Sie das Timingdiagramm

Nehmen Sie an, dass das D-FlipFlop ideal ist : der Ausgang ändert sofort mit der pos. Flanke

Als Hilfe : Zeichnen Sie die Ausgangssignale der FlipFlop ebenfalls ein : Q_A / Q_B



Antwort



Wie wir in Aufgabe 2.13 gesehen haben, können wir D-FlipFlops einsetzen um komplizierte Digitalschaltungen zu «de-glitchen». Wie immer erkaufen wir uns diese Eigenschaft mit einem Nachteil. Wir müssen sicherstellen das die Periodendauer des Taktsignals (CLK) kleiner ist, als die längste Signal-Durchlaufzeit (engl. «propagation-delay») der Logikschaltungen zwischen den FlipFlops.

Was ich noch unterschlagen habe, sind die sogenannte «setup» und «hold» Zeiten, diese sollten Sie aber bereits kennen.

Nice to Know

Sie kennen nun einen weiteren Grund wieso die CPU-Taktrate heutiger PC's seit Jahren bei ca. $3 - 4\text{GHz}$ verharrt.

Der propagation-delay der Elektronen auf einem Silizium Wafer ist so gross, dass man mit einem Signal nur wenige Logikgatter hintereinander schalten kann, bis wieder ein D-FlipFlop zum «deglitching» gebraucht wird.

Angenommen wir betreiben eine CPU mit 4 GHz, so dürfen die Signale zwischen zwei FlipFlops gerade einmal einen propagation-delay von $t_{pd} = \frac{1}{4\text{GHz}} = 0.25\text{ ns}$ aufweisen.

Register

Wenn wir gleich beim D-FlipFlop sind : Wenn wir mehrere D-FlipFlop zu parallel zusammen schalten, sprechen wir von einem Register.

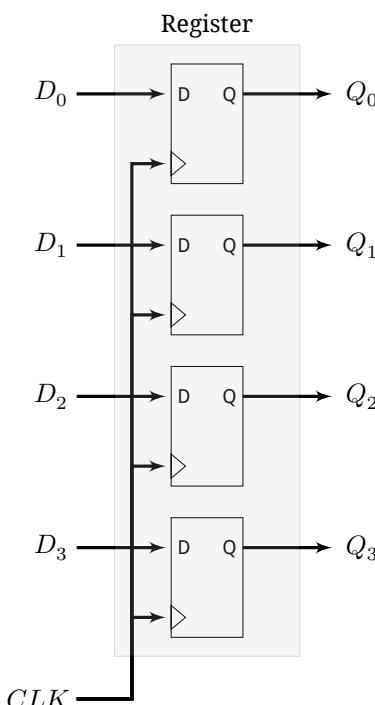


Abbildung 2.20: D-FlipFlop Register

Ein Register finden wir zum Beispiel in unserm PC / Laptop. Die CPU verarbeitet parallel immer mehrere digitale Signale zusammen. Wenn die CPU nun Daten zwischenspeichern möchte, schiebt es diese in ein Register.

Nice to Know

Sie haben sicher schon einmal von einem RAM Baustein gehört. Dieser besteht im Grunde aus vielen, sehr vielen D-FlipFlops, welche in Registern à 8 / 16 / 32 / 64 Bit angesprochen werden können.

Vielleicht haben Sie ebenfalls gehört das ihr Laptop eine «32 oder 64 Bit» Machine ist. Die 32 / 64 Bit stehen für die Registergrösse, welche die CPU direkt verabreiten kann.

Schiebe-Register

Wenn wir mehrere D-FlipFlop seriell zusammen schalten, sprechen wir von einem Schiebe-Register.

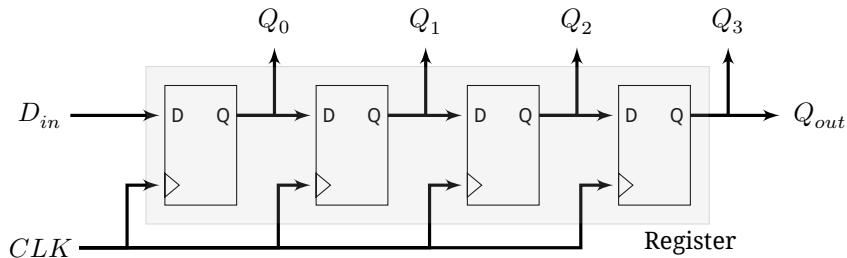


Abbildung 2.21: D-FlipFlop Schiebe-Register

Das Schiebe-Register in Abbildung 2.21 ist ein Serial-in, Parallel-out Schiebe-Register und wird beispielsweise bei der RS-232 (Kapitel 3 : Die serielle Schnittstelle RS-232 / UART) Schnittstelle als Empfangs-Register benutzt. Dazu aber später mehr.

2.3 Zahlensysteme

In der Informationstheorie rechnet man in der Regel im Binärsystem. Für uns Menschen ist allerdings das Dezimalsystem gängiger. Wir gehen nun kurz auf das Wesen der wichtigsten Zahlensysteme ein. Sehen wir uns zunächst das Dezimalsystem an: Im Dezimalsystem können pro Ziffer alle Zahlen im Werte $\mathbb{R}_{10} \in [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$ auftauchen. Ist eine Zahl grösser, als die einzelnen Ziffern, werden zwei Ziffern hintereinander geschrieben und bilden zusammen die Zahl im Dezimalsystem ab.

$$237_{10} = 2 \cdot 100 + 3 \cdot 10 + 7 \cdot 1 = 2 \cdot 10^2 + 3 \cdot 10^1 + 7 \cdot 10^0 \quad (2.7)$$

Wie wir in (2.7) sehen, wird jede Ziffer mit dem Faktor 10^n multipliziert, abhängig davon an welcher Stelle sich die Ziffer befindet.

Benutzen wir die gleiche Theorie nun für das Binäre-Zahlensystem:

$$\begin{aligned} 237_{10} &= 1110\ 1101_2 = 1 \cdot 128 + 1 \cdot 64 + 1 \cdot 32 + 0 \cdot 16 + 1 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 \\ &= 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 \end{aligned} \quad (2.8)$$

Wir sehen in (2.8) wieder das gleiche Schema. Jede Ziffer wird mit einem Faktor 2^n multipliziert, abhängig davon an welcher Stelle sich die Ziffer befindet.

Beide Zahl sind gleichwertig, sie sind lediglich anders dargestellt.

2.3.1 Binäre- / Logische Einheiten

Diese Einheiten bilden das Binäre-Zahlensystem. Das heisst für jede Ziffer gibt es zwei Zustände, danach wird eine weitere Ziffer benötigt.

Möchte man nun ausrechnen wieviele Stellen die dezimale Zahl 237_{10} als Binärzahl benötigt, kann man das ganze folgendermassen ausrechnen :

$$\begin{aligned} n &= \text{ceil}(\log_2(237_{10})) \\ &= \text{ceil}(7.8) \\ &= 8 \end{aligned} \quad \text{ceil}(x) : x \text{ aufrunden} \quad (2.9)$$

2.3.2 Hexadezimalsystem

In der Informationstheorie wird viel im Binärsystem gerechnen. Allerdings wachsen die Anzahl Stellen bei einer Binärzahl mit zunehmendem Wert schnell an.

Es hat sich deshalb etabliert, dass man die Zahlen meist als Hexadezimal-Zahl schreibt.

Das Hexadecimalsystem besteht aus dem Wertebereich $\mathbb{R}_{16} \in [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F]$

$$\begin{aligned} 237_{10} &= ED_{16} = E \cdot 16 + D \cdot 1 \\ &= E \cdot 16^1 + D \cdot 16^0 \\ &= 14 \cdot 16^1 + 13 \cdot 16^0 \end{aligned} \quad (2.10)$$

Nice to Know

In den meisten Programmiersprachen wird eine Binärzahl mit einem 0b Prefix geschrieben: 0b1110 1101

Eine Hexadecimalzahl hat meist den Prefix 0x : 0xED

Die Gross- / Kleinschreibung wird bei Hexzahlen nicht unterschieden.

Binär- / Hexadecimal-Umrechnung

Will man eine Hexzahl in eine Binärzahl umrechnen, so lässt sich dies recht einfach realisieren.

Für jede Hexziffer brauchen wir 4 Binärziffern.

Dies daher, da jede Hexziffer max die Zahl 16 annehmen kann. 4 Binärziffern können ebenfalls genau 16 Zahlen darstellen.

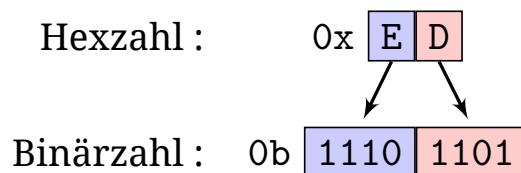


Abbildung 2.22: Binär- / Hexadecimal-Umrechnung

Für das Umrechnen selbst ist ein klein wenig Kopfrechnen angesagt, aber die Umrechnung lässt sich recht schnell erlernen.

Wir lernen die folgende Umrechnungstabelle auswendig:

Dezimal	Hexadezimal	Dezimal	Hexadezimal
0	0x0	8	0x8
1	0x1	9	0x9
2	0x2	10	0xA
3	0x3	11	0xB
4	0x4	12	0xC
5	0x5	13	0xD
6	0x6	14	0xE
7	0x7	15	0xF

Tabelle 2.7: Hex-Dezimal Umrechnung

Wir können die Dezimalzahl jeweils durch zwei teilen und prüfen ob die Zahl gerade oder ungerade ist. Ist die Zahl ungerade ist das kleinste Bit = 1, danach teilen wir die Zahl (jeweils abrunden) und testen erneut:

$$\begin{aligned}
 0xE &= 14 : 2 = 7_{\text{Rest } 0} \Rightarrow 0b\underline{\hspace{2pt}}0 \\
 7 : 2 &= 3_{\text{Rest } 1} \Rightarrow 0b\underline{\hspace{2pt}}10 \\
 3 : 2 &= 1_{\text{Rest } 1} \Rightarrow 0b\underline{\hspace{2pt}}110 \\
 1 : 2 &= 0_{\text{Rest } 1} \Rightarrow \underline{0b}1110
 \end{aligned} \tag{2.11}$$

Aufgabe 2.14 : Hex, Binäre, Dezimal Umrechnung

Frage

Berechnen Sie die fehlenden Werte in der Tabelle:

Antwort

Dezimal	Hexadezimal	Binär
15	0xF	0b1111
130	0x82	0b1000 0010
1025	0x401	0b100 0000 0001
170	0xAA	0b1010 1010
125	0x55	0b0101 0101
240	0xF0	0b1111 0000
325	0x145	0b1 0100 0101
71	0x47	0b100 1111

Bits and Bytes

In der Kommunikationstechnik werden viele Vorgänge in Fixed Point Berechnungen durchgeführt. Man sollte daher die Bezeichnungen für die Registergrößen in einem Prozessor für Fixed Point Integer kennen.

Wert	Bedeutung
1 bit	kleinste binäre Einheit
4 bit	nibble
8 bit	byte
2 byte / 4 byte	1/2 word / short / word / int*
2 word	1 long*
2 long	1 long long*
bit [0 : 3] von einem byte	lower nibble
bit [4 : 7] von einem byte	higher nibble
bit ₀	least significant bit (LSB)
bit _{max}	most significant bit (MSB)

Tabelle 2.8: Bits and Bytes

Leider wurden bei den Anfängen des PC Zeitalters keine klaren Regeln für Integer Bezeichnung aufgestellt. Bis zu Byte ist alles klar, danach folgen die Integer Größen. Ein Short ist sicher ein halbes Word, ein Word kann allerdings der Prozessor Architektur Verarbeitungsgröße entsprechen. * Daher entweder [8, 16, 32, 64] oder noch etwas dazwischen. Wir verwenden im Text für ein Word die Größe 32 bit. (Also 16 bit für Short)

Binäre Präfixe

Die Präfixe für Binäre Zahlen sind leider nicht eindeutig. Es haben sich zwei Standards etabliert, den JEDEC Standard und der IEC Standard. Es sollte nur noch die Bezeichnung nach IEC verwendet werden und ich versuche im Skript auch alle Zahlen nach IEC zu formulieren.

Ich bin allerdings noch mit der JEDEC Bezeichnung aufgewachsen und es kann nicht ausschließen, dass sich mal ein Fehler einschleicht.

Wieso gibt es überhaupt zwei Standards? Lange war nichts definiert, und etwa zur gleichen Zeit, haben zwei unterschiedliche Gremien eine Standard hervorgebracht.

Aber als kleine Vorwarnung: Wir werden uns noch viel über solche Doppelspurigkeiten aufregen.

Dezimal			Binär		
Wert	IEC		Wert	IEC	JEDEC
1000	10 ³	kB kilobyte	1024	2 ¹⁰	KiB kibibyte
1000 ²	10 ⁶	MB megabyte	1024 ²	2 ²⁰	MiB mebibyte
1000 ³	10 ⁹	GB gigabyte	1024 ³	2 ³⁰	GiB gibibyte
1000 ⁴	10 ¹²	TB terabyte	1024 ⁴	2 ⁴⁰	TiB tebibyte
1000 ⁵	10 ¹⁵	PB petabyte	1024 ⁵	2 ⁵⁰	PiB pebibyte
1000 ⁶	10 ¹⁸	EB exabyte	1024 ⁶	2 ⁶⁰	EiB ebibyte
1000 ⁷	10 ²¹	ZB zettabyte	1024 ⁷	2 ⁷⁰	ZiB zebibyte
1000 ⁸	10 ²⁴	YB yottabyte	1024 ⁸	2 ⁸⁰	YiB yobibyte

Tabelle 2.9: Präfixe bei Binären Zahlen

Wir sehen in der Tabelle 2.9, dass die Definition nach IEC das Kilobyte als 1000 Byte definiert und das JEDEC das Kilobyte als 1024 Byte definiert.

⇒ Wir verwenden die IEC Norm: 1000 Byte = 1 kB

2.4 Schaltungstechnik

Wir machen nun noch einen kurzen Abstecher in die Schaltungstechnik.

Wir gehen hier nicht all zu sehr ins Detail, da dies eher in eine Elektronik / Elektrotechnik Vorlesung gehört.

2.4.1 Treiber Schaltungen

Die elektrische Anbindung unserer digitalen Signale wird über sogenannte Treiber-Schaltungen realisiert.

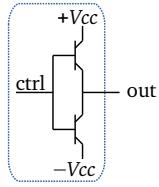
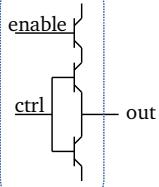
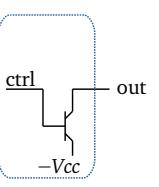
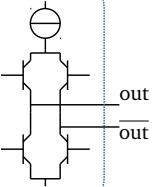
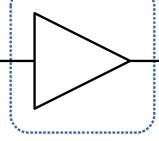
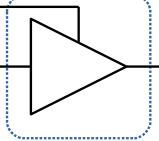
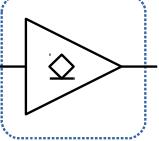
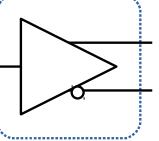
Typ	Totem Pole	Totem-Pole (Tristate)	Open-Collector Open-Drain	Low-Voltage Differential Signaling LVDS
Schaltung				
Symbol				
Ausgang verketten?	Niemals Kurzschluss Gefahr	Möglich, falls nur ein Treiber aktiv	immer	Niemals, Kurzschluss Gefahr

Tabelle 2.10: Treiber Schaltungen

Pull-Up / Pull-Down

Zur Illustration hier noch ein Pull-Up / Pull-Down dargestellt. Diese werden bei Open-Collector / Open-Emitter Schaltungen benötigt

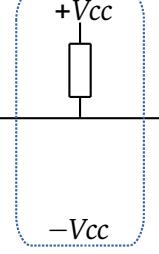
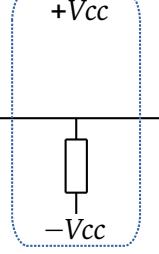
Typ	Pull-Up	Pull-Down
Schaltung		

Tabelle 2.11: Pull-Up / Pull-Down Schaltungen

2.4.2 Detektor Schaltungen

Bei den Detektor Schaltriggern haben sie eine Hysterese Funktion. In der Praxis setzen sie Schmitt-Trigger für Takt-Eingänge ein. Schlussendlich haben Sie zwei Typen von Eingangsschaltungen. Die einfacheren haben eine Schwelle bei der sie das zwischen 0/1 unterscheiden und bei den Schmitt-Trigger hilft die Hysterese sog. Glitches zu vermeiden. Sind allerdings auch teurer.

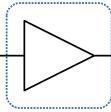
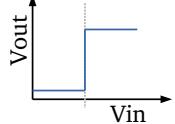
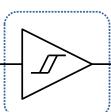
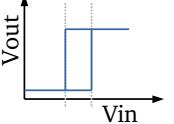
Normaler Eingang	Schmitt-Trigger
 	 

Tabelle 2.12: Detektor Schaltungen

2.4.3 Signal Directivity

Die Signalanschlüsse welche an das Übertragungsmedium angeschlossen werden, enthalten entweder eine **Treiber**-, eine **Detektor-Schaltung** oder **beide**. Einen Signalanschluss an einem IC nennt man in der Regel ein Port. Ist nur ein Treiber verbaut spricht man von einem **Ausgang (output)**. Bei einem Detektor von einem **Eingang (input)** und sind beide verbaut bezeichnet man dies als **bidirektional (bidirectional)**.

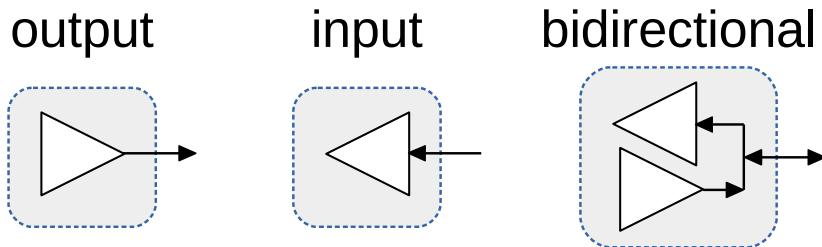


Abbildung 2.23: Direktivität

2.5 Mathematik

2.5.1 dBm - Watt umrechnen

Vor allem in der Funk-Technik insbesondere in Standards- und Zertifizierungs-Dokumenten werden Leistungen in dBm angegeben.

dBm ist ein logarithmisches Mass, welches auf 1 mW bezogen wird.

$$\begin{aligned} P_{\text{dBm}} &= 10 \cdot \log_{10} \left(\frac{P_W}{1 \text{ mW}} \right) & [P_W] = \text{W}[P_{\text{dBm}}] &= \text{dBm} \quad (2.12) \\ P_{\text{dBm}} &= 10 \cdot \log_{10} (P_W) + 30 \text{ dB} \end{aligned}$$

3

Die serielle Schnittstelle RS-232 / UART

Inhalt des Kapitel

3.1	Grundlagen	38
3.1.1	Kommunikationsmodell	38
3.1.2	parallele Datenübertragung	38
3.1.3	serielle Datenübertragung	39
3.1.4	Vergleich : serielle / parallele Datenübertragung	40
	Datenleitungen	40
	Komplexität	40
	Taktrate	40
	Bit	41
	Bitrate	41
	Symbol	41
	Symbolwert	41
	Symbolrate / Baudrate	42
	Symboldauer	42
	Bitwichtigkeit	42
3.1.5	Kurzer Exkurs : Modulation	45
3.2	LPT : Parallele Schnittstelle	46
3.3	RS-232 : Serielle Schnittstelle	47
3.3.1	Leitungscode	48
3.3.2	Einsatzgebiet	49
	Verbindung	49
3.4	UART : Universal Asynchronous Receiver Transmitter	51
3.4.1	Leitungscode	51
3.4.2	Optionen	52
	Anzahl Datenbits	52
	Baudraten	52
	Parity Bit	52
	Anzahl Stop-Bits	52
	LSB / MSB first	53
	Endianess	53
3.5	Fazit	54

Zum Inhalt

In diesem Kapitel lernen wir die zwei Grundlegenden Übertragungsarten: die serielle resp. die parallele Datenübertragung kennen.

Wir werden ausserdem zwei Vertreter dieser Übertragungsarten, die «serielle Schnittstelle» RS-232

(COM) [W62] und kurz die «parallele Schnittstelle» Centronics / IEEE1284 (LPT) [W50] besprechen.

Damit wir für das Praktika gewappnet sind, werden wir ausserdem die sog. UART Schnittstelle [W70] be sprechen. Welche sehr ähnlich zu RS-232 ist.

3.1 Grundlagen

Die «parallele Schnittstelle» [W50] und die «serielle Schnittstelle» [W62] am PC sind zwei der ältesten Kommunikations-Schnittstellen der PC Geschichte.

Seit der grossen Verbreitung von USB [W71] (Universal Serial Bus) sind beide fast vollständig verschwunden.

Immerhin die «serielle Schnittstelle» RS-232 (genauer UART) wird heute über USB-Adapter relativ einfach zur Verfügung gestellt. Sie werden im Praktikum mit solch einem «USB-Serial Adapter» arbeiten.

Wieso beginnen wir mit diesem beiden Schnittstellen?

Die serielle und die parallele Schnittstelle sind sehr einfach aufgebaut. Damit kann ich Ihnen die Grundlagen der elektronischen Kommunikation gut vermitteln.

3.1.1 Kommunikationsmodell

Erinnern wir uns noch einmal an das «Beispiel eines Sernder- / Empfangs-Systems» Abbildung 1.3.

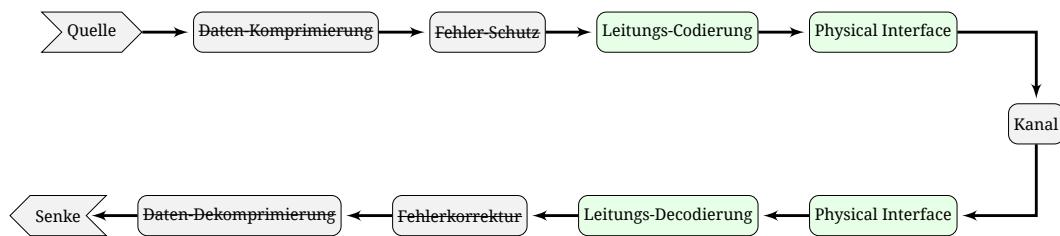


Abbildung 3.1: Einfaches Sender- / Empfänger-Systems (LPT / COM)

In Abbildung 3.1 ist das Beispiel aus Abbildung 1.3 angepasst worden. Wir haben weder eine Daten-Kompression noch Fehlerschutz in diesem Standard definiert.

Wir haben einen sog. Leitungscode definiert und das Physikalische Interface (Stecker, Pinbelegung) definiert.

Die serielle und die parallele Schnittstelle unterscheiden sich, wie der Name schon recht eindeutig sagt, wie ein «Datenwort» übertragen wird.

3.1.2 parallele Datenübertragung

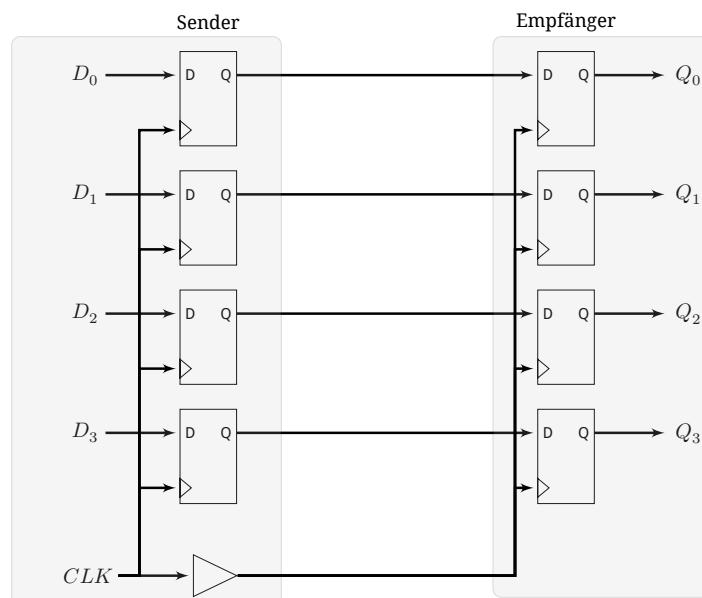


Abbildung 3.2: parallele Datenübertragung

Betrachten wir zunächst die parallele Datenübertragung. Wie wir in Abbildung 3.2 sehen, besteht unser Kommunikationssystem aus einem Sender und einem Empfänger. Der Sender möchte ein Datenwort $[D_0, \dots, D_3]$ an den Empfänger $[Q_0, \dots, Q_3]$ schicken. Die Sende-, sowie Empfangs-Logik besteht aus einem D-FlipFlop Register. Alle Signalleitungen werden mit einem Kabel vom Sender zum Empfänger inkl. Taktsignal CLK und natürlich eine Masseleitung GND (nicht eingezeichnet) verbunden. Mit jeder Taktflanke des CLK wird nun ein Datenwort an den Empfänger gesendet.

Info

Vergessen Sie nie die Masseleitung!
Meist ist die Masseleitung nicht in einer Übersicht gezeichnet.

Info 8: Masseleitung in der Praxis

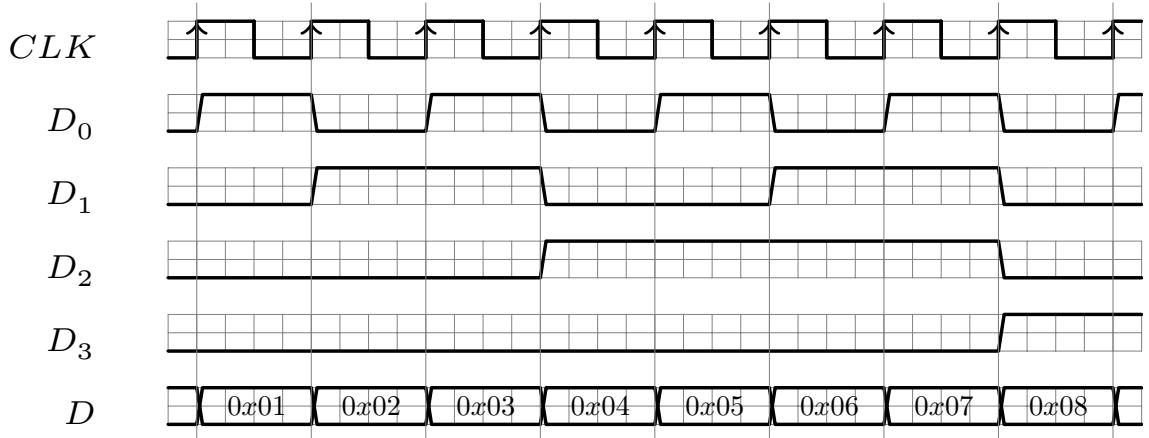


Abbildung 3.3: parallele Datenübertragung : Beispiel Timing Diagram

In Abbildung 3.3 ist das Timing-Diagramm einer parallelen Datenübertragung abgebildet. Wir sehen die einzelnen Datensignale $[D_0, \dots, D_3]$ und das Taktsignal CLK .

In der letzten Zeile finden sie die Datenleitungen zusammengefasst in einer Zeile. Damit spare ich mir ein wenig Zeit beim Zeichnen und zum anderen möchte ich mit dieser Notation den Begriff des sog. «Symbols», der «Symbolrate / Baudrate» erläutern.

3.1.3 serielle Datenübertragung

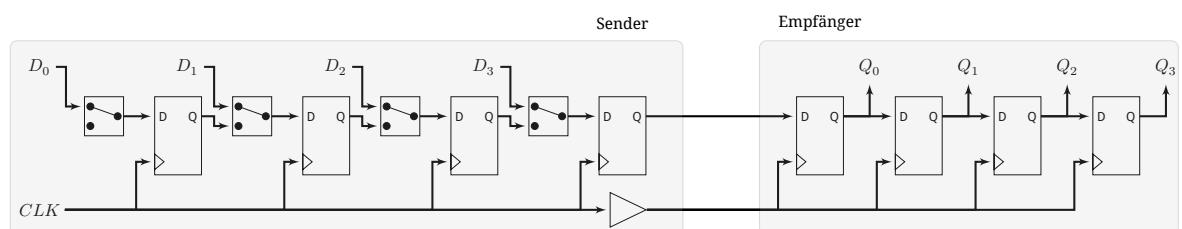


Abbildung 3.4: serielle Datenübertragung

Betrachten wir nun die serielle Datenübertragung.

In Abbildung 3.4 ist ein Beispiel einer seriellen Datenübertragung abgebildet. Die serielle Datenverbindung ist ein wenig komplizierter aufgebaut. Wir nehmen unser Datenwort $[D_0, \dots, D_3]$ und müssen dieses zunächst in ein Schiebegregister laden. (auch als «latching» Bezeichnet)

Dies wurde mit den Schaltern angedeutet. Zur besseren Übersicht wurde die Steuerleitung der Schalter weg gelassen.

Wie bereits gesagt, das Datenwort wird zunächst an die Schalter angelegt und mit einem Taktflanke in das Schieberegister geladen. Danach werden die Schalter umgelegt, so dass die Daten mit $[D_3]$ zuerst auf das Kabel ausgegeben wird.

Beim Empfänger befindet sich ebenfalls ein Schieberegister. Mit jeder Taktflanke von CLK wird ein Datenbit in das Schieberegister geladen. Nach 4 Schiebe-Taktzyklen von CLK befindet sich unser Datenwort im Empfänger und wird als Empfangswort $[Q_0, \dots, Q_3]$ ausgegeben.

Das ganze ist schon sehr einfach dargestellt.

Übrigens, die Taktflanke, welche zum latching verwendet wird, wird nicht an den Ausgang des CLK -Buffers ausgegeben.

3.1.4 Vergleich : serielle / parallele Datenübertragung

Wir haben nun zwei grundsätzliche Datenübertragungs-Arten betrachtet.

Wenn sie ein Mikrokontroller oder ein Datenübermittlungsmodul auswählen, haben Sie meist die Auswahl zwischen einem System, welche die Daten parallel oder seriell Übermittelt.

Was sind die Fundamentalen Unterschiede?

Datenleitungen

Die parallele Datenverbindung braucht mehr Datenleitungen als eine serielle Datenverbindung.

Komplexität

Die serielle Datenverbindung ist komplexer, da wir mehrere Schritte vor der eigentlichen Datenübertragung ausführen müssen.

Taktrate

Für die Übertragung der gleichen Datenmenge wie bei einer parallel Datenübertragung mit n -Daten-Leitungen, muss bei der seriellen Verbindung die Taktrate um Faktoren $f_{Clk\ ser} = n \cdot f_{Clk\ par}$ höher sein.

Aufgabe 3.15 : Zwischenfrage

Frage

Welche Art der Datenübertragung wird heute häufiger eingesetzt
: Parallel oder Serielle Datenübertragung?
Nennen Sie Beispiele.

Antwort

Die serielle Datenübertragung: zB. SATA, PCIe, WLAN, Ethernet
Aber ganz richtig ist das auch nicht. Bei PCIe können Sie mehrere «Lanes» parallel schalten um den Datendurchsatz zu erhöhen.
Bei WLAN verwenden Sie komplizierte Modulationen, bei welcher Sie mehrere Bits pro Sendepuls übermitteln, was man auch als parallele Datenübertragung bezeichnen könnte.
Bei Ethernet haben Sie mehrere Adresspaare, welche wie bei PCIe zusammengeschaltet sind.

Bit

Unsere Grundeinheit ist das Bit.

Es kann 2 Werte annehmen : **0** oder **1**.

Die Einheit des Bit ist folgendermassen definiert: $[bit] = 1$

Bitrate

Übermitteln wir eine gewisse Menge an Daten $\Delta bits$ über eine definierte Zeit Δt so sprechen wir von der Bitrate:

$$R_b = \frac{\Delta bits}{\Delta t} \quad [R_b] = bit/s \quad (3.1)$$

Symbol

Ein Symbol ist das Datenwort, welches wir aus einzelnen Bits zusammen setzen und gleichzeitig zum «Referenzclock» übertragen. Ein Symbol besteht aus der Anzahl m -bits.

$$[Symbol] = bit \quad (3.2)$$

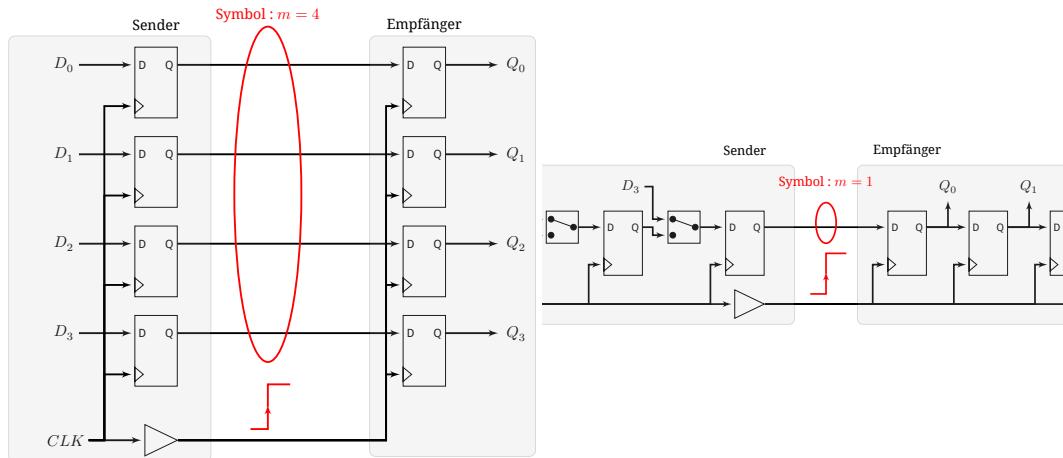


Abbildung 3.5: Beispiel eines Symbols in der parallelen / seriellen Datenübertragung

In Abbildung 3.5 ist jeweils das Symbol aus unserem Beispiel der parallelen und der seriellen Datenübertragung eingezzeichnet.

Symbolwert

Der Wert eines Symbols kann dabei maximal L Werte annehmen.

$$\begin{aligned} L &= 2^m & [L] &= 1 \\ m &= \log_2(L) \end{aligned} \quad (3.3)$$

Der Wertebereich eines Symbols ist wie folgt definiert:

$$\mathbb{R}_m = [0, \dots, 2^m - 1] \quad (3.4)$$

Achtung

Wir werden später auch auf Symbole treffen, welche aus Signalen mit $k > 2$ Spannungspegel zusammen gesetzt werden. (Binäres-Signal $k = 2$) zB. ein Tertiär-Code oder Quarternär-Code.

Dabei können wir noch immer eine Bitrate berechnen. Wir müssen zunächst den Symbolwert bestimmen $L = k^m$ und danach die äquivalente Anzahl Bits ausrechnen $m = \log_2(L)$.

Info 9: Definition des Symbols

Symbolrate / Baudrate

Die Symbolrate ist die Rate, mit welcher wir Symbole versenden. Diese wird auch Baudrate genannt. Wir können dies auch über die Zeit definieren : Die Symbolrate ist die Rate bei der eine definierte Menge Symbolen $\Delta Symbol$ in einer definierten Zeitspanne Δt übertragen wird.

$$B_d = \frac{\Delta Symbol}{\Delta t} = \frac{R_b}{m} \quad [B_d] = \text{Baud} \quad (3.5)$$

Die Symbolrate entspricht dem *Sende-Takt* f_{clk} in unseren Beispielen.

Symboldauer

Betrachten wir die Symbolrate resp. die Baudrate als eine Art Frequenz, welche die Symbol-Änderungsrate angibt, so ist die Symboldauer die Zeit welche ein Symbol auf der Signalleitung anliegt.

$$T_s = \frac{1}{B_d} = \frac{1}{f_{clk}} \quad [T_s] = s \quad (3.6)$$

Bitwertigkeit

Die Bitwertigkeit [W9] ist die sog. Bit-Reihenfolge. Sehen wir uns das Abbildung 3.4 nochmals an. Wir sehen, dass zuerst das $[D_3]$ Bit übertragen wird. Das muss nicht zwingend so sein. Und es gibt durchaus Übertragungsstandards, welche die umgekehrte Reihenfolge definieren.

Für uns steht das $[D_3]$ in Abbildung 3.4 für das höchstwertigste Bit. Wir nennen diese Bit-Reihenfolge «MSB-0» oder «MSB-first». Wenn wir die Daten anders herum, also zuerst mit dem niederwertigsten Bit $[D_0]$ die Daten übermitteln, sprechen wir von «LSB-0» oder «LSB-first».

Als Vertreter der MSB-0 Reihenfolge sei hier I2C genannt und dem gegenüber als LSB-0 Vertreter RS-232.

Aufgabe 3.16 : Bit and Bitrates

Frage

Nehmen wir an, wir übertragen mit einer Taktrate von $f_{Clk} = 1 \text{ kHz}$ Daten über die Schnittstelle aus Abbildung 3.2

Wie gross sind die folgenden Parameter:

- Bitrate R_b
- Baudrate B_d
- Symbolwert L

Antwort

$$m = 4$$

$$L = 2^m = 2^4 = 16$$

$$R_b = \Delta bits \cdot f_{Clk} = 4 \text{ bit} \cdot 1 \text{ kHz} = 4 kbit/s$$

$$B_d = \Delta Symbol \cdot f_{Clk} = 1 \text{ Symbol} \cdot 1 \text{ kHz} = 1 kBaud$$

Aufgabe 3.17 : Serial Bit and Bitrates

Frage

Nehmen wir an, wir übertragen mit einer Taktrate von $f_{Clk} = 4 \text{ kHz}$ Daten über die Schnittstelle aus Abbildung 3.4

Wie gross sind die folgenden Parameter:

- Bitrate R_b
- Baudrate B_d
- Symbolwert L

Antwort

$$m = 1$$

$$L = 2^m = 2^1 = 2$$

$$R_b = \Delta bits \cdot f_{Clk} = 1 \text{ bit} \cdot 4 \text{ kHz} = 4 kbit/s$$

$$B_d = \Delta Symbol \cdot f_{Clk} = 1 \text{ Symbol} \cdot 4 \text{ kHz} = 4 kBaud$$

Es verwundert Sie nun vielleicht ein wenig. Aber die Symbolrate ist gar nicht so aussagekräftig wie viele Daten wir übermitteln. Die Baudrate ist schlussendlich eine «verschwürbelte» Art, die Taktfrequenz zu benennen.

Die Bitrate gibt uns dabei die Geschwindigkeit an, wie schnell wir Daten übermitteln können.

Aufgabe 3.18 : kopieren

Frage

Sie kopieren auf Ihrem PC eine Datei von ihrem USB Laufwerk auf Ihre Festplatte.

Sie benötigen dafür $\Delta t = 14 \text{ sec}$ und die Datei ist 28 MB gross.

Berechnen Sie die Bitrate dieser Übermittlung.

Antwort

$$R_b = \frac{\Delta bits}{\Delta t} = \frac{28 \cdot 10^6 \cdot 8}{14} = 16000000 \text{ bit/s} = 16 \text{ Mbit/s}$$

Beachten Sie : MB = MegaByte : Die SI-Einheit der Bitrate ist aber «Bit» nicht Byte.

Info

Sie haben vielleicht im Zusammenhang mit einem Internet Abo den Begriff Bandbreite gehört : «jetzt mit 100MBit Bandbreite»

Sie wissen nun, dass diese Werbung an so vielen Stellen falsch ist.

1. 100 MBit ist eine Datenmenge und dabei noch falsch geschrieben
2. Bandbreite meint eigentlich Datenrate
3. Die Abkürzung für die Datenrate ist Mbit/s

Merken Sie sich : Die Bandbreite bezeichnet die Durchlass- oder Stop-Bandbreite eines Filters. Im Zusammenhang mit der Kommunikationstechnik, bezeichnet die Bandbreite das «brauchbare» Frequenzband, in welchem wir Daten übermitteln können.

Info 10: Fehlinterpretation «Bandbreite» in der Werbung

3.1.5 Kurzer Exkurs : Modulation

Wie sich in der Vergangenheit häufig gezeigt hat, werden die Begriffe «Bitrate» und «Symbolrate» schlecht verstanden. Daher machen wir hier einen kleinen Exkurs in das Thema der Modulation.

Wie wir in Kapitel 2.2.2 gesehen haben, unterscheiden wir die Bitwichtigkeit eines digitalen Signals in dem wir die Spannung des Signals messen. Es ist ein **0** wenn der Pegel unterhalb eines Schwellwert liegt und ein **1** wenn es über dieser Schwelle liegt.

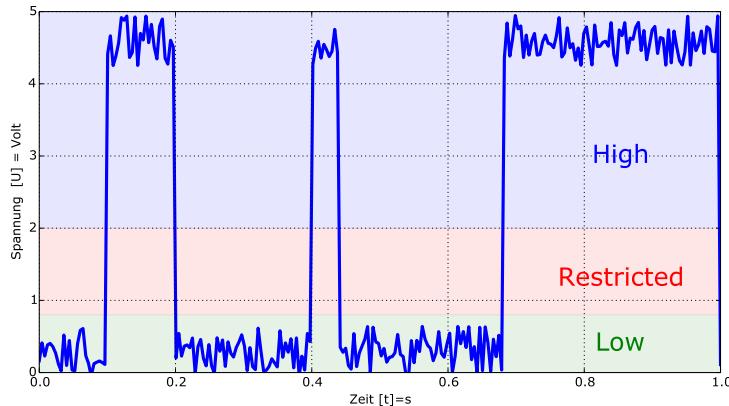


Abbildung 2.8 : Digitales Signal unterteilt nach dem TTL-Standard [W40]

Wir sehen hier nochmals Abbildung 2.8.

Nun ändern wir die Pegel des Schwellwerts und setzen noch zwei weitere Schwellen ein.

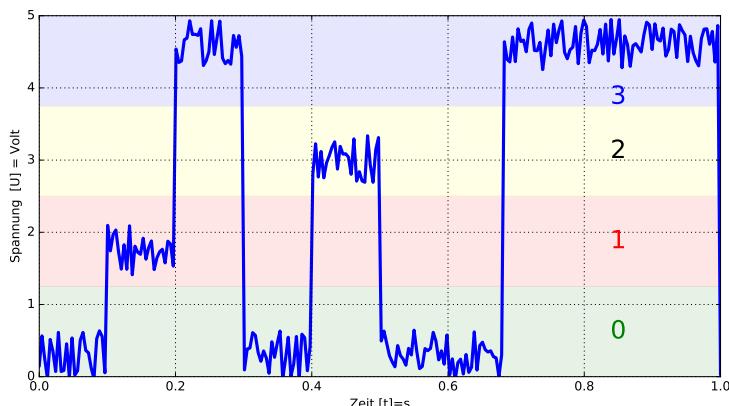


Abbildung 3.6: Quarternär Signal

Wir sehen in Abbildung 3.6 ein sog. «quarternär Signal». Der Wortteil «quart» kommt aus dem Lateinischen (*quattuor* = 4) und zeigt uns auch sprachlich das ein quarternär Signal aus 4 Signalpegeln besteht. Wir können mit jedem Takt-Zyklus somit 4 Zustände, resp. 2 Bit gleichzeitig übertragen.

Vergleichen wir dies mit einem klassischen binären Signal können wir die doppelte Bitrate übertragen.

$$R_{\text{b-quart}} = 2 \cdot R_{\text{b-binär}} \quad (3.7)$$

Nun zur interessanten Frage:

Aufgabe 3.19 : Tertiär Signal

Frage

Berechnen Sie die Bit- und die Symbolrate eines tertiar Signals im Vergleich zu einem binären Signal mit gleicher Übertragungsrate. Das tertiar Signal besteht aus 3 möglichen Signalpegeln.

Antwort

Fangen wir hier mit besser mit der Symbolwert an :
Wir wissen, wir haben pro Zeiteinheit 3 Zustände $L_{\text{tertiär}} = 3$. Somit können wir die Anzahl Bits pro Symbol ausrechnen.

$$m_{\text{binär}} = \log_2(L) = \log_2(2) = 1$$

$$m_{\text{tertiär}} = \log_2(L) = \log_2(3) = 1.5849 \text{ bit}$$

Das heisst, dass pro Symbol eines tertiar Signals 1.5849 Bits übertragen werden können.
Damit können wir auch gleich sagen, dass die Bitrate des tertiar Signal $R_{\text{b-tertiär}} = 1.5849 \cdot R_{\text{b-binär}}$ beträgt.

Und für viele überraschend : Die Symbolrate ist bei beiden Signalen identisch!
Wir übertragen die Symbole ja nach Aufgabenstellung immer mit der selben Rate!

$$B_{\text{d-binär}} = B_{\text{d-tertiär}}$$

3.2 LPT : Parallele Schnittstelle

Sehen wir uns nun eine «echte» Schnittstelle an. Zunächst kurz die sog. «parallele Schnittstelle» Centronics / IEEE1284 (LPT) [W50]. Diese wurde von der Firma Centronics in den 1970 Jahren für seine Drucker spezifiziert.

Die Schnittstelle benutzt einen D-Sub Stecker mit 25 Pins.



Abbildung 3.7: Parallel Port Female D-Sub25 Stecker
https://upload.wikimedia.org/wikipedia/commons/f/fa/Parallel_computer_printer_port.jpg

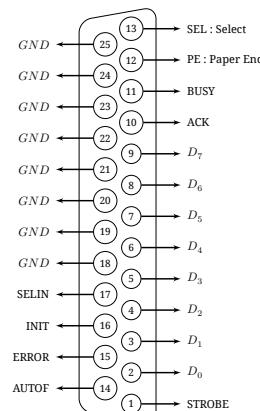


Abbildung 3.8: Parallel Port Female D-Sub25 Pinout

In Abbildung 3.8 ist die Anschlussbelegung für den «female» Stecker abgebildet. Wir sehen zum einen diverse Datensignale [D_0, \dots, D_7] und das sog. Strobe Signale. Das Strobe Signal entspricht unserem Clk Signal aus Abbildung 3.2. Wir erkennen auch andere Signale: PE , SEL , $SELIN$, $INIT$, $ERROR$ und $AUTOFF$ welche für die Centronics Drucker spezielle Funktionen bereitstellte. Die Signale $Busy$ und ACK werden uns später noch weiterbeschäftigen und werden auch bei anderen Systemen verwendet.

Die parallele Schnittstelle war vor allem bei Bastlern beliebt. Man konnte die Datensignale $[D_0, \dots, D_7]$ einzeln setzen und ein *Strobe* Puls auslösen.

Damit genug zur parallelen Schnittstelle. Sie ist alt, langsam und benutzt einen unglaublich grossen Stecker. Es wurde Zeit das sie verschwindet.

3.3 RS-232 : Serielle Schnittstelle

Sehen wir uns nun die serielle Schnittstelle an. Diese wurde in den 1962 Jahren definiert und schlussendlich als RS-232 [W62] 1969 spezifiziert.

Die Schnittstelle wurde für die Verbindung von PC und Modem entworfen und hat hierfür ein paar Eigenheiten. Man hat sich dazumals für eine serielle Übertragung entschieden, da die Datenraten auf dem Modem sehr gering waren und man vor allem die Anzahl Signalleitungen im Kabel klein halten wollte.



Abbildung 3.9: Serial Port Male / Female D-Sub9 Stecker
<http://media.digikey.com/Photos/CnC%20Tech/720-10020-00300.jpg>

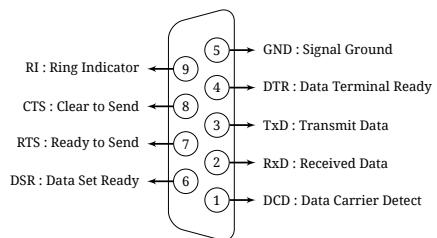


Abbildung 3.10: Serial Port Female D-Sub9 Pinout

In Abbildung 3.10 ist die Anschlussbelegung des female D-Sub 9 Steckers angegeben, so wie er bis vor ein paar Jahren bei jedem PC zu finden war. Wie wir gibt es die folgenden Signale: *RI*, *CTS*, *RTS*, *DSR*, *DTR*, *TxD*, *RxD*, *DCD*. Das Signal *TxD* entspricht unserem Daten-Sende-Signal aus Abbildung 3.4 und *RxD* entspricht unserem Daten-Empfangs-Signal aus Abbildung 3.4. Die RS-232 Schnittstelle ist demzufolge fähig Daten zu senden und zu empfangen!

Aufgabe 3.20 : welches Signal fehlt bei RS-232?

Frage

Wie sie sicherlich schon gemerkt haben fehlt in Abbildung 3.10 ein Signal.
 Welches?

Antwort

Es ist das *Clk* Signal! Aber wir wissen : Es muss eines geben!

Wie sieht nun so ein Signal auf dem Kabel aus?

3.3.1 Leitungscode

Die Beschreibung wie eine Bitfolge auf dem Kanal dargestellt wird, nennt man den «Leitungscode». Bei RS-232 steckt die Bitfolge in der Spannungspegel auf dem *RxD / TxD* Signal.

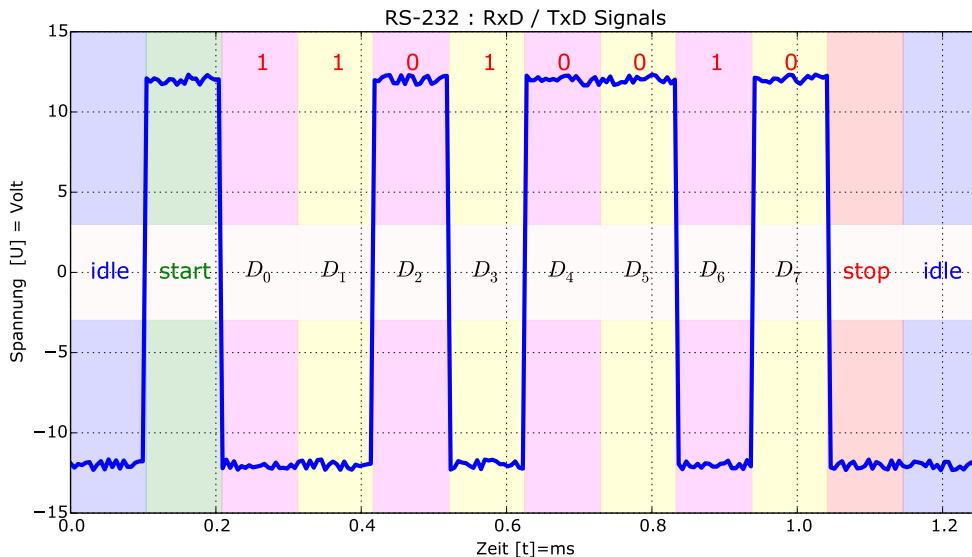


Abbildung 3.11: RS-232 Leitungscode

Sehen wir uns die Abbildung 3.11 genauer an.

Zunächst sehen wir die Spannungsfolge. Diese bildet eine ganze «Transaktion» mit 8 Bit Daten ab. Die zulässigen Spannungspegel sind bei RS-232 folgendermassen definiert:

Bit 0 Ein Spannungspegel zwischen $[+3V, \dots, +15V]$

Bit 1 Ein Spannungspegel zwischen $[-3V, \dots, -15V]$

Neben den eigentlichen Daten-Signalen gibt es noch ein *idle*-, ein *start*- und *stop-bit*.

Das Signal «idle» (engl. für Leerlauf) bezeichnet den Ruhezustand des Kommunikationssystems und hat immer den Zustand log. 1 Für RS-232 heisst das, dass wenn wir gerade keine Daten versenden, immer ein Signalpegel von $TxD_{idle} = [-3V, \dots, -15V]$ anliegen muss.

Interessant wird es beim *start-bit*. Die erste positive Flanke, nach einem *idle-Signal* ist das *start-bit*. Es ist immer auf einem log. 0 Pegel. Damit erkennt der Empfänger, dass nun gleich Datensignale gesendet werden und kann sich darauf vorbereiten.

Eine der Aufgaben des Empfängers ist es zum Beispiel sein internes Taksignal auf das *start-bit* zu synchronisieren.

Info

Bei RS-232 haben der Sender und der Empfänger eigene Signalgeneratoren, welche mit dem *start-bit* synchronisiert werden!

Info 11: RS-232 : Synchronisation der Sender- / Empfänger-Signalgeneratoren

Nach dem *start-bit* folgen die Daten-Bits. RS-232 kann auf verschiedene Datenformate konfiguriert werden. Das heisst es können zwischen [5 Bit, ..., 8 Bit] folgen. Danach kann ein optionales sog. «Parity-Bit» stehen. Aber dazu kommen wir später.

Am Ende folgt noch das *stop-bit*. Es ist immer auf einem log. 1 Pegel. Das *stop-bit* wird gebraucht, damit der Empfänger Zeit hat die empfangenen Daten zu verarbeiten und sorgt dafür, dass die Empfänger-Hardware wieder bereit ist neue Daten zu empfangen.

Das *stop-bit* ist aus gutem Grund auf dem gleichen Pegel wie das *idle* Signal. Haben wir nämlich gleich zwei Transaktionen hintereinander folgt nach dem ersten *stop-bit* gleich das *start-bit* der nächsten Transaktion.

Aufgabe 3.21 : idle Signalpegel bei RS-232

Frage

Wieso wurde der *idle* Signalpegel bei RS-232 nicht auf 0 V definiert?
Gibt es einen guten Grund dafür

Antwort

Natürlich gibt es einen guten Grund dafür. Wenn der Sender in der Ruhephase immer ein Signal aktiv auf einen definierten Pegel ziehen muss, so kann der Empfänger sehen, dass eine Verbindung zum Sender besteht. Somit weiß der Empfänger, dass das Kommunikationssystem kein Kabelunterbruch oder ähnliches aufweist.

Aufgabe 3.22 : RS-232 Baudrate

Frage

Bestimmen Sie die Baudrate aus der Abbildung 3.11.

Antwort

$$B_d = \frac{12 \text{ bit}}{12.5 \text{ ms}} = 9600 \text{ bit/s}$$

3.3.2 Einsatzgebiet

Die serielle Schnittstelle meist in der RS-232 Ausführung wird immer noch häufig in vielen industriellen Steuergeräten verwendet. Es ist daher sinnvoll das Sie diese Schnittstelle gut kennen. Obwohl RS-232 für vor allem für die Kommunikation mit einem Modem entwickelt wurde, können aufgrund der Einfachheit der Schnittstelle (und das darüber liegende Protokoll) auch diverse andere Geräte damit angesprochen werden.

Verbindung

Die RS-232 Schnittstelle verbindet unter normalen Bedingungen zwei Teilnehmer. Dies nennt man eine **point-to-point** Verbindung. (engl. für Punkt zu Punkt Verbindung) In der einfachsten Form wird nur eine Leitung vom *TxD* des Senders zum *RxD* zum Empfänger verbunden. Damit kann nur der Sender «sprechen» und der Empfänger kann nur «zuhören». Man sagt dazu auch, die Verbindung ist nur **Simplex** fähig. Werden beide *TxD* und *RxD* paare verbunden, können beide Teilnehmer senden, sowie empfangen. Man sagt dazu, dass die Verbindung (**Voll- Duplex**) fähig ist. Es gibt noch eine dritte Variante, dies nennt man **Halbduplex** und meint, dass jeder Teilnehmer senden und empfangen kann, aber nicht zur selben Zeit. Aber dazu später mehr.

Aufgabe 3.23 : RS-232 Verbinden von zwei Teilnehmern

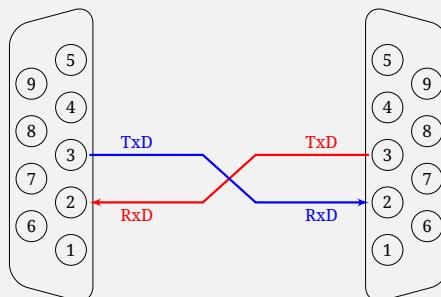
Frage

Verbinden Sie nun zwei RS-232 Teilnehmer mit einer Duplex Verbindung.
Dazu müssen Sie in der einfachsten Variante lediglich das TxD und oder das RxD Signal verbinden.

Antwort

DTE : Data Terminal Equipment (PC)

DCE : Data Communication Equipment (Modem)



Die anderen Signale sind für das sog. Handshaking und werden wir ebenfalls später betrachten

Aufgabe 3.24 : RS-232 Verbinden von mehr als zwei Teilnehmern?

Frage

Ist ein Verbinden von mehr als zwei Teilnehmern an RS-232 möglich? Was sind die Voraussetzungen, dass dies gehen würde?

Antwort

Nein, das ist nicht ohne weiteres möglich. Wir erinnern uns : Bei *idle* wird auf der TxD Leitung auf -15 V gezogen. Würde nun ein dritter Teilnehmer an das Netz angeschlossen, hätten wir mind. zwei TxD Leitungen zusammen geschlossen. Dies würde dazu führen, dass wenn ein Teilnehmer Daten senden würde, wir einen Kurzschluss erzeugen würden.

Damit wir mehr als zwei Teilnehmer an das gleiche Netz hängen können, muss die Schnittstelle dafür ausgelegt sein. zB. Open-Kollektor Ausgänge oder es muss durch das Protokoll sichergestellt sein, dass niemals zwei Teilnehmer gleichzeitig senden.

Praxis-Tipp

Man kann für RS-232 normale und sog. gekreuzte Kabel kaufen. Die normalen Kabel haben die einzelnen Signale direkt verbunden, während die gekreuzten Kabel die RxD, TxD und die weiteren Handshake Signale für RS-232 richtig ausgetauscht haben.

Blöderweise gibt es auch Hardware welche die Signale direkt auf dem Print ausgetauscht. Nehmen Sie im Zweifelsfall also immer ein normales und ein gekreuztes Kabel mit.

Info 12: RS-232 : normale und gekreuzte Kabel

3.4 UART : Universal Asynchronous Receiver Transmitter

Wir werden im Praktikum ein «USB zu Serial Adapter» verwenden. Das ist im wesentlichen ein «USB zu RS-232 Adapter» der andere Spannungspegel aufweist. Das interessante an diesem Adapter ist, dass wir auf unserem PC / Laptop einen Standard Treiber haben, welcher diesen Adapter wie eine RS-232 Schnittstelle behandelt.

Was ist nun die UART:

UART steht für «Universal Asynchronous Receiver Transmitter» und war die Grundlage für RS-232. Das *Asynchronous* steht dafür, dass Sender und Empfänger ihre eigene Taktquellen aufweisen. Die beiden Taktquellen sind demnach asynchron zu einander und werden nur kurzzeitig über das *start-bit* synchronisiert.

Receiver Transmitter dürfte klar sein : Empfänger und Sender.

3.4.1 Leitungscode

Sehen wir uns nun den Leitungscode von UART an:

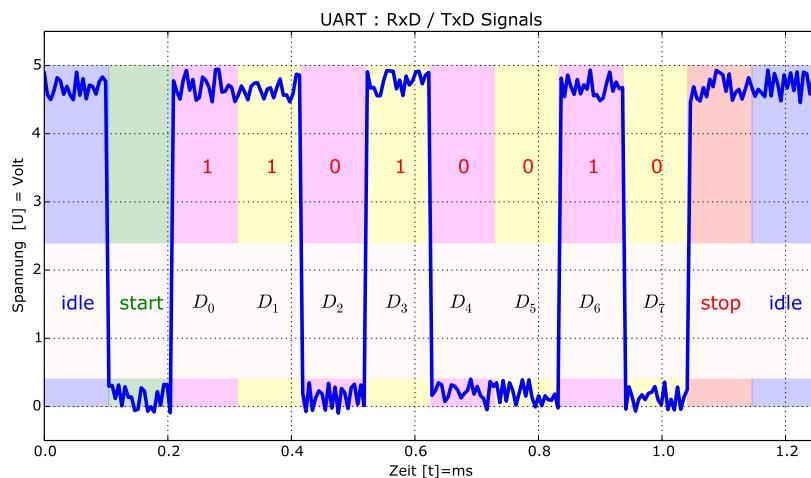


Abbildung 3.12: UART Leitungscode

Wir sehen in Abbildung 3.12 den Leitungscode einer Beispiel Übertragung. Die Datensignale sind die selben wie im Leitungscode von RS-232, siehe Abbildung 3.11. UART selbst definiert keine Spannungspegel, aber im Beispiel wurden TTL-Pegel verwendet.

Bit 0 = 0 V

Bit 1 = 5 V

Die Signalpegel sind somit umgekehrt wie bei RS-232. Aber der Rest ist identisch.

Das *idle* Signale entspricht wieder einem log. 1 und das *start-bit* einem log. 0. Danach folgen die Daten und danach ein optionales *parity-bit* welches hier nicht abgebildet ist. Am Schluss folgt das *stop-bit* mit einem log. 1 Pegel.

3.4.2 Optionen

Wie bereits erwähnt kann man bei RS-232 und auch UART diverse Parameter konfigurieren.

Wichtig

Die Parameter welche man für RS-232 und UART einstellen muss, damit man mit einem Gerät kommunizieren kann, muss man aus dem Manual des Geräts herauslesen!

Wenn man Pech hat, muss man es selbst herausfinden, durch messen und Sachverstand.

Info 13: RS-232 / UART Parameter

Anzahl Datenbits

Bei RS-232 / UART kann man die Anzahl Datenbits konfigurieren. Diese Modes sind nach dem Standard unterstützt.

- 8 Bit
- 7 Bit
- 6 Bit
- 5 Bit

Kleine Anmerkung: nicht jedes Betriebssystem unterstützt alle Modes. Meist wird ein 8 Bit oder 7 Bit Mode verwendet.

Baudraten

Für RS-232 / UART sind die folgenden Baudraten definiert:

75 Baud	110 Baud	134.5 Baud
150 Baud	300 Baud	600 Baud
1200 Baud	1800 Baud	2400 Baud
4800 Baud	7200 Baud	9600 Baud
14400 Baud	19200 Baud	38400 Baud
56000 Baud	57600 Baud	115200 Baud

Tabelle 3.1: RS-232 definerte Baudraten

Parity Bit

Es sind die folgenden Modes konfigurierbar. Was sie genau machen und was das Parity-Bit ist, später mehr.

- Odd-Parity
- Even-Parity
- None

Anzahl Stop-Bits

Eine weitere Parameter welcher man bei RS-232 einstellen kann ist die Anzahl Stop-Bits.

- 1
- 1.5
- 2

LSB / MSB first

Wir müssen nun noch weitere Begriffe kennen lernen:

Das erste ist «LSB / MSB first»

Wenn wir Daten seriell übertragen, können wir entweder mit höchstwertigsten Bit, dem MSB, beginnen oder mit dem niederwertigsten Bit, dem LSB.

Aufgabe 3.25 : UART Quiz**Frage**

Kleine Zwischenfrage : Was wird bei UART zuerst übertragen? Das LSB oder das MSB?

Antwort

Wir sehen in *Abbildung 3.12* das zuerst das LSB : D_0 gesendet wird.
Man nennt dies «LSB first»

Je nach Übertragungsstandard kann dies variieren. Beispielsweise wird bei I2C zuerst das MSB übertragen. Falls sie also mal eine Datenübertragung messen ist es möglich, dass die Daten «verkehrt» gemessen werden.

Endianess

Die nächsten wichtigen Begriffe sind: «big-endian» und «little-endian»

Aufgabe 3.26 : UART Quiz (2)**Frage**

Stellen wir uns vor, wir haben ein UART System mit 8 Bit pro Übertragung und wir wollen eine 32 Bit Zahl übertragen.

Wir werden mit diesem UART System die 32 Bit Zahl in 4 Bytes aufteilen und übertragen.
Welches Byte wird zuerst übertragen? Das höchstwertigste Byte oder das niederwertigste Byte

Antwort

Leider ist keine Antwort richtig. Die Endianess wurde bei UART nie standardisiert.
Der Programmierer muss wissen wie die Endianess von Sender und Empfänger definiert ist
oder muss sie selbst definieren.

Wenn wir bei einem System zuerst das höchstwertigste Byte übertragen, so nennen wir das System ein **big-endian** und wenn wir zuerst das niederwertigste Byte übertragen ist es ein **little-endian** System.

3.5 Fazit

Wir haben in diesem Kapitel zunächst die Grundlagen der Seriellen und der Parallelen Datenübertragung besprochen. Danach haben wir 3 Kommunikationsschnittstellen diskutiert welche diese Grundlagen recht gut umsetzen.

Einige Themen (wie zB. den Leitungscode oder das Parity-Bit) werden wir in einem späteren Kapitel noch genauer Betrachten.

Die Formeln für die folgenden Parameter sind aber nun bekannt und werden bei einer Prüfung auch vorausgesetzt:

- Bitrate
- Symbolwert
- Symbolrate
- Symboldauer

Sie müssen wissen was die Bitwichtigkeit ist und welche Varianten es gibt.

Sie wissen ausserdem wie Sie eine RS-232 für ein *Simplex*- oder *Duplex*- System zusammenschalten müssen.

4

Standardisierung und das OSI-Kommunikationsmodell

Inhalt des Kapitel

4.1	Standardisierung	56
4.1.1	Definition	56
4.1.2	Organisation von Standardisierungsgremien	57
4.1.3	Wichtige Standardisierungsgremien	57
4.1.4	Internet Standardisierung	58
4.1.5	Ablauf der Standardisierung für Internetstandards	59
4.2	ISO/OSI Modell	60
4.2.1	Schichtenmodell	60
4.2.2	Datenübertragung im Schichtenmodell	61
4.2.3	Gliederung des OSI/ISO Referenzmodells	62
	OSI Layer 7	63
	OSI Layer 6	63
	OSI Layer 5	63
	OSI Layer 4	63
	OSI Layer 3	63
	OSI Layer 2	63
	OSI Layer 1	64
	Übersicht Layer 1-2	64
4.2.4	Protokoll Stack	67
4.2.5	Bestandteile des TCP/IP Protokoll Stacks	67
4.2.6	Beispiel Schichtenmodell	68
4.3	Zusammenfassung	69

Zum Inhalt Die Kommunikationstechnik ist ein komplexes Fachgebiet. Es verbindet viel «know-how» im Gebiet der analogen Schaltungstechnik, der digitalen Signalverarbeitung und der Informatik. Damit man in diesem Gebiet den Überblick behalten kann, wurden die einzelne Probleme in beherrschbare Größen unterteilt und wo nötig Standardisiert. Wie wir weiter sehen werden, sind die heutigen Kommunikationssysteme in Schichten unterteilt und diese Schichten sind mit standardisierten Protokollen oder Spezifikationen definiert.

4.1 Standardisierung

4.1.1 Definition

Als Standard bezeichnet man die Auflistung von *Regeln, Bedingungen, Anforderungen* oder *Prozeduren*, welche die *Auslegung*, den *Einsatz*, oder die *Arbeitsweise von Bauteilen, Komponenten, Systemen* oder *Verfahrensweisen* beschreibt.

Wichtig

Ein Standard legt fest was zu machen ist, aber nicht wie

Ein integraler Teil der Kommunikationstechnik sind Standards.

Vergleichen wir das mit dem Menschen:

Wir haben in unserem Umfeld diverse Standards gebildet. Zum Beispiel unsere Sprache. Wir können uns untereinander Unterhalten, da wir die gleiche Sprache sprechen. Während sich die Standards für Sprachen regional quasi von alleine gebildet haben, müssen die Standards für die Technik erst von jemanden definiert werden.

Entwerfen Firmen eigene Standards, welche sie nicht offen legen, so spricht man von einem **proprietary Standard**. Diese sind in der Regel nicht öffentlich zugänglich und mit den Lösungen anderer Hersteller inkompatibel. Diese Standards werden heutzutage in der Industrie nur noch schwer aufgenommen. Wird ein Standard von einer Firma oder von einem Gremium definiert und der Allgemeinheit zugänglich gemacht (auch durch Lizenzierung) spricht man von einem **offenen Standard**.

Ist ein Standard durch seine weite Verbreitung oder durch die Verknüpfung an ein spezielles Verfahren untrennbar verbunden, so spricht man von einem **de-facto Standard** oder auch **Industriestandard**.

Wichtig

proprietary Standard nicht öffentlich

offener Standard öffentlich, lizenzierbar

de-facto Standard durch Verbreitung, zum quasi Standard erhoben

4.1.2 Organisation von Standardisierungsgremien

Die Standardisierungsgremien sind unter sich organisiert. So gibt es solche für internationale Standards und solche für nationale Standards. Generell werden wir mit Standards konfrontiert, welche aus einer der folgenden Quelle hervorgegangen sind

internationale Gremien	<ul style="list-style-type: none"> • Beispiel : ITU, ISO • bestehen aus Staaten, Firmen und Interessengruppen (auch Mischformen) • Hier entstehen generalisierende als auch spezielle Standards
europäische Gremien	<ul style="list-style-type: none"> • Beispiel : CEPT, ECMA • bestehen aus (europ.) Staaten, Firmen und Interessengruppen • generalisierende und spezielle Standards zur Harmonisierung (GSM-Netz, Euro-ISDN)
nationale Gremien	<ul style="list-style-type: none"> • Beispiel : ANSI, IEEE, DIN, SNV • bestehen aus öffentlichen Institutionen, Interessengruppen von Herstellern und Anwendern • nationale Standards und Vorschläge für internationale Gremien
Hersteller	<p>Anbei einige Beispiel von Quasi-Weltstandards:</p> <ul style="list-style-type: none"> • der Befehlssatz des Modem-Herstellers HAYES • das Betriebssystem Windows von Microsoft • die Office-Anwendungen von Microsoft • die Apple-Bedienung vom iPhone

Tabelle 4.1: Organisation von Standardisierungsgremien

4.1.3 Wichtige Standardisierungsgremien

ISO	<ul style="list-style-type: none"> • Internationale Standardization Organisation • internationaler Zusammenschluss nationaler Institutionen • ISO entwickelte Standards wie z.B. das OSI Referenzmodell
ITU	<ul style="list-style-type: none"> • International Telecommunications Union • Unterorganisation der UNO und wurde 1865 gegründet • Für den Telematikbereich ist deren Unterorganisation die ITU-T von bedeutender Wichtigkeit
CCITT	<ul style="list-style-type: none"> • Comite Consultatif International Telegraphique et Telephonique • Die CCITT entwickelte Standards wie z.B. V.22, V.32, V.34 • Diese Standards werde heute als ITU-T Standards bezeichnet
ITU-T	<ul style="list-style-type: none"> • Telecommunication Standardization Sector der ITU • Unterorganisation der ITU und beinhaltet die CCITT Standards • Die ITU-T entwickelte Standards wie z.B. V.24, X.21, X.25
ANSI	<ul style="list-style-type: none"> • American National Standards Institute • private, amerikanische, gemeinnützige Organisation und wurde 1918 gegründet • Die ANSI vertritt die USA in der ISO • entwickelt allgemeine Standards wie z.B. ASCII
IEEE	<ul style="list-style-type: none"> • Institute of Electrical and Electronic Engineers • amerikanische Vereinigung von Ingenieuren und wurde 1884 gegründet • IEEE entwickelte und standardisierte das IEEE Architekturmodell für LAN's (Schichten 1 und 2) wie z.B. 802.3 CSMA/CD (Ethernet)

Tabelle 4.2: Wichtige Standardisierungsgremien

Interessanterweise werden Standards meistens von «unten noch oben» entwickelt. Damit meint man, dass Standards zunächst von Firmen oder Organisationen vorgeschlagen werden (zB. IEEE / ETSI) und später von Nationalen oder Internationalen Gremien (zB. ITU / ISO) übernommen werden.

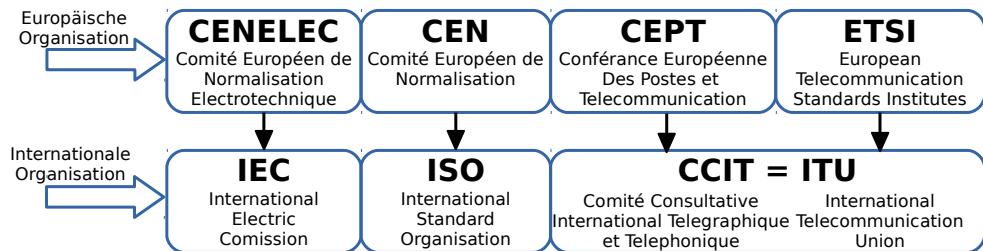


Abbildung 4.1: Standardisierungsgremien

4.1.4 Internet Standardisierung

Für die Standardisierung der Internet Standards haben sich diverse Gremien und Organisationen gebildet. Hier kurz die Wichtigsten:

Internet Society (ISOC)	Die Internet Society ist eine Vereinigung, welche sich mit den sozialen, politischen und technischen Aspekten des weltweiten Internet mit folgendem Ziel beschäftigt: « Continues to develop as an open platform that empowers people to share ideas and connect in new and innovative ways»
Internet Architecture Board (IAB)	Das Internet Architecture Board ist das technische Beratungsorgan des ISOC und ist für die Ernennung der IESG-Mitglieder und IETF-Gruppenleiter zuständig.
Internet Engineering Steering Group (IESG)	Die Internet Engineering Steering Group ist für die organisatorische Abwicklung der Internet Standardisierung nach RFC 1602 und die Aktivitäten der Internet Engineering Task Force und der Internet Research Task Force zuständig.
Internet Engineering Task Force (IETF)	Die Internet Engineering Task Force erarbeitet Vorschläge für die technische Verbesserung und Weiterentwicklung des Internet und bereitet die dazu notwendigen Standards vor. Es bestehen folgende 8 Untergruppen innerhalb der IETF: <ul style="list-style-type: none"> • Applications • Internet • Network Management • Operational Requirements • Routing • Security • Transport • User Services
Internet Research Task Force (IRTF)	Die Internet Research Task Force behandelt die langfristigen Forschungsprojekte.
Internet Assigned Number Authority (IANA)	Die Internet Assigned Number Authority verwaltete und publizierte im Auftrag des IAB bis 1998 alle im Internet vorkommenden Adressen, Keywords und Nummern.
Internet Corporation for Assigned Names and Numbers (ICANN)	1998 wurde mit der ICANN eine internationale Organisation gegründet und mit der Verwaltung der verschiedenen Namens- und Adress-Räume beauftragt.

Tabelle 4.3: Internet Standardisierung

4.1.5 Ablauf der Standardisierung für Internetstandards

Exemplarisch für andere Standards diskutieren wir, wie die Standardisierung der IETF abläuft.
Beim IETF durchläuft ein Standard die folgenden Phasen:

1. Proposed Standard
2. Draft Standard
3. RFC : Request for Comment
4. Standard

Zunächst wird der Standard als **Proposed Standard** intern bearbeitet. Sobald dieser eine gewisse Qualität und Stabilität erreicht hat, wird dieser als **Draft Standard** einer größeren Zielgruppe zugänglich gemacht. In dieser Phase werden nur noch geringe Änderungen vorgenommen.

Vor der Veröffentlichung des Standards, werden die Papiere als sog. **Request for Comment** im Internet publiziert. Es können sich nun alle User zum Standard äußern. Falls keine weitreichenden Änderungen vorgeschlagen werden, wird der Standard veröffentlicht.

Aufgabe 4.27 : Standards Definition

Frage

Was steht in einem Standard drin?

Antwort

Ein Standard legt fest was zu machen ist, aber nicht wie

Aufgabe 4.28 : Standard Gremium

Frage

Welches Gremium standardisierte die Standards V.24, X.21, X.25?

Antwort

ITU-T
V.24 ⇒ RS-232
X.21 ⇒ ≈ RS-485
X.25 ⇒ ≈ europäisches «Internet» Protokoll

Aufgabe 4.29 : Welches Gremium ist das?

Frage

Welches Gremium standardisierte die Ethernet Standards wie zB, IEEE802.3?

Antwort

IEEE Institute of Electrical and Electronic Engineers
IEEE 802.3 ⇒ kabelgebundenes Ethernet

4.2 ISO/OSI Modell

In diesem Kurs ist das sog. ISO / OSI Referenzmodell (Open Systems Interconnection Model) von zentraler Bedeutung. Aber zuerst ein wenig Hintergrundwissen. Wir schreiben das Jahr 1977. Commodore, derzeit Taschenrechner Hersteller bringt gerade sein Commodore PET heraus und positioniert sich Hersteller von «home computers». So etwas wie eine Netzwerkschnittstelle hatten diese «Rechenknechte» nicht. Jeder Hersteller hat seine eigenen inkompatiblen Standards entwickelt. In der Regel haben Sie die Zusatzhardware (zB ein Drucker) vom Computerhersteller gekauft, da andere damit nicht funktionieren. Eine Verbindung von zwei Computern des gleichen Herstellers konnte schon zur Glücksache werden.

In dieser Zeit beschloss die ISO, dass man ein Referenzmodell entwickeln sollte, dass die Kommunikation eines Telekommunikationssystems oder eines Computersystems, ohne Rücksicht auf die unterliegende interne Struktur und Technologie ermöglicht. Ebenfalls beauftragte das CCITT ein Modell, welches erstaunlicherweise beinahe identisch mit dem ISO/OSI Modell war.

Nice to Know

Das ISO/OSI Referenzmodell wurde schlussendlich 1984 unter dem Namen **ISO 7498** und unter der CCITT als **X.200** veröffentlicht.

4.2.1 Schichtenmodell

Beide Gremien kamen zum Schluss, dass es sinnvoll ist, das Referenzmodell in unabhängige *Abstraktionschichten* (*Layers*) aufzuteilen. Jede Schicht, resp. Layer, stellt dem darüber liegenden Layer einen Dienst an. Dieser wird über einen *Service Access Point* (*SAP*) zur Verfügung gestellt. Die Verbindung von einem Layer zum andern wird als *Interface* (*Schnittstelle*) bezeichnet.

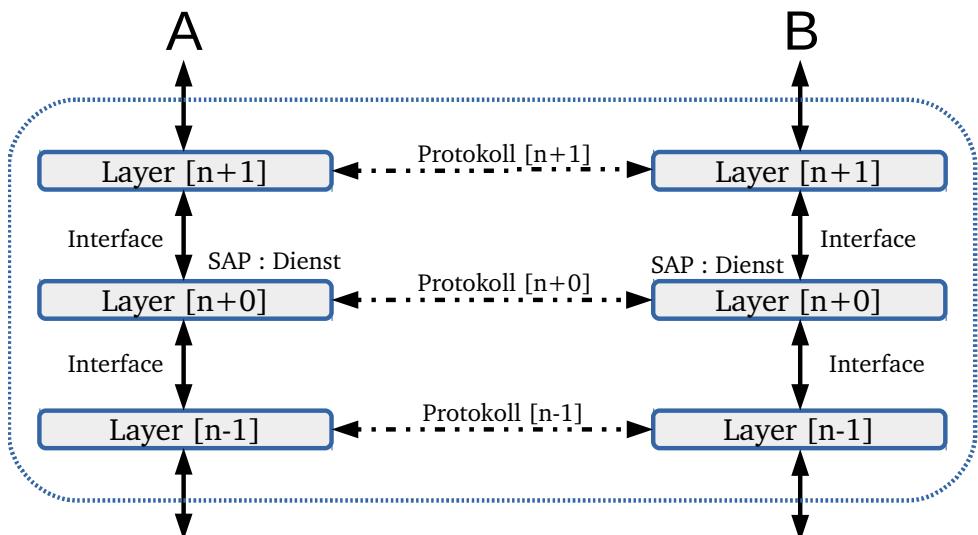


Abbildung 4.2: Schichtenmodell

Das interessante dabei ist, dass eine höher liegende Schicht (Layer[n+1]), nichts über die Funktion einer darunterliegenden Schicht (Layer[n]) weiß. Kommuniziert eine Schicht mit einer «entfernten» Partnerschicht, so werden die Regeln, nach denen kommuniziert wird, als *Protokoll* bezeichnet. Es erscheint der aktuellen Schicht, als würde Sie direkt mit der Partnerschicht über das Protokoll schreiben. Tatsächlich wandern die Befehle auf der einen Seite durch alle darunter liegenden Schichten, bis der Sender und der Empfänger auf der untersten Schicht physikalisch miteinander verbunden sind. Danach propagieren die Daten hinauf bis zur korrespondierenden Schicht.

4.2.2 Datenübertragung im Schichtenmodell

Sehen wir uns die Datenübertragung im Schichtenmodell genauer an:

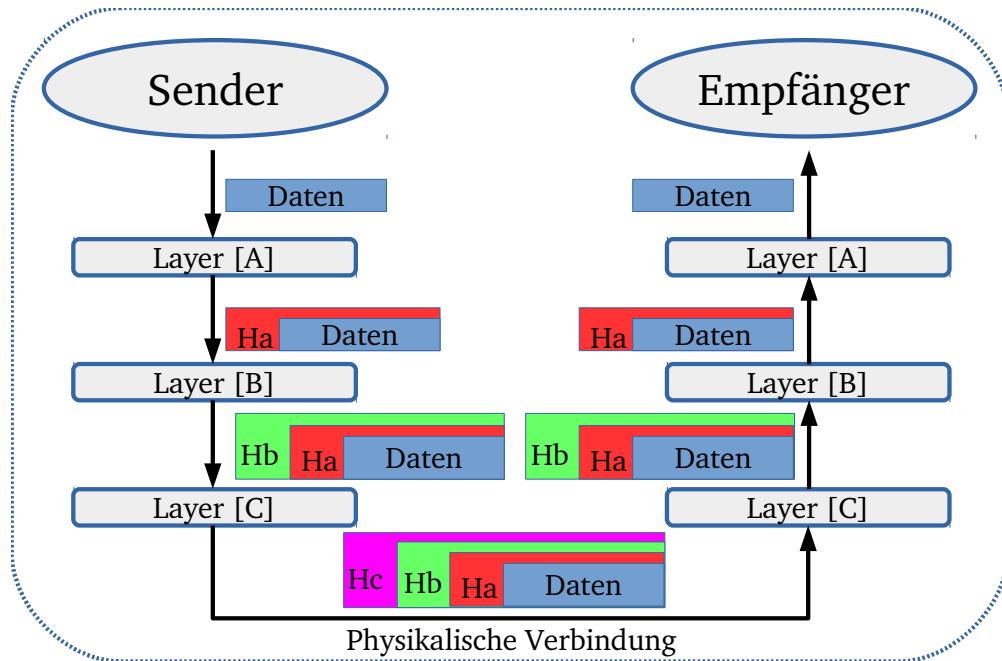


Abbildung 4.3: Datenübertragung im Schichtenmodell

Der Sender verschickt seine Daten indem er diese dem *Layer A* des Übertragungssystems übergibt. Über den SAP des *Layers A* kann er, falls nötig, die Adresse des Empfängers mitgeben. Die *Layer A* Schicht fügt dem Datenpacket zusätzliche Informationen hinzu. Diese Zusatzinformationen werden **Header** genannt und «umschliessen» das Datenpacket. Die *Layer B* Schicht macht dies ebenfalls. Nur wird der Header um das gesamte Daten und Header des *Packets A* gelegt. Ebenso bei *Layer C*. Da dies die unterste Schicht darstellt, ist hier die physikalische Verbindung zum Empfänger angebracht. In *Layer C* des Empfängers werden die Daten und Header entgegengenommen und verarbeitet. Außerdem werden die Headerinformationen des *Header C* entfernt und an die höhere Schicht weiter geleitet. Ebenso bei *Layer B* und *Layer A*, bis nur noch die Daten vorhanden sind. Diese werden dem Empfänger übergeben. Was genau im Header drin steht wird im Protokoll des Layers definiert. Daher werden die im Header enthaltenen Informationen auch als Protokollinformationen bezeichnet. Die Protokollinformationen dienen der Umsetzung des Protokolls. So geben diese Auskunft darüber, wer die Daten abgesendet hat und wer sie empfangen soll, welchen Weg sie während der Übertragung nehmen sollen, wie sie weiterverarbeitet werden dürfen oder wie sie vom Empfänger behandelt werden sollen.

Hier in dieser Grafik sieht man übrigens den ersten grossen Nachteil eines Schichtmodells:

Wichtig

Je mehr Schichten, desto mehr Headerinformationen werden versendet. Es steigt die zu übertragende Datenmenge an, obwohl nicht mehr Nutzdaten übertragen werden. Das Kommunikationssystem als ganzes wird ineffizient.

4.2.3 Gliederung des OSI/ISO Referenzmodells

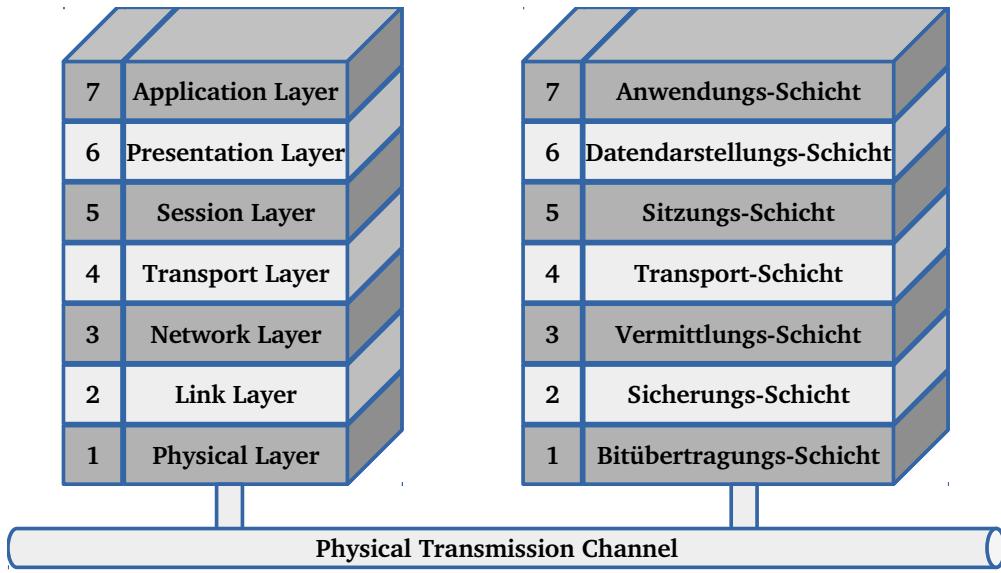


Abbildung 4.4: ISO/OSI Modell

Das Standard ISO/OSI Referenzmodell hat klar definierte Aufgaben für die einzelnen Schichten. Hier kurz einen Überblick:

ISO/OSI Referenzmodell					
Layer		Einheit	Funktion		
Application Layer	7. Application		Daten	High-Level API's, File access, Virtual Terminals	
	6. Presentation			Zeichencodierung, Datenkompression, Verschlüsselung	
	5. Session			Verbindungs-aufbau, -haltung, -abbau	
Transport Layer	4. Transport		Segmente	Flusskontrolle, Fehlersicherung (Zuverlässige Verbindung), ACK/NACK, Multiplexing	
	3. Network		Packet / Datagram	Addressierung, Routing, Quality of Service	
	2. Data Link	LLC Logical Link Control	Bit / Frame	Framebildung, Fehlerhandlung (Blockcodes, CRC), logischer Kanal	
		MAC Media Access Control		Medium Access (CSMA/CD, FDM, TDM)	
	1. Physical		Bit	Modulation, Leitungscodierung	

Tabelle 4.4: ISO/OSI Referenzmodell

Die *Layer 1-4* werden zusammen auch als *Transport Layers* bezeichnet, die *Layer 5-7 Application Layers*. Die *Transport Layers* sind stark auf die verwendete physikalische Schnittstelle ausgelegt, während die *Application Layers* bereits stark abstrahiert sind und unabhängig von der verwendeten Schnittstelle definiert werden können. Der *Data Link Layer* wird häufig zusätzlich unterteilt. Zum einen in den *Logical Link Control Layer*, welcher sich um die Sicherung der gesendeten / empfangenen Daten kümmert, während der *Media Access Control Layer* den Zugriff auf das verwendete Medium definiert.

OSI Layer 7

OSI-Layer 7 wird auch **Anwendungsschicht** oder **Application Layer** genannt und beschreibt wie das Protokoll mit dem Benutzer interagiert. (zB. GUI, Konsolenkommandos etc) Aber auch die **Schnittstellen** zu anderen Protokollen.

OSI Layer 6

OSI Layer 6 wird auch **Datendarstellungsschicht** oder **Presentation Layer** genannt (selten auch als **Syntax Layer** bezeichnet) und beschreibt wie die Daten des *Application Layer* für die Versendung **formatiert, komprimiert** oder **verschlüsselt** werden.

OSI Layer 5

OSI Layer 5 wird auch **Sitzungsschicht** oder **Session Layer** genannt und beschreibt wie eine Verbindungsorientierte Verbindung (session) über einen beliebigen Vermittlungsschicht **aufgebaut, abgebaut** und bei Verlust **wiederhergestellt** wird. Der Session Layer ist ausserdem dafür zuständig, das eine Session **autorisiert** und **authentifiziert** wird.

Eine Session kann **stateful** (weist einen Zustand auf) oder **stateless** (Zustandsfrei) sein.

Wichtig

TCP/IP weist keinen Session Layer auf. (Jede TCP Verbindung ist eine kleine Session) Falls doch ein Session Management benötigt wird, muss dies die Applikation selbst definieren / verwalten.

OSI Layer 4

OSI Layer 4 wird auch **Transport Layer** oder **Transportschicht** genannt und beschreibt wie eine **Punkt-zu-Punkt** Verbindungen über ein Netz aufgebaut und abgesichert werden. Ebenfalls Aufgabe des Transport Layer ist die **Flusskontrolle auf Verbindungsebene**. Nicht zu vergessen ist die sog. **same-order-delivery**, der Network Layer garantiert uns **nicht** die richtige Reihenfolge der Datenpakete. Dies wieder zu ordnen ist die Aufgabe des Transport Layers.

Ein interessanter Aspekt des Transport Layers ist, dass dieser aus einem *Verbindungslosen Dienst eines Paketvermittelndes Netzes*, einen *Verbindungsorientierten Dienst* bereitstellen kann.

Es kommen hier wieder die Konzepte Flusskontrolle, Punkt-zu-Punkt Verbindung, etc. zum Einsatz.

OSI Layer 3

OSI Layer 3 wird auch **Network Layer** oder **Vermittlungsschicht** genannt und beschreibt wie die Daten den Weg durch ein Netz mit mehreren Knoten findet. Je nach Routing-Schemata muss der Network Layer Datenpakete **duplizieren** und an **verschiedene Adressen** senden (Point-Multipoint Verbindung). Der Network Layer stellt je nach Konfiguration eine sog. **Verbindungslose** oder **Verbindungsorientierte** Verbindung zur Verfügung.

OSI Layer 2

OSI Layer 2 wird auch **(Data-) Link Layer** oder **Sicherungsschicht** genannt und beschreibt wie eine gesicherte Übertragungsstrecke realisiert werden kann. Die Anforderungen an den Link Layer sind davon abhängig ob genau **zwei** oder **mehrere Teilnehmer** durch das Übertragungsmedium angeschlossen sind.

2 Teilnehmer

- Multiplexing
- Fluss-Steuerung
- Fehlererkennung / Fehlerkorrektur
- Bitstuffing
- Aufteilen von OSI Layer 3 Datenpaketen in Datenrahmen / Dataframe

Und bei mehreren Teilnehmern müssen zusätzlich die folgenden Aufgaben übernommen werden

- Adressierung der Teilnehmer
- Medium Zugriffsverfahren

OSI Layer 1

OSI Layer 1 wird auch **Physical Layer** oder **Bitübertragungsschicht** genannt und beschreibt wie auf ein Medium zugegriffen wird. Dies beinhaltet insbesondere die elektrischen Eigenschaften (zB. Pegel, Frequenzen, Zeiten, Anpassung etc.), die Codierung der Daten (zB. AMI, non return to Zero etc.) und die mechanischen Eigenschaften (zB. Stecker, Pinbelegung etc.). Das Übertragungsmedium selbst ist ausserhalb des OSI Modells, wird je nachdem aber im Layer 1 vorgeschrieben.

Übersicht Layer 1-2

Wir werden uns in diesem Kurs vor allem mit den Layern 1-3 auseinander setzen und die Layer 4-7 werden wir leider nur thematisch anreissen.

Damit wir uns besser zurecht finden hier noch einen Überblick über die Layer 1-3

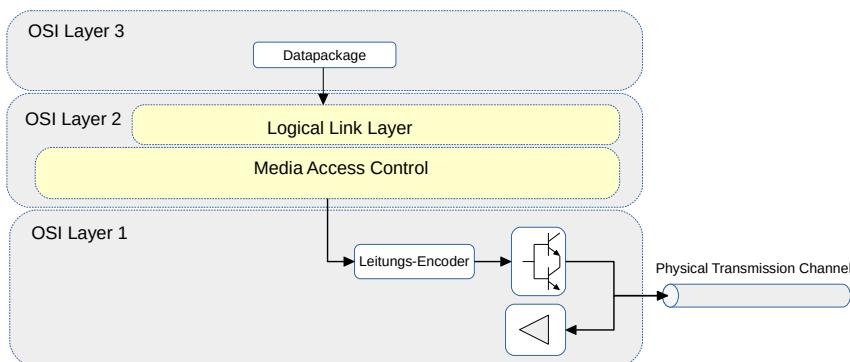


Abbildung 4.5: OSI Layer 1-3 : Simple

In Abbildung 4.5 sehen wir die Layer 1-3 abgebildet. Wir sehen, dass unser Kommunikationssystem von Layer 3 Daten an den Layer 2 übergeben werden. Dieses Daten werden als **Paket** oder **Datagram** bezeichnet. Das Layer 3 *Datagram* wird zunächst vom Layer 2 *Logic Link Layer* und danach vom *Media Access Control Block* bearbeitet. Die Daten, welche den Layer 2 verlassen nennt man **Bitstrom**. Im Layer 1 befindet sich im wesentlichen der sog. Leitungs-Encoder, welcher aus dem Bitstrom einen sog. **Leitungscode** generiert.

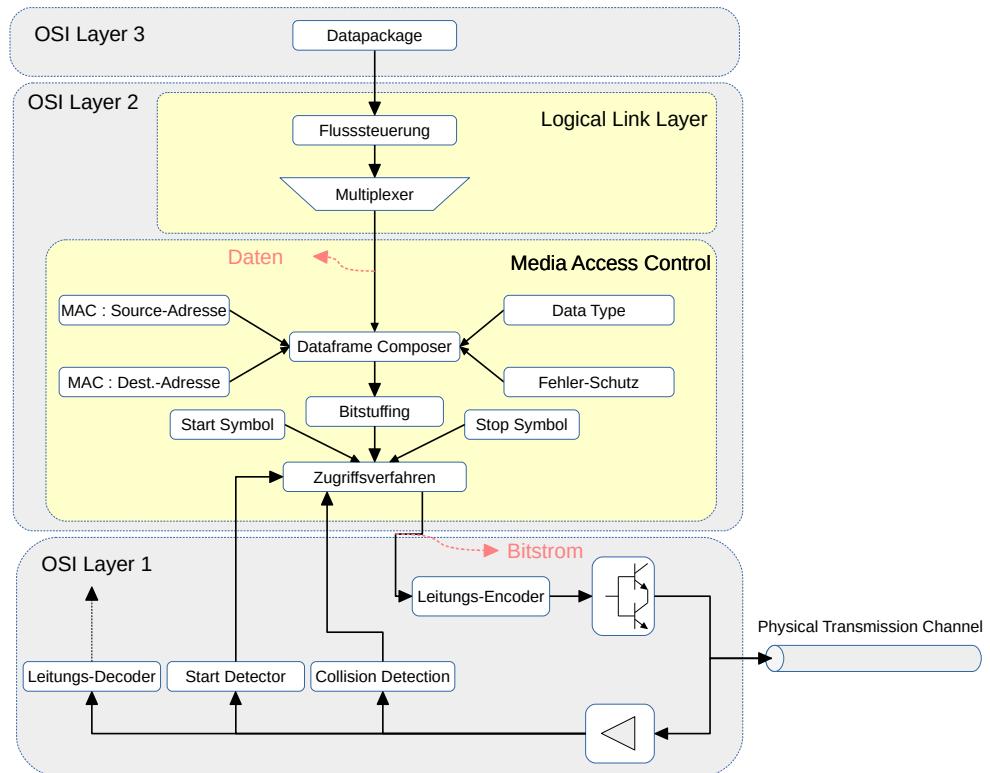


Abbildung 4.6: OSI Layer 1-3 : Tx

In Abbildung 4.6 ist der Sendeteil noch ein wenig detaillierter dargestellt.

Im **Logical Link Layer** ist die sog. **Flussteuerung** untergebracht. Diese sorgt dafür, dass verloren gegangene Datenpakete neu gesendet werden.

Der **Multiplexer** im Layer 2 kontrolliert unter anderem die Grösse der Datagramme. Nicht jede Hardware kann beliebig lange Datagramme versenden. Der Multiplexer zerstückelt gegebenenfalls die Datagramme in kleinere Datagramme und versendet diese nacheinander.

Im **Media Access Control** werden die Datagramme weiter mit Zusatzinformationen angereichert. Man nennt dieses angereicherte Datenpaket **Dataframe** oder einfach nur **Frame**. Zusatzinformationen sind zB. Die *Empfänger- und / oder Sender-Adresse* (falls mehr als zwei Teilnehmer im gleichen Netz sind), **Fehlerschutz** und Frame **Start- und / oder Stop-Symbole**.

Layer 1 enthält den **Leitungs-Encoder** und -**Decoder** und vieles anderes, welches man nicht klar vom Layer 2 trennen kann. Keine Sorge, wir werden das im Detail früh genug besprechen.

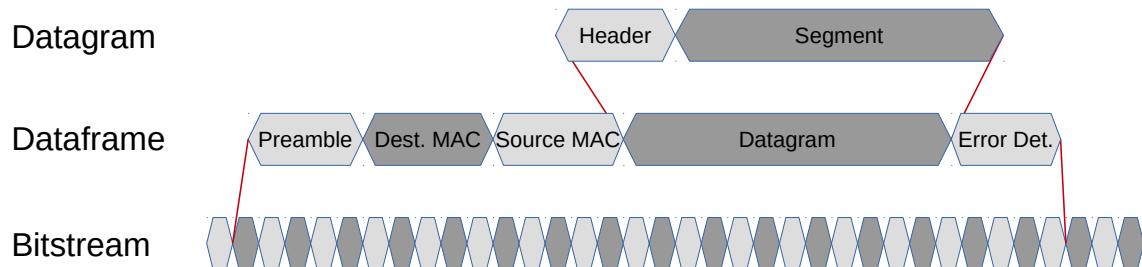


Abbildung 4.7: Verschachtelung

Damit wir das ein wenig besser verstehen: Zu unterst haben wir unseren Bitstream in Layer 1, dieser enthält das Dataframe aus Layer 2 und dieses enthält das Datagramm aus Layer 3. Und das Datagramm aus Layer 3 enthält die Segmente aus Layer 4.

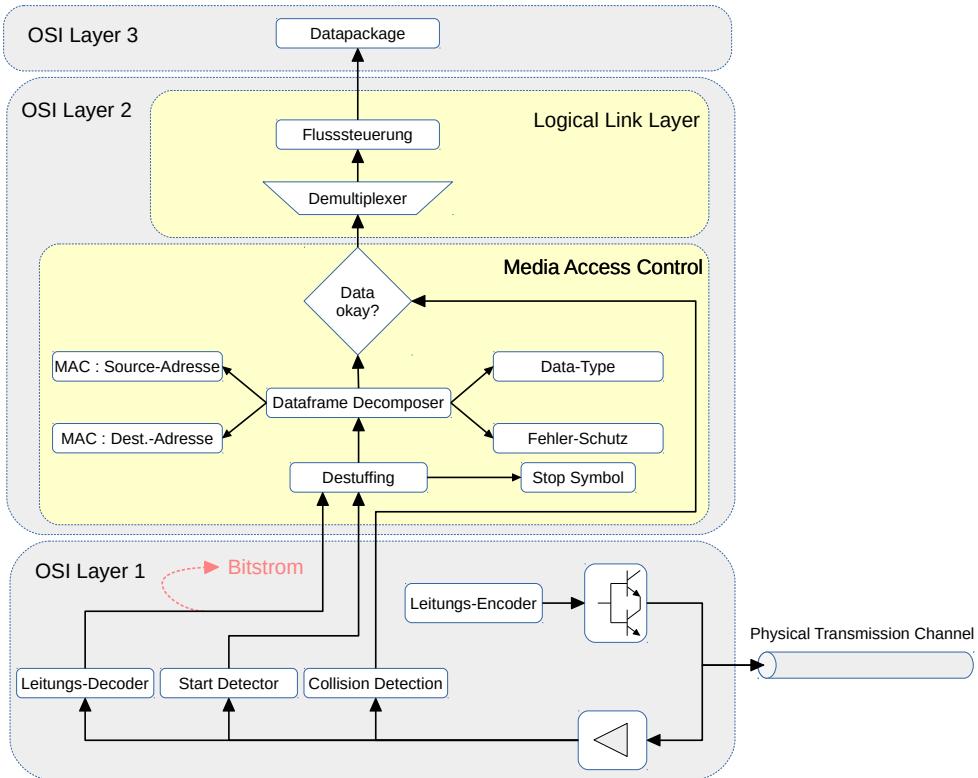


Abbildung 4.8: OSI Layer 1-3 : Rx

Auf der Empfangsseite ist sieht das ganze recht ähnlich aus. Wir sehen in *Abbildung 4.8* wieder die 3 Schichten. In Layer 1 wird aus dem Leitungscode der Bitstream decodiert und zusätzlich Informationen detektiert. Zum Beispiel ein Start, resp. Stop Symbol erkennung oder eine sog. Collision Detection. Aus dem Bitstream wird im Layer 2 das Dataframe extrahiert und in die einzelnen Bestandteile zerlegt. Ist das einzelne Dataframe in Ordnung, wird es an den Logical Link Layer übergeben. Je nach Protokoll setzt der Demultiplexer das Datagramm aus mehrere Dataframes wieder zusammen und übergibt es der Flusssteuerung. Diese sendet je nach Protokoll nun eine «Nachricht erhalten» Antwort an den Absender zurück und gibt das Datagramm dem Layer 3.

4.2.4 Protokoll Stack

Werden diverse Protokolle zusammen eingesetzt wird dieser Verbund auch als *Protokoll Stack* bezeichnet. Es muss dabei nicht das gesamte ISO/OSI Modell abgebildet sein. Damit nun zwei Teilnehmer miteinander kommunizieren können, müssen beide den gleichen Protokoll Stack implementiert haben. Die jeweils gleichen Schichten werden auch *Peer Layer* genannt. Als Beispiel für einen Protokoll Stack sei hier der berühmte TCP/IP Stack inklusive bekannter Internet Dienste genannt.

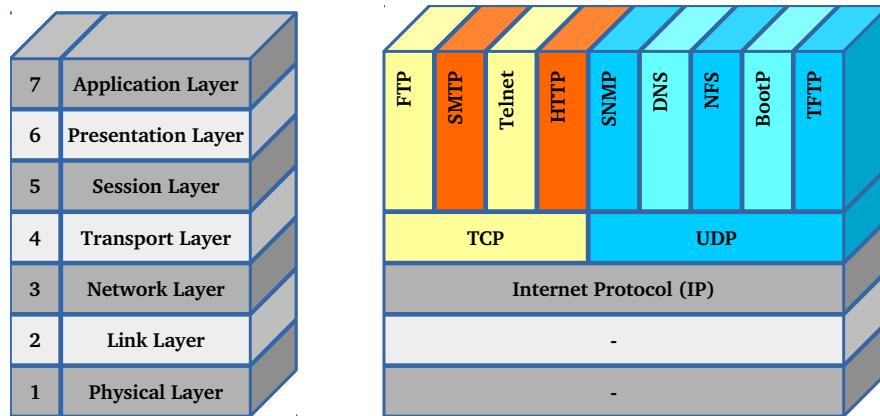


Abbildung 4.9: TCP/IP Protokoll Stack

4.2.5 Bestandteile des TCP/IP Protokoll Stacks

Protokoll	Standard	Beschreibung
Internet Protocol (IP)	IPv4 / IPv6	Addressierung der IP Teilnehmer, Routing
Transmission Control Protocol (TCP)	TCP	Verbindungsorientierter Transportdienst, Handshaking, zuverlässige Verbindung
User Datagram Protocol (UDP)	UDP	Verbindungsloser Transportdienst, kein Handshaking, keine zuverlässige Verbindung

Tabelle 4.5: Bestandteile des TCP/IP Protokoll Stacks

Protokoll	Standard	Beschreibung
File Transfer Protocol (FTP)	RFC 114 / 795 / 959 / 2228 / 2428	Dateitransfer Protokoll, seit RFC2428 mi IPv6 Unterstützung
Simple Mail Transfer Protocol (SMTP)	RFC 821 / 5321	e-Mail Protokoll
Telnet	RFC 15 / 854	virtuelles Terminal
Hypertext Transfer Protocol (HTTP)	RFC 2068 / 2616	Protokoll des World Wide Web (www), HTTP/2 ist mit dem RFC 7540 bereits ratifiziert
Domain Name System (DNS)	RFC 882 / 883 /1034 /1035	Protokoll für die Umwandlung von Namen in IP-Adressen
Network File System (NFS)	NFSv2 RFC1094	Protokoll für ein verteiltes Dateisystem, NFSv3 RFC 1813, NFSv4 RFC 3010 / RFC 7530 /5661
Bootstrap Protocol (BootP)	RFC 951	Konfiguriert TCP/IP Stack zum laden eines Boot-Image über das Netzwerk, RFC 3942 / 2132 / 1542 /1534 /1533 /1532 /1497 /1395 /1084 /1048
Trivial File Transfer Protocol (TFTP)	RFC 1350 / 2347 / 2348	Filetransfer Protokoll (ähnlich zu FTP) aber verbindungsloser Dienst

Tabelle 4.6: diverse Internet Protokolle

4.2.6 Beispiel Schichtenmodell

Nehmen wir als Beispiel ein Laptop. Dieses besitzt meistens eine LAN- und eine WLAN-Netzwerkkarte. Aber wenn wir unser Laptop verwenden, ist es unseren Programmen egal, ob wir über die LAN oder die WLAN Netzwerkkarte kommunizieren. Sehen wir uns hierzu den TCP/IP Stack in Verbindung mit 100 MBit/s Ethernet an. (LAN)

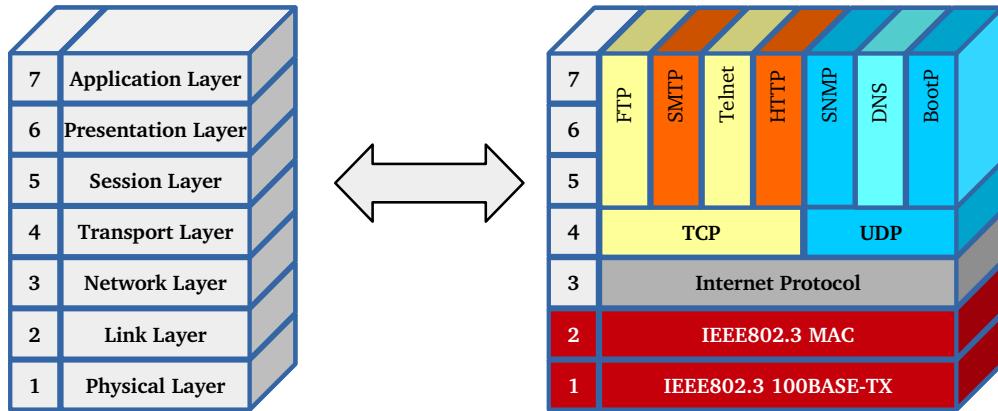


Abbildung 4.10: Beispiel Schichtenmodell

In Abbildung 4.10 ist der TCP/IP Stack in Verbindung mit 100 MBit/s Ethernet abgebildet. Man erkennt, dass Layer 3-7 unverändert, wie in Abbildung 4.9 abgebildet wurde. Einzig der OSI Layer 1/2 sind speziell auf 100 MBit/s Ethernet angepasst.

Wenn wir nun anstatt der LAN Verbindung, lieber die WLAN Verbindung verwenden wollen, müssen wir die Schichten 1/2 durch IEEE802.11 WLAN spezifische OSI Layer ersetzen.

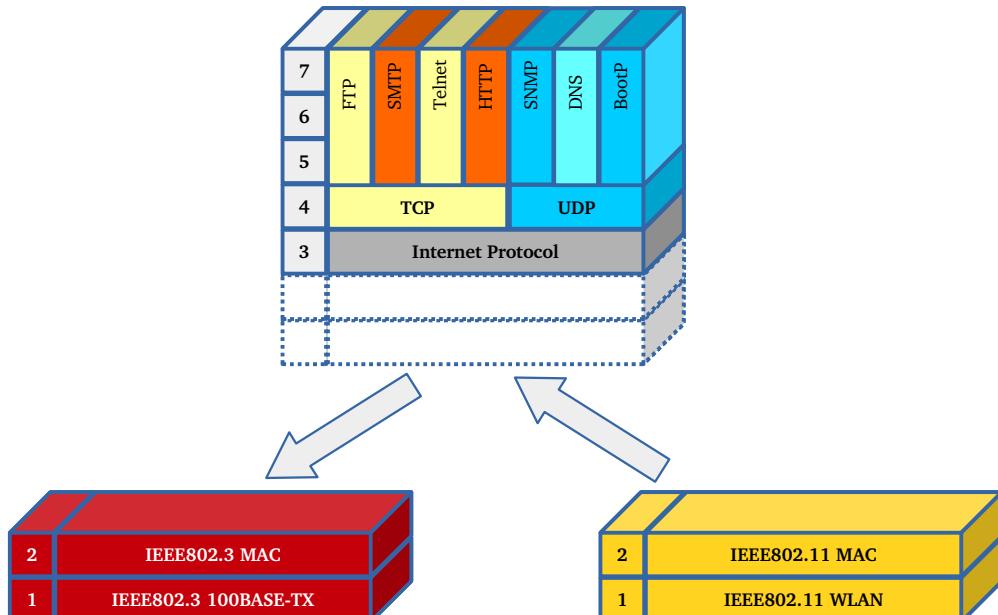


Abbildung 4.11: Austausch von Schichten im OSI Layer Modell

Damit wir eine oder mehrere Schichten austauschen können, müssen die **Interfaces** und das **Protokoll** der jeweiligen Schichten identisch sein!

Aufgabe 4.30 : ISO/OSI Modell

Frage

- Wie viele Schichten besitzt das ISO / OSI Modell?
- Beschreiben Sie die Vorteile des Schichtenaufbaus im ISO / OSI Modell
- Beschreiben Sie die Nachteile des Schichtenaufbaus im ISO / OSI Modell

Antwort

- 7
- Unabhängige Abtraktionsschichten, klar definiertes Interface, Austauschbar
- Viel Headerinformationen, Overhead nimmt mit Anzahl Schichten zu

Aufgabe 4.31 : Protokoll Stack

Frage

Wenn wir bei einem vorhandenen Protokoll-Stack eine Schicht austauschen wollen, was muss in den beiden (zu Ersetzende- und Ersatz-) Schichten identisch sein?

Antwort

Die Ersatzschicht muss die gleichen Interfaces und das gleiche Protokoll bereit stellen, damit man die Schichten gegeneinander austauschen kann.

4.3 Zusammenfassung

Die Wahl für ein Schichtenmodell hat sich bewährt. Auch die heutigen Kommunikationssysteme sind nach einem Schichtenmodell aufgebaut.

Wie wir gesehen haben, hilft uns das Schichtenmodell, damit wir einzelne Teile eines Kommunikationssystems austauschen können. Für die darüberliegenden Schichten ist es egal was für eine Schicht darunter liegt, solange das Interface identisch bleibt.

Wenn Sie in Zukunft mit einer SPS oder einem anderen Automatisierungssystem arbeiten, werden Sie sehen, das es für sie einfacher wird, wenn Sie nach dem Schichtenmodell programmieren.

Praxis

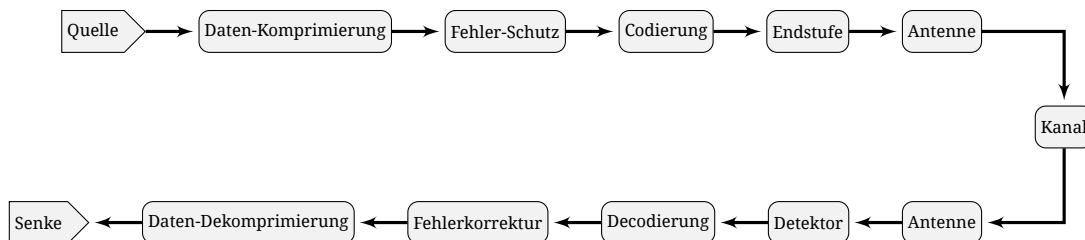
In der Praxis lässt sich das OSI Modell nicht exakt abbilden.
Moderne Kommunikations-Chips bilden einen Teil bereits in Hardware ab.
(zB. Fehlererkennung, Addressierung, Flusssteuerung)
Meist lässt sich dadurch der Layer 1,2 manchmal auch Layer 3 nicht trennen.
Aber wir müssen trotzdem wissen, wie diese funktionieren, damit wir diese richtig einsetzen können

5

Kanal / Leiter

Inhalt des Kapitel

5.1	Übersicht	73
5.2	Aufbau eines Leitungssystems	75
5.2.1	Dämpfung	75
5.2.2	Bandbreite	77
5.3	Skin-Effekt	77
5.4	Allgemeines Zweidrahtsystem	79
5.4.1	Zusammenfassung	81
5.5	Twisted Pair	81
5.5.1	Übersprechen / Cross-Talk	82
5.5.2	symmetrische Übertragung	82
5.5.3	Induktive Paar-Kopplung	83
5.5.4	Symmetrierer / Balun	84
	Gegenphasige Endstufe	84
	Balun : Balanced-Unbalanced Transformer	84
5.5.5	Zusammenfassung	85
5.6	Koaxial Kabel	86
5.6.1	Schirmdämpfung	87
5.6.2	Zusammenfassung	87
5.7	Lichtwellen-Leiter LWL	88
5.7.1	Monomode Faser	89
5.7.2	Multimode Faser	89
	Gradientenindexfaser	89
5.7.3	POF	89
5.7.4	Zusammenfassung	90
5.8	Ausbreitungsgeschwindigkeit	90
5.8.1	Einfluss auf die verschiedenen Kabelarten	92
5.9	Fazit	92



Zum Inhalt Eines der wichtigsten Elemente eines Kommunikationssystems ist der **Kanal**. (engl. **Channel**) In den meisten Fällen ist dies eine Kabelverbindung. Man kann aber auch eine Optische-Verbindung (mit Licht, zB. Laserdiode) oder eine Funk-Verbindung (mit einer elektromagnetischer Welle) aufbauen. In diesem Kapitel werden wir die verbreitesten Technologien diskutieren und wir erarbeiten uns die Formeln damit wir die für uns wichtigsten Parameter berechnen können.
Die Wireless / Funk Verbindung wird im *Kapitel 6 : Wireless / Drahtlos* abgehandelt.

5.1 Übersicht

Wir werden in diesem Kapitel die Unterschiede verschiedener Leiter kennen lernen. Damit sollen Sie lernen, wie man einen geeigneten Leiter für ein bestimmtes System auswählt und vor allem sollen Sie verstehen, was ein guter Leiter ausmacht.

Wir werden außerdem die theoretischen Grundlagen verschiedener Eigenschaften von Leitungssystemen besprechen.

Vermutlich erinnern Sie sich an das *URI-Dreieck* aus der Elektrotechnik: Wir haben das *Bitraten-Dreieck*

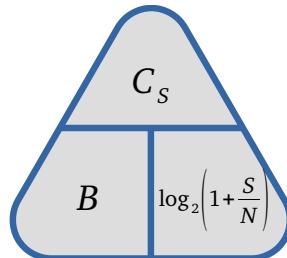


Abbildung 5.1: Bitraten-Dreieck

$$C_s = B \cdot \log_2 \left(1 + \frac{S}{N} \right) \quad (5.1)$$

In Abbildung 5.1 und Formel (5.1) sehen wir gleiche 3 neue Formelzeichen. Zum einen sehen wir das C_s , dieses bezeichnet die max. (theoretisch-) mögliche **Bitrate** eines Kommunikationssystems. (R_B bezeichnet die «erreichte» Bitrate). Als nächstes sehen wir das $\frac{S}{N}$. Dieses bezeichnet das sog. Signal-zu-Rauschleistungs Verhältnis. Erinnern wir uns an Abbildung 2.7: Wir haben dort ein Signal abgebildet, welches wir mit Rauschen überlagert haben. In Abbildung 5.2 ist das gleiche Signal mit anderer Beschriftung abgebildet. Das $\frac{S}{N}$ beschreibt nun das Verhältnis der Signalleistung (vereinfacht $U_{Signal} \approx 5 \text{ V}$) zur Rauschleistung (vereinfacht $U_{Noise} = 1 \text{ V}$). Damit wir das $\frac{S}{N}$ haben, müssen wir noch die Leistung der beiden Spannungen rechnen: (vereinfacht)

$$\frac{S}{N} = \frac{P_{Signal}}{P_{Noise}} \approx \frac{\frac{U_{Signal}^2}{R}}{\frac{U_{Noise}^2}{R}} = \frac{U_{Signal}^2}{U_{Noise}^2} \quad (5.2)$$

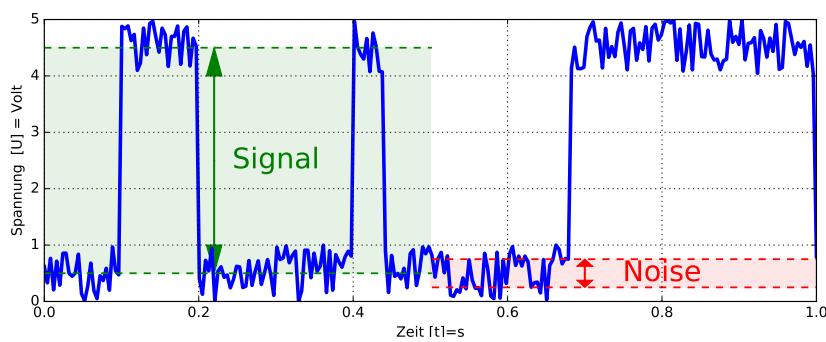


Abbildung 5.2: Signal-zu-Rauschleistungs Verhältnis

Für die Berechnung wird das Rauschen N (engl. Noise) beim Empfänger modelliert (hinzugerechnet). Daher ist es für unsere Kabel / Channel Betrachtung uninteressant.

Noch eine Anmerkung zur Abbildung 5.2: Man kann die Noise-Amplitude nicht wie abgebildet herauslesen, die Grafik ist nur zur Visualisierung.

Die Bandbreite B :

Vielleicht erinnern Sie sich an den Elektrotechnik Unterricht: Nehmen wir ein einfaches RC-Tiefpassfilter:

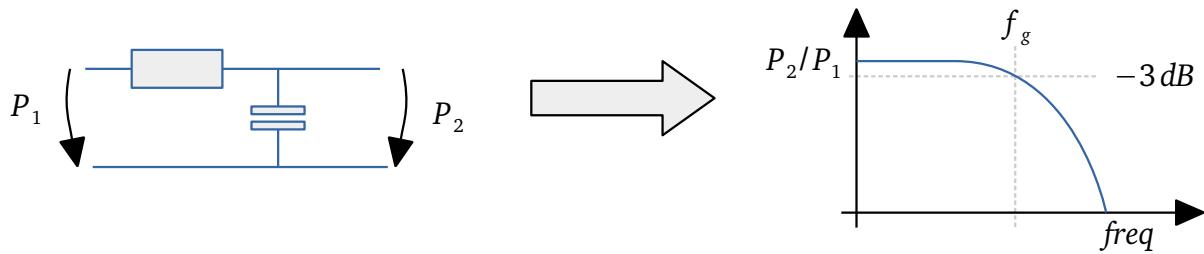


Abbildung 5.3: Frequenzgang eines einfachen RC-Tiefpass

In Abbildung 5.3 ist ein RC-Tiefpassfilter abgebildet. Ein Tiefpass-Filter lässt tiefe Frequenzen ungehindert passieren, aber bei höheren Frequenzen leitet der Kondensator immer besser und schliesst das Signal kurz. Messen wir am Ausgang nur noch die Hälfte der Energie, welche am Eingang angelegt wird, spricht man vom der -3 dB Bandbreite und nennt die Frequenz an diesem Punkt die Grenzfrequenz f_g . Damit ist die Bandbreite eines RC-Tiefpass wie folgt definiert: $B = [0, \dots, f_g]$

Lange Rede kurzer Sinn: Die maximal übertragbare Bitrate eines Kommunikationssystems ist davon abhängig, wieviel Bandbreite zur Verfügung steht und wie viel Signalleistung am Empfänger eintrifft.

Wir werden die Kabel also auf die Eigenschaften Bandbreite B und Dämpfung A analysieren. Die Dämpfung gibt an, um wie viel das Signal gedämpft wird.

Die Formel (5.1) folgt übrigens aus dem berühmten *Shannon-Hartley Theorem*.

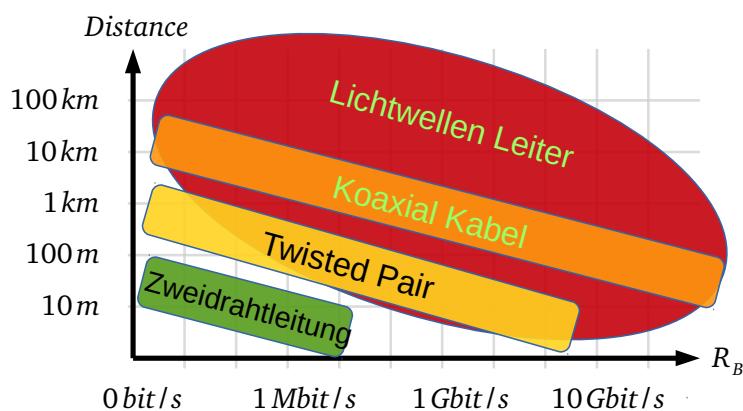


Abbildung 5.4: (Simple) Übersicht verschiedener Leitungssysteme

In Abbildung 5.4 ist eine Übersicht über die verschieden Leitungssysteme abgebildet. Wir sehen diese sortiert nach ihrem geeigneten Einsatzgebiet. Eine einfache Zweidrahtleitung eignet sich für kurze Verbindungen und kleine Datenraten. Ein Twisted-Pair Kabel ist für höherer Datenraten und grösseren Distanzen geeignet. Das Koaxial-Kabel ist das beste kabelgebundene Übertragungsmedium. Dieses wird nur noch von optischen Leitern überboten.

5.2 Aufbau eines Leitungssystems

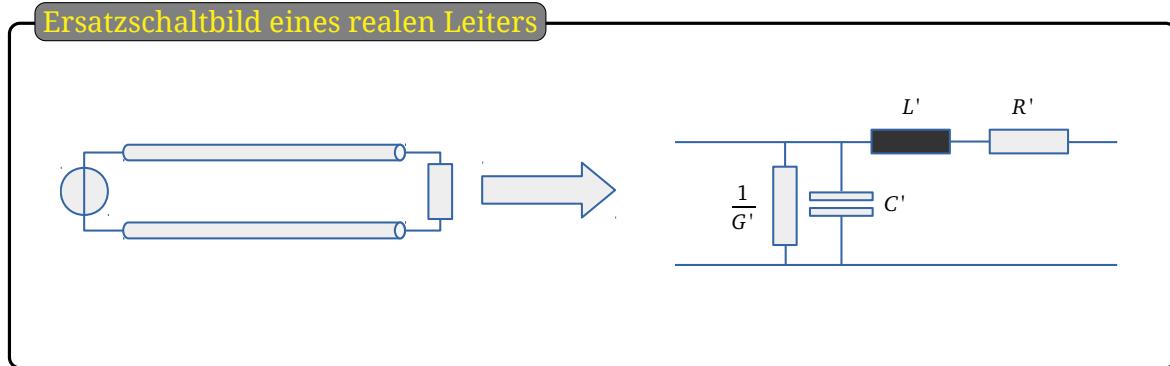


Abbildung 5.5: Ersatzschaltbild eines realen Leiters

In Abbildung 5.5 sehen wir auf der linken Seite eine schematische Darstellung einer beliebigen Schaltung mit einer Spannungsquelle und einem Lastwiderstand, verbunden mit einem **realen Leiter**. Auf der rechten Seite sehen wir den realen Leiter in seine schematischen Bestandteile zerlegt. Oben den hin Leiter, unten den Rückleiter. Wir sehen, dass der Leiter in die 3 Grundbauteile der Elektrotechnik ausgedrückt werden kann. Wir sehen, dass der Leiter zum einen **Resistiv** R' wirkt und zwar seriell zur Stromrichtung. Dies sind die **Kupferverluste** und sorgen dafür, dass sich der Leiter erwärmt. Außerdem seriell zur Stromrichtung ist eine **Induktivität** L' eingezeichnet. Stellen Sie sich das wie eine Spule mit nur einer Wicklung vor. Jeder von Strom durchflossene Leiter erzeugt ein Magnetfeld um sich und hat somit einen *parasitären Induktiven-Anteil*. Dieser alleine ist nicht all zu schlimm, aber mit dem *parasitären Kapazitiven-Anteil*, welcher durch das Elektrische Feld zwischen Hin- und Rückleiter erzeugt wird, bildet sich ein Tiefpass Filter, welches die Bandbreite begrenzt. Dieser **Kapazitive** Anteil C' ist vor allem durch das Dielektrikum der Leiterisolation und dem Abstand der Kabel zueinander definiert. Zu Letzt sehen wir noch den parallel geführten **Leitwert** G' . Dieser steht für den Resistiven-Anteil der Isolation der beiden Leiter. Meistens ist die Isolation so gross, dass wir $G' \gg 1 \text{ M}\Omega$ vernachlässigen können.

5.2.1 Dämpfung

Die Dämpfung ist folgendermassen definiert:

$$A = \frac{P_{in}}{P_{out}} \quad [A] = 1$$

$$A_{dB} = 10 \log_{10} (A) = 10 \log_{10} \left(\frac{P_{in}}{P_{out}} \right) \quad [A_{dB}] = dB \quad (5.3)$$

Die Dämpfung ist also der Leistungsverlust unserer Leitung. Im Datenblatt eines Kabels steht die Dämpfung meist als dB/m also Dämpfung pro Meter. Der Grund ist einfach : Die Kabelhersteller wollen für jeden Typ Kabel nur ein Datenblatt schreiben. Sie müssen die Dämpfung also immer selbst auf ihre Kabellänge ausrechnen.

Was ist aber der Grund für die Dämpfung? Betrachten wir nochmals die Abbildung 5.5. Die Induktivität L' und die Kapazität C' bilden zusammen einen Tiefpass-Filter. Solange wir unser Signal innerhalb der Bandbreite des LC-Filters ist, erfährt das Signal nur eine vernachlässigbare kleine Dämpfung von Filter. Der Parallel-Leitwert G' ist meist vernachlässigbar klein, also bleibt noch der Serie-Widerstand R' . Der Serie-Widerstand R' hat nun eine interessante Eigenschaft. Er ist über die Frequenz **nicht konstant!** Der sog. Skin-Effekt sorgt dafür, dass im Leitermaterial der Strom mit zunehmender Frequenz nur noch auf der Oberfläche des Leiters fliesst.

Wir werden im Kapitel 5.3 : Skin-Effekt genauer betrachten.

Nice to Know

Praxis-Tipp : Die Dämpfung misst man mit einem Netzwerkanalyzer oder extrapoliert die Dämpfung aus den Datenblatt Parametern

Dämpfung pro Meter Sehen wir uns noch die Berechnung für die Dämpfung pro Meter genauer an. Wir kaskadieren hierzu einzelne Leitungsstücke:

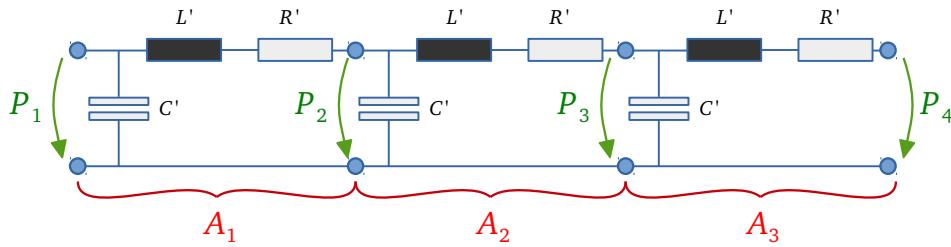


Abbildung 5.6: Kaskadieren einzelner Leitungsstücke

Wir sehen in Abbildung 5.6 ein vereinfachtes Leitungsmodell. Wir ignorieren für die nächste Betrachtung die parasitären Induktivitäten und Kapazitäten.

Wenn wir nun diese 3 Leitungsteile zusammenschliessen, müssen wir die Dämpfungen multiplizieren:

$$\begin{aligned} A_{tot} &= \frac{P_1}{P_4} = \frac{P_1}{P_2} \cdot \frac{P_2}{P_3} \cdot \frac{P_3}{P_4} = A_1 \cdot A_2 \cdot A_3 \\ A_{dB\ tot} &= 10 \log_{10} (A_1 \cdot A_2 \cdot A_3) = 10 \log_{10} (A_1) + 10 \log_{10} (A_2) + 10 \log_{10} (A_3) \quad (5.4) \\ A_{dB\ tot} &= A_{dB\ 1} + A_{dB\ 2} + A_{dB\ 3} \end{aligned}$$

Wir sehen hier den Vorteil der logarithmischen Darstellung : In der linearen Darstellung wird jedes Dämpfungsstück multipliziert.

In der logarithmischen Darstellung wird das ganze summiert. Hat man die Werte in logarithmischer Darstellung lassen sich Schaltung und Zusammenhänge einfach im Kopf ausrechnen.

Aufgabe 5.32 : Dämpfung eines kaskadierten Leitungssystems

Frage

Gegeben ist ein Leitung mit einer Dämpfung von 0.3 dB/m und ist 3.3 m lang. Wie gross ist die Dämpfung dieses Kabels? (in dB)

Antwort

$$A_{dB} = l \cdot \frac{A_{dB}}{1 \text{ m}} = 3.3 \text{ m} \cdot 0.3 \text{ dB/m} = 0.99 \text{ dB}$$

5.2.2 Bandbreite

Wie bereits mehrfach erwähnt, bilden die parasitäre Induktivität L' und die parasitäre Kapazität C' zusammen einen LC-Tiefpass Filter.

Wir wollen nun untersuchen wie sich die parasitären Eigenschaften auf das Filterverhalten eines realen Leiters auswirken.

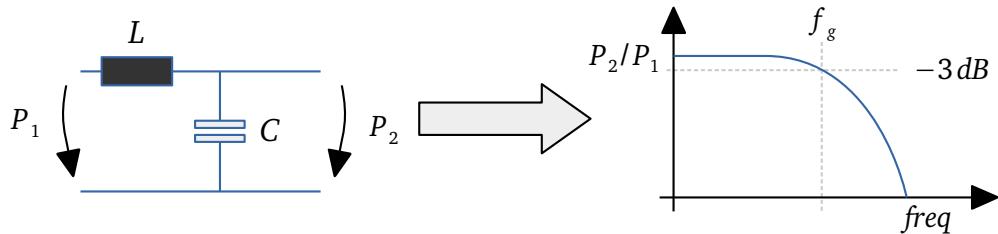


Abbildung 5.7: LC-Filter

Wir haben in *Abbildung 5.7* einen LC-Filter inkl. Frequenzgang abgebildet. Wieder ist die Kenngröße die sog. Grenzfrequenz f_g und wird folgendermassen berechnet:

$$f_g = \frac{1}{2\pi\sqrt{L \cdot C}} \quad (5.5)$$

Aus dem Shannon-Hartley Theorem (5.1) wissen wir, dass wir möglichst eine hohe Bandbreite haben möchten, damit wir bei einem Kommunikationssystem auch eine hohe Datenrate erreichen können. Damit sind wir daran interessiert, dass die Grenzfrequenz möglichst gross ist.

$$f_g = \frac{1}{2\pi\sqrt{L \cdot C}} \Rightarrow \underbrace{\sqrt{L \cdot C}}_{\text{möglichst klein}}$$

Wichtig

Aus dem Shannon-Hartley Theorem folgt noch eine interessante Eigenschaft:
Wenn wir über ein Leiter eine Baudrate B_d übertragen möchten, muss der Leiter mind. die folgende Bandbreite aufweisen:

$$B_{min} \geq \frac{B_d}{2} \quad (5.6)$$

5.3 Skin-Effekt

Der Resistive-Anteil R' ist für hochfrequente Signale leider **nicht** konstant.

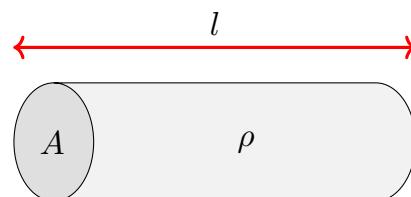


Abbildung 5.8: sehr einfaches Modell eines Ohmschen Widerstandes

Bei hohen Frequenzen verhält sich der Strom ein wenig merkwürdig. Der Strom fängt an, nur noch an der Oberfläche zu fließen. Das Zentrum des Leiters wird zunehmend Stromlos, während die Hülle eine immer höhere Stromdichte aufweist. Grund hierfür sind Wirbelströme, verursacht durch den schnell

ändernden Wechselstrom im Leiter. Die Stromdichte nimmt im Leiter gegen das Zentrum exponentiell ab. Dieser Effekt nennt sich Skin-Effekt [W64].

$$\delta = \sqrt{\frac{2\rho}{2\pi \cdot f \cdot \mu}} \quad [\delta] = m \quad (5.7)$$

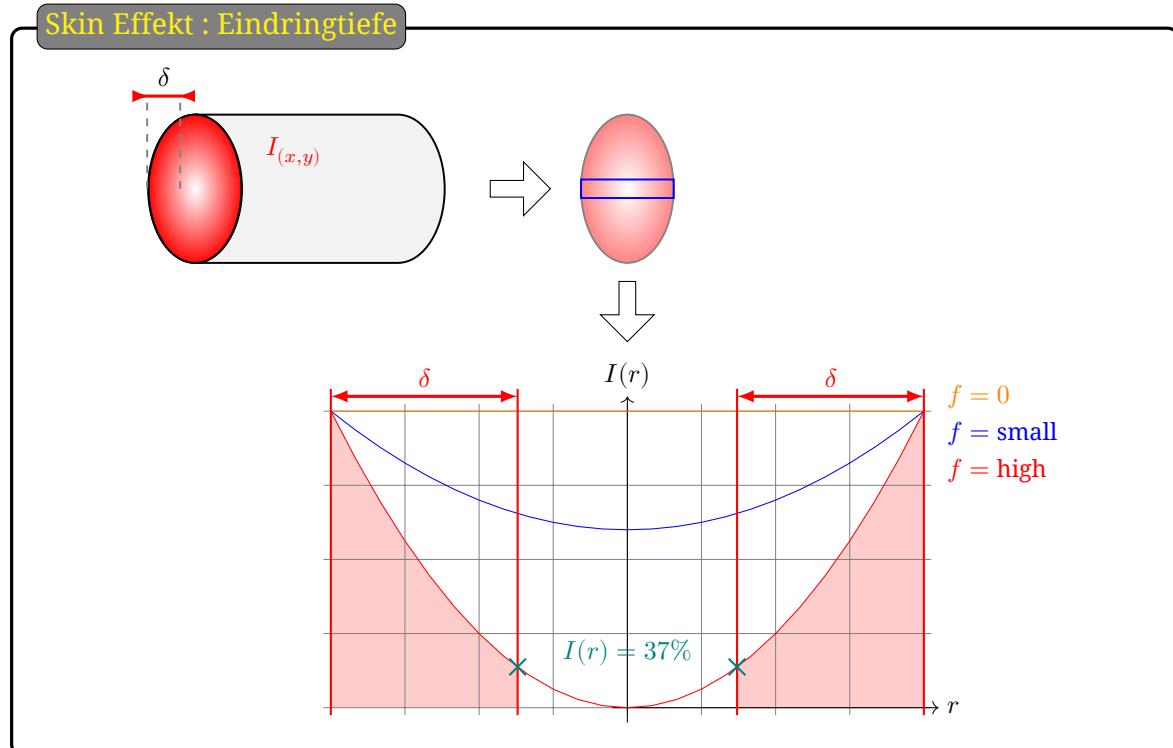


Abbildung 5.9: Skin Effekt : Eindringtiefe

Der Faktor δ heisst «Eindringtiefe», hier fliesst noch ca. $e^{-1} \approx 37\%$ des Stromes. Die Formel stimmt allerdings nur für runde Leiter, deren Durchmesser erheblich grösser ist als die Eindringtiefe. Ich möchte hier nicht weiter ins Detail gehen. Merken sollten Sie sich aber, dass bei hohen Frequenzen zusätzlich der Skin-Effekt wirkt und dem Ohmschen Widerstand die Wirkfläche reduziert und somit weitere Verluste entstehen.

Grösse	Formelzeichen	Einheit
Eindringtiefe	δ	m
Leiterdurchmesser	d	m
Frequenz	f	Hz
Kreiszahl	π	1
elektrische Leitfähigkeit	ρ	$\Omega \frac{m^2}{m}$
Permeabilität	μ	$\frac{H}{m} = \frac{Vs}{Am}$

Tabelle 5.1: Formel-Zeichen / -Einheiten : Skin Effekt

Hochfrequenzkabel sind meist mit Litzen aufgebaut. Die Summe der Oberfläche der einzelnen Adern ist um ein Vielfaches höher, als ein einziger dicker Draht und somit ist die Dämpfung durch den Skin-Effect geringer. In der Technik hat der Skin-Effect aber auch einen Nutzen. In modernen Aluminiumverarbeitungsanlagen wird der Skin-Effekt dazu genutzt, Aluminiumteile zu verschweißen. Aluminium

schmilzt beim Autogen-schweissen einfach in einem Stück weg. Mit dem Skin-Effekt können Sie punktuell die Oberfläche des Materials erhitzen und verschweissen.

Interessant ist ausserdem, dass Leiter für sehr hohe Frequenzen meist ausgehöhlt sind. Siehe Abbildung 5.19 Da im Zentrum sowieso kein Strom fliesst spart man sich hier unnötige Materialkosten und Gewicht.

Aufgabe 5.33 : Skin Effekt

Frage

Berechnen Sie die Eindringtiefe eines Kabels mit einem Radius von $r = 0.35 \text{ mm}$, Material Kupfer $\rho = 17 \cdot 10^{-3} \Omega \frac{\text{mm}^2}{\text{m}}$, $\mu_r \approx 1$ bei einer Frequenz von $f = 170 \text{ MHz}$

Antwort

$$\begin{aligned}\delta &= \sqrt{\frac{2 \cdot \rho}{2 \cdot \pi \cdot f \cdot \mu}} = \sqrt{\frac{2 \cdot 17 \cdot 10^{-9}}{2 \cdot \pi \cdot 170 \cdot 10^6 \cdot 4\pi \cdot 10^{-7}}} \\ &= \sqrt{\frac{17}{4 \cdot 170 \cdot 10^8 \pi^2}} = \frac{1}{2\pi} \cdot \sqrt{\frac{1}{10^9}} = 5.03 \mu\text{m}\end{aligned}$$

5.4 Allgemeines Zweidrahtsystem

Der einfachste Fall ist nun ein Zweitdrahtsystem ohne irgendeine spezielle Leiterführung und Dielektrikum.

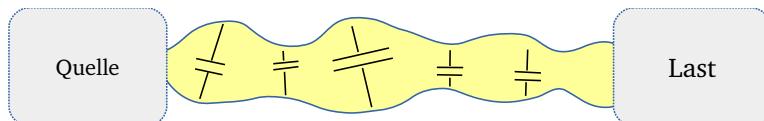


Abbildung 5.10: Allgemeines Zweidrahtleitung

Wie in Abbildung 5.10 dargestellt, ist die parasitäre Kapazität stark durch den Leitungsverlauf abhängig. Somit wird das Abschätzen des Frequenzganges der Leitung massiv erschwert.

Nehmen wir als Annäherung an, dass es sich um stückweise Plattenkondensatoren handelt, dann können wir die einzelnen parallel Kapazitäten über die Formel (2.3) abschätzen.

Seriell wirkt neben dem Ohmschen Widerstand auch die Serie-Induktivität eines stückweit parallel geführten Leiters hinzu (2.5).

Wir sehen, dass die kapazitiven Anteile nur durch den Abstand des Leiters dominiert werden. Ein weiterer Punkt ist die Störanfälligkeit. Störungen des Elektrischen Feldes (zB. Durch Funk, oder eines benachbarten Leiterpaars) werden über die aufgespannte Fläche des Leiterpaars aufgefangen (Abbildung 5.10 Gelb eingezeichnet). Einfache Zweidrahtsysteme werden in der Kommunikationstechnik höchstens für einen fliegenden (temporären) Testaufbau gebraucht oder wenn der Preisdruck enorm und die Anforderung extrem niedrig sind. Ansonsten ist ein solcher Aufbau nicht empfehlenswert.

Aufgabe 5.34 : allg. Zweidrahtsystem

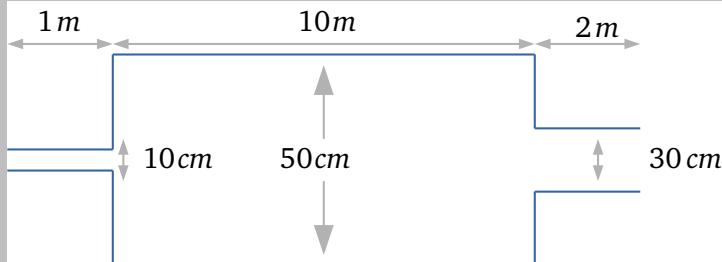
Frage

Schätzen Sie die Kapazität und die Induktivität der Kupferleitung mit dem Durchmesser

$2r = 0.7 \text{ mm}$, dem Dielektrikum Luft und dem folgenden Leitungsverlauf ab:

(Schätzen heisst: Mit einer Schätzformel berechnen!)

$$\mu_r \text{ Kupfer} \approx 1, \mu_r \text{ Luft} \approx 1$$



Antwort

$$C_x = \frac{\epsilon \cdot A}{d} = \frac{\epsilon_0 \cdot \epsilon_r \cdot A}{d} = \frac{\epsilon_0 \cdot \epsilon_r \cdot 2 \cdot r \cdot l}{d}$$

$$C_1 = \frac{\epsilon_0 \cdot 1 \cdot 0.7 \cdot 10^{-3} \cdot 1}{100 \cdot 10^{-3}} = \frac{0.7 \cdot \epsilon_0}{100} = 0.062 \text{ pF}$$

$$C_2 = \frac{\epsilon_0 \cdot 1 \cdot 0.7 \cdot 10^{-3} \cdot 10}{500 \cdot 10^{-3}} = \frac{7 \cdot \epsilon_0}{300} = 0.124 \text{ pF}$$

$$C_3 = \frac{\epsilon_0 \cdot 1 \cdot 0.7 \cdot 10^{-3} \cdot 2}{300 \cdot 10^{-3}} = \frac{1.4 \cdot \epsilon_0}{500} = 0.041 \text{ pF}$$

$$L_x = \frac{\mu}{\pi} \operatorname{arccosh} \left(\frac{d}{2 \cdot r} \right) \cdot l \\ = \frac{\mu_0 \cdot \mu_r}{\pi} \operatorname{arccosh} \left(\frac{d}{2 \cdot r} \right) \cdot l$$

$$L_1 = \frac{\mu_0}{\pi} \operatorname{arccosh} \left(\frac{100}{0.7} \right) \cdot 1 = 2.262 \text{ uH}$$

$$L_2 = \frac{\mu_0}{\pi} \operatorname{arccosh} \left(\frac{500}{0.7} \right) \cdot 10 = 29.058 \text{ uH}$$

$$L_3 = \frac{\mu_0}{\pi} \operatorname{arccosh} \left(\frac{300}{0.7} \right) \cdot 2 = 5.403 \text{ uH}$$

Was sehen wir nun aus dem Zahlenhaufen? Die Kapazität ist relativ klein. Variiert aber linear über den Abstand der Leiter zueinander und der Länge der Leiter. Die Induktivität ist ziemlich gross! Aber skaliert linear mit der Länge der Leiter und variiert nur schwach über den Abstand der Leiter zueinander.

$$C \sim \frac{1}{d} \\ L \sim \operatorname{arccosh}(d)$$

$$C \sim l \\ L \sim l$$

Aber was interessiert uns das genau?

Erinnern wir uns an Abbildung 5.5. Wir sehen im Ersatzschaltbild eine parallel Kapazität und eine serie Induktivität. Diese beiden Bauteile Formen einen (dominanten) Tiefpass-Filter. (Das serie R können wir dabei vernachlässigen, da klein)

Wir werden noch sehen, dass die ständige Kapazitätsänderung uns bei hohen Frequenzen sehr viel Probleme bereiten, aber dazu mehr im Kapitel 7: Leitungstheorie.

Aufgabe 5.35 : Filtereigenschaften eines allg. Zweidrahtsystems

Frage

Berechnen Sie die Filtergrenzfrequenzen der Teilstücke aus Aufgabe 5.34

Antwort

$$f_{gx} = \frac{1}{2\pi\sqrt{L_x \cdot C_x}}$$

$$f_{g1} = \frac{1}{2\pi\sqrt{0.062 \cdot 10^{-12} \cdot 2.262 \cdot 10^{-6}}} = 424.989 \text{ MHz}$$

$$f_{g2} = \frac{1}{2\pi\sqrt{0.124 \cdot 10^{-12} \cdot 29.058 \cdot 10^{-6}}} = 83.844 \text{ MHz}$$

$$f_{g3} = \frac{1}{2\pi\sqrt{0.041 \cdot 10^{-12} \cdot 5.403 \cdot 10^{-6}}} = 338.151 \text{ MHz}$$

Wir sehen, dass die Werte doch recht hoch sind. Die Filterwirkung ist also nicht der einzige Grund, weshalb sich ein einfaches Zweidrahtsystem nicht für eine High-Speed Schnittstelle eignet.

Aber das werden wir schon bald in *Kapitel 7 : Leistungstheorie* auflösen

5.4.1 Zusammenfassung

- Das allg. Zweidrahtsystem ist stark von der Kabelführung abhängig.
- Störungen werden über die aufgespannte Fläche zwischen den Kabeln aufgefangen
- Der Induktivitätsbelag der Leitung ändert sich, abhängig vom Abstand der Leitungen zueinander
- Der Kapazitätsbelag der Leitung ändert sich, abhängig vom Abstand der Leitungen zueinander
- Die Bandbreite des Kabelsystems ist schlecht abschätzbar und aufgrund der Kabelführung individuell

5.5 Twisted Pair

Die logische evolutionäre Entwicklung von einer allgemeinen Zweitdrahtleitung führt uns zu den sog. Twisted-Pair Kabeln (Verdrillte Kabelpaare).

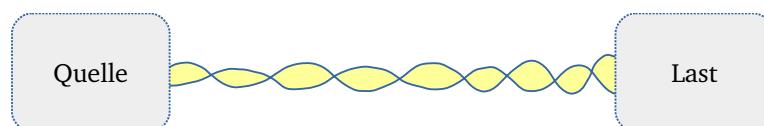


Abbildung 5.11: Twisted-Pair

Man erkennt schon leicht aus *Abbildung 5.11* das die aufgespannte Fläche zwischen den Leitern viel kleiner ist, als im Fall des allgemeinen Zweidrahtsystems. Dadurch sind Twisted Pair Kabel einiges störungsempfindlicher. Man erkauft sich dies mit höheren parasitären Kapazitäten. Aber man hat nun stabile Verhältnisse. Somit lässt sich der Frequenzgang bestimmen und die Kabel nach Normen spezifizieren. Es gibt verschiedene Variationen. Die einen sind wirklich nur gedrillte Kabel, andere sind mit Litzen gefertigt, wieder andere haben eine Schirmung mit dünner Folie oder haben ein Stahlgeflecht als Schirmung. Je nach Anwendung. Twisted Pair Kabel werden heute häufig eingesetzt. Da sie billig und recht störungsempfindlich sind. Man findet sie zB. hier:

- USB
- Ethernet 100 Mbit/s / 1 Gbit/s / 10 Gbit/s
- RS485 / Profibus / CAN
- Telefonleitung

In einem einzigen Kabel sind meist mehrere Twisted-Pair Paare zusammen verbaut. Für Twisted-Pair Kabel ist mit der Norm ISO/IEC-11801 ein Bezeichnungsschema der Form XX/YZZ entworfen worden:

	U	Ungeschirmt
Gesamtschirmung : XX	F	Folienschirmung
	S	Geflechtschirmung
	SF	Geflecht- und Folienschirmung
Aderpaarschirmung : Y	U	Ungeschirmt
	F	Folienschirmung
	S	Geflechtschirmung
Twisted-Pair / Twisted-Quad : ZZ	TP	Twisted-Pair
	TQ	Twisted-Quad

Tabelle 5.2: ISO / IEC-11801 : Bezeichnungsschema

5.5.1 Übersprechen / Cross-Talk

Bei Ethernet werden Kabel eingesetzt, welche mehrere Twisted-Pair Leitungspaare aufweisen. Ein Leitungspaar ist dabei für das Senden und das andere für das Empfangen zuständig. Werden auf einem Leitungspaar Daten übertragen, so wird ein Teil über die Kabel abgestrahlt und das andere Leitungspaar empfängt diese (Stör-)Signale. Diesen Effekt nennt man **Übersprechen** oder engl. **Crosstalk**. Wie gut ein Kabel crosstalk verhindert, hängt von der Aderpaarschirmung ab.
Ein U/UTP Kabel ist schlechter als ein U/FTP. Ein S/STP beim crosstalk etwa gleich gut wie ein U/STP, aber allgemein besser gegen Störungen von Außen geschützt.

5.5.2 symmetrische Übertragung

Lässt sich die Störempfindlichkeit von Twisted-Pair Kabeln weiter verbessern?
Ja, das ist sogar recht einfach möglich. Durch die starke Kopplung der beiden Leiter in einem Twisted-Pair Verbund lohnt es sich, diese Signale symmetrisch, also Gegenphasig anzusteuern. Störungen wirken bei so stark gekoppelten Leiter gemeinsam auf beide Leiter nahezu gleich ein.

Figure 7. Single-ended Input Retains Signal Noise

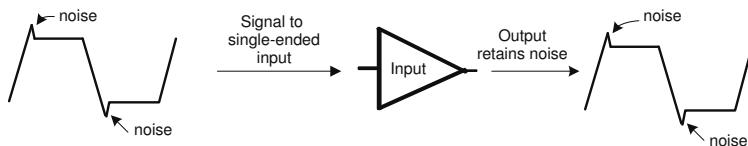


Figure 8. Differential Input Eliminates Common-Mode Noise

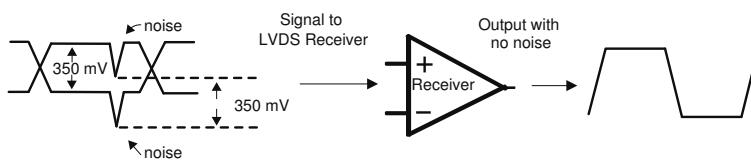


Abbildung 5.12: Auszug aus Application Note Lattice : TN1752 : (Seite 7)
http://www.latticesemi.com/view_document?document_id=47960

Wir sehen in Abbildung 5.12 einen Auszug aus einem «Application Note» von einem Bauteil der Firma Lattice. In diesem Application Note werden die Vorteile einer symmetrischen Übertragung (auch *differential transmission* genannt) vorgestellt.

Wie in Figure 8 abgebildet, kann eine Störung welche auf beide Twisted-Pair Leiter gleichzeitig einwirkt, beim Empfänger wieder herausgerechnet werden. Eine symmetrische Übertragung ist somit Störungsresistenter als eine einfache Übertragung (engl. *single-ended*) mit nur einem Signalleiter.

Aber wir brauchen für eine differential Übertragung immer 1 Leiterpaar

(⇒ 2 Adern für 1 Signal ⇒ kein «Common Ground»)

5.5.3 Induktive Paar-Kopplung

Wir haben gesehen, dass uns der Skin-Effekt mit zunehmender Frequenz den Leiterquerschnitt abschnürt. Außerdem haben wir gesehen, dass wenn wir ein Leiterpaar näher zueinander bringen, nimmt die Kapazität der Leitung zu und in kleinerem masse nimmt die Induktivität ab.

Wir würden nun erwarten, dass eine starke Kopplung sich eher negativ auf unser Leitungssystem auswirkt. Es gibt aber auch Vorteile eine starke Kopplung der Leiter zu haben:

Transformer

Kurze Rekapitulation des Transformators [W67]:

Ein Transformator ist ein Leitungssystem mit zwei isolierten Leitungsschleifen, welche stark induktiv gekoppelt sind. Normalerweise ist diese Kopplung mit einem Eisen-Material realisiert, welches ein sehr hohes ϵ_r aufweist.

Ein Transformator «transformiert» Leistung von der Primär- zur Sekundärseite. Die Wicklungszahl der Leiterschleifen definiert dabei das Strom / Spannung-Verhältnis der Ausgangsspannung.

Wichtig dabei : Der Transformator funktioniert **nur mit Wechselsignalen!**

Wenn wir die Primärseite mit einem Wechselsignal anregen erzeugt uns der durchfliessenden Strom der Primärleitung eine Änderung des magnetischen Fluss. Die Änderung des magn. Fluss führt zu einem Gegenfeld, welches der Veränderung des magn. Fluss entgegenwirkt. Das Gegenfeld wiederum führt zu einem Strom in der Sekundärleitung.

Wichtig dabei : **Es wird nur ein Gegenfeld erzeugt, wenn sich der magn. Fluss ändert!**

Info 14: Rekapitulation Transformator

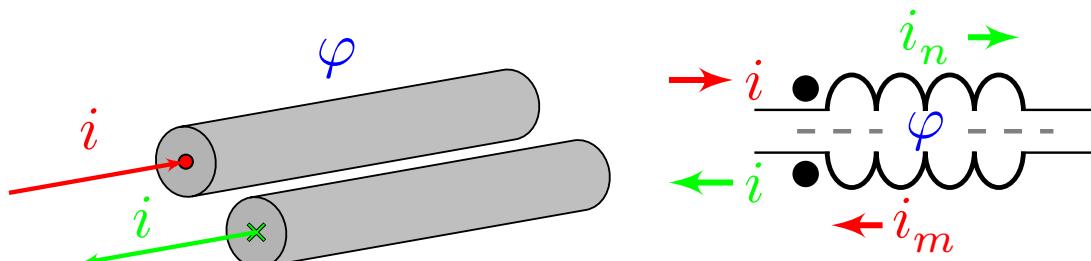


Abbildung 5.13

Wie sie sich vielleicht schon denken : Ein eng gekoppeltes Leitungspaar funktioniert wie ein Transformator. Wir machen nun ein Gedankenexperiment : Wir argumentieren mit der Superposition.

Wir betrachten nun den hineinfliessenden Strom und den herausfliessenden Strom separat.

Wenn wir ein Leitungspaar mit einem Wechselsignal anregen, erzeugt der hineinfliessende Strom i eine magnetische Flussänderung ϕ . Die magn. Flussänderung ϕ führt wiederrum zu einem Strom i_m im parallel liegenden Leiter. Dieser verläuft in der entgegengesetzten Richtung wie der hineinfliessende Strom. Das gleiche gilt nun auch für den Strom i im zweiten Leiter. Dieser erzeugt uns einen entgegengesetzten Strom i_n im ersten Leiter. Nach der Superposition können wir nun alle diese Ströme überlagern und addieren.

Daraus folgt nun folgender Effekt : Wenn wir ein Leitungspaar mit gegenphasigen Wechselsignalen anregen, dann unterstützt die induktive Kopplung den Stromfluss im Leitungspaar.

Wichtig

Der (Wechselspannungs-) Widerstand im Leitungspaar wird durch die induktive Kopplung reduziert. Die induktive Paar-Kopplung **reduziert** uns sogar die negativen Effekte des **Skin-Effekts** und **reduziert** uns somit die **Dämpfung** des (gesamt) Leiters!

Info 15: Induktive Paar-Kopplung

5.5.4 Symmetrierer / Balun

Wir haben verschiedene Varianten, wie wir ein symmetrisches Signal erzeugen können.

Gegenphasige Endstufe

Wir können dies über eine gegenphasige Endstufe realisieren:

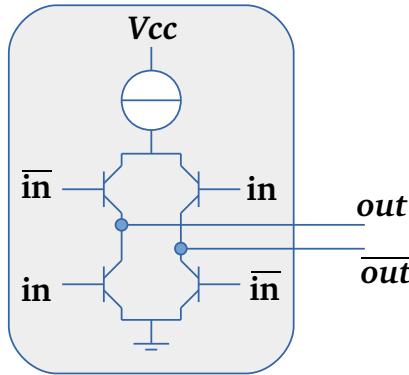


Abbildung 5.14: LVDS (Differenzielle Stromgekoppelte Endstufe)

Das LVDS Signal hat noch immer einen Massen-Bezug. Es muss daher mind. eine zusätzliche Masseleitung mit dem LVDS Signal mitgeführt werden.

Balun : Balanced-Unbalanced Transformer

Die einfachste passive Variante wie man dies Effekt in den Griff bekommt, ist ein sog. Balun (**Balanced-Unbalanced**) oder auch Symmetrierer genannt. Dies ist ein Transformator, es gibt verschiedene Bauformen, diese zwei sind die häufigsten:

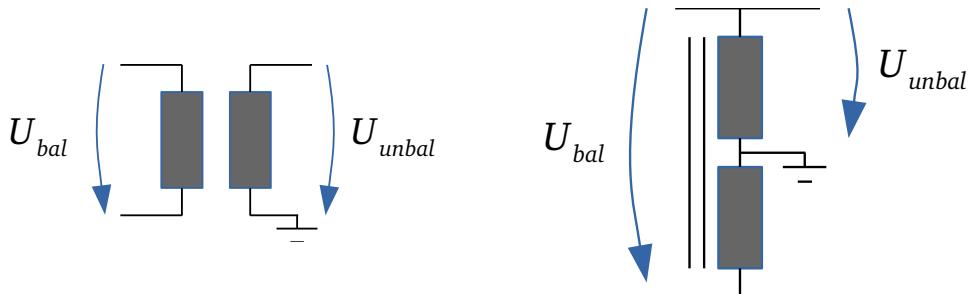
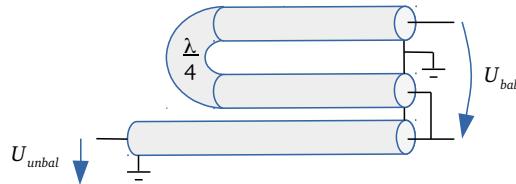


Abbildung 5.15: Balun

Passive Baluns werden vor allem bei Ethernet eingesetzt. Dies zum einen um das Signal zu symmetrieren, aber auch für eine galvanische Isolation. (Berührungsschutz) Sie sind reziprok, können also auch für eine Transformation **Unbalanced to Balanced** durchführen. Allerdings sind diese Bauartbedingt in der Bandbreite begrenzt.

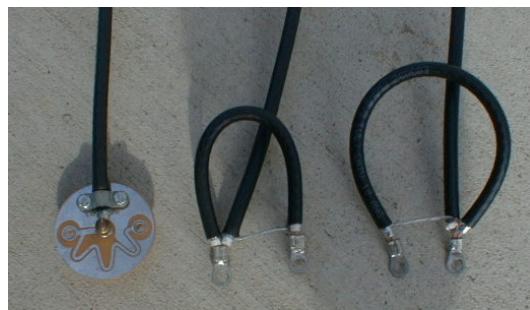
Abbildung 5.16: $\frac{\lambda}{4}$ Transformer (quarterwave transformer)

Bei höheren Frequenzen wird in der Regel mit sog. $\frac{\lambda}{4}$ Transformatoren gearbeitet. Dies sind Koaxial-Leitungen, die aufgrund ihrer Abmessungen, hochfrequente Signale in einem relativ schmalen Frequenzbereich transformieren. (Ich möchte hier nicht ins Detail gehen, das gehört in eine Hochfrequenzvorlesung)

Mit einem (Transformator-) Balun kann außerdem der **Massebezug** vom Signal entkoppelt werden. Dies hat den Vorteil, dass wir nicht explizit eine Masseleitung mit dem Signal mitziehen müssen und wichtiger: Wir reduzieren die Kapazitive-Kopplung gegenüber Erde.

Allerdings muss der Transformator auch höhere Ausgleichsströme aushalten, da weit entfernte Sender und Empfänger, teils massive Masse-Spannungsdifferenzen aufweisen können.

Ein weiterer Nachteil ist, dass das Signal sog. Gleichspannungsfrei sein muss, ansonsten wird der Transformator in Sättigung getrieben. Aber dazu später mehr...

Abbildung 5.17: Drei $\frac{\lambda}{4}$ Transformer für unterschiedliche Frequenzbereiche
http://www.atechfabrication.com/images/coax_balun_002.jpg

5.5.5 Zusammenfassung

- Bessere Störungsempfindlichkeit gegenüber allg. Zweileiterpaar \Rightarrow wenig Wirkfläche zwischen den Aderpaaren
- Mit symmetrischen Signalen nochmals störungsempfindlicher!
- grösere parasitäre Kapazität, kleinere parasitäre Induktivität
- Da die parasitären Eigenschaften **nicht** vom Leitungsverlauf abhängig sind und variieren diese nur gering
- Die Bandbreite des Twisted-Pair Kabels lässt sich gut abschätzen
- Kabelaufbau ist normiert: ISO/IEC-11801
- Crosstalk lässt sich mit zusätzlicher Aderpaarschirmung minimieren

Nice To Know

Ethernet Phy's haben am Ausgang immer einen Transformator!
 Eigentlich zwei: Einen Balun und eine Common-Mode Choke

Info 16: Ethernet Transformator

5.6 Koaxial Kabel

Die nächste Steigerung in der Bandbreite sind Koaxial-Kabel. Das ist auch wieder ein Zweidrahtsystem, der Rückleiter ist der Schirm selbst, und der Hinleiter ist die Seele des Kabels.

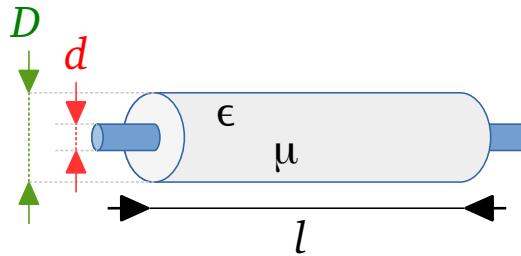


Abbildung 5.18: Koaxial Kabel

Das Koaxial-Kabel nutzt den Effekt des Faradayschen Käfigs und schirmt den Innenleiter von externen Störungen ab. Die Schirmwirkung ist nicht unendlich hoch und wird als Schirmdämpfung vom Hersteller spezifiziert. Die Symmetrie des Koaxial-Kabels, lässt uns ein paar Grundwerte recht einfach rechnen.

$$\begin{aligned} C &= l \cdot \frac{2 \cdot \pi \cdot \epsilon}{\ln\left(\frac{D}{d}\right)} \\ L &= l \cdot \frac{\mu}{2\pi} \left(\frac{D}{d}\right) \end{aligned} \quad (5.8)$$

Beachten Sie hier einmal die mathematische Symmetrie der Kapazität und Induktivität!
Als Kabel-Dielektrikum wird häufig Polyethylen, aber auch Teflon eingesetzt.

Grösse	Formelzeichen	Einheit
Kapazität	C	F
Induktivität	L	H
Länge des Kabels	l	m
Permeabilität	μ	$\frac{H}{m}$
Dielektrizität / Permittivität	ε	$\frac{F}{m}$
Innendurchmesser (Schirm)	D	m
Aussendurchmesser (Seele)	d	m

Tabelle 5.3: Formel-Zeichen / -Einheiten : Koaxial Kabel

Koaxial-Kabel sind für besonders Breitbandige Signale optimiert. Sie sind auch dafür geeignet Signale über längere Distanzen zu transportieren.

Der Nachteil der Koaxial-Kabel sind zum einen der recht hohe Preis, sowie die starre Konstruktion, so dass minimale Biegungsradien eingehalten werden müssen. Auch beim Koaxial-Kabel gibt es diverse Schirmungen und Dielektrikums, je nach Frequenz und Impedanzbereich.

Praxis-Tipp

Je flexibler das Kabel, desto kleiner die Schirmdämpfung

Koaxial-Kabel können über längere Distanzen sog. Masseschlaufen aufspannen. Das heisst, dass sich auf dem Schirm eine DC Spannung aufprägt, welche den Innenleiter beeinflusst. Bei schlecht geerdeten Fernsehkabeln konnte man früher auf dem Röhrenfernseher ein sich bewegender «Balken» im Bild beobachten.

Koaxialkabel werden vor allem im Gerätebau für Hochfrequenzanwendungen, aber auch für Fernsehkabel und früher auch für Ethernet verwendet.

Koaxial-Kabel für sehr hohe Frequenzen oder sehr hohe Leistungen haben in der Regel Luft als Dielektrikum und weisen eine Wellenform auf um die Oberfläche zu erhöhen. Siehe Kapitel 5.3 : Skin-Effekt.



Abbildung 5.19: HJ8-50B, Air Dielectric Coaxial Cable
http://www.commscope.com/catalog/imagesCache/0000023/t006_r00740_v5.jpg

5.6.1 Schirmdämpfung

Leider wird in der Literatur meist nie über die Schirmdämpfung gesprochen. Die Schirmdämpfung ist im wesentlichen ein anderer Begriff für die Störungsempfindlichkeit eines Koaxialkabels. Bei Koaxialkabel ist meist die Schirmdämpfung im Datenblatt angegeben. Ein sehr gutes Kabel erreicht Werte um die 90 dB, schlechtere Kabel eher um die 40 dB

5.6.2 Zusammenfassung

- sehr gute Schirmdämpfung durch den Effekt des Faradayschen Käfigs
- Symmetrie der parasitären Induktivität und Kapazität führen zu einem gut berechenbaren Leiter
- verschiedene Dielektriken erhältlich : auch Luft → wichtig für Hoch-Leistungssysteme
- für grosse Distanzen mit grosser Bandbreite geeignet
- Nachteil : Hoher Preis, starre Konstruktion ⇒ Biegeradius

5.7 Lichtwellen-Leiter LWL

Die Lichtwellenleiter kurz **LWL** gehören in eine völlig andere Klasse Leiter. Wie der Name schon sagt, werden anstatt elektrischer Signale, optische Signale übertragen. (Licht im Allgemeinen) Dadurch sind Lichtwellenleiter immun gegenüber elektromagnetischen Störungen. Die geringe Signaldämpfung ermöglicht die Übertragung über sehr grosse Distanzen. Außerdem sind Lichtwellenleiter sehr breitbandig und damit sehr grosse Übertragungsraten möglich. Klassischerweise wird unter den Lichtwellenleitern das sog. Glasfaserkabel verstanden. Es gibt auch sog. *Polymer optische Faser POF*, diese bestehen aus einer «Plastik-Faser».

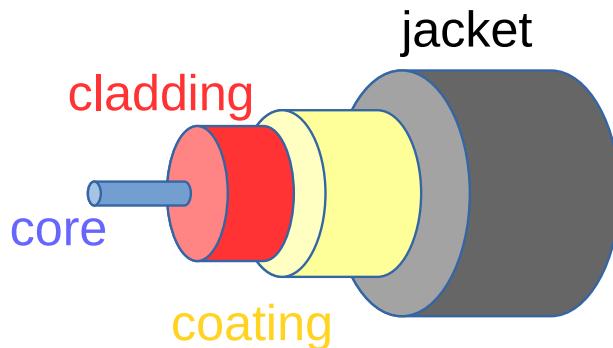


Abbildung 5.20: Aufbau eines Lichtwellenleiters

Der Lichtwellenleiter sind folgendermassen aufgebaut: Zunächst aus dem Kern (core) welcher aus einem Licht-leitenden Material besteht, in der Regel Quarzglas SiO_2 . Der Mantel (cladding) umgibt den Kern und muss einen Brechungsindex $n_{core} > n_{cladding}$ aufweisen. Dadurch wird der Lichtstrahl im Kern immer am Übergang zum Mantel reflektiert. Die Schutzhülle (coating) Schützt den Kern und den Mantel gegenüber feinen mechanischen Beschädigungen am Mantel, meist ist dies eine Art Lackierung. Die äussere Schutzhülle (jacket) Schützt vor groben Beschädigungen und sorgt für zusätzliche Steifheit des Kabels.

Wie die Koaxial-Kabel sind Lichtwellenleiter nur schlecht biegsam. Wird der spezifizierte Biegeradius nicht eingehalten, können schnell Beschädigungen am Mantel und/oder Kern entstehen, die zu einem Defekt führen.

Wichtig

Achten Sie bei Lichtwellenleiter immer auf den zulässigen Biegeradius

Bei der Fiber-to-the-home (FTTH) Distribution, werden in der Regel ganze Glasfaserbündel verlegt. Je nach Standard passen 144/288 einzelne Fasern in ein Kabel. In der Inhouse Verkabelung sind 1/12/24/48 Fasern pro Kabel Standard.

Auch beim Glasfaserkabel gibt es einen Effekt welcher der Dämpfung im Kupferkabel entspricht. Die sog. **UV-Absorption** ist ein Effekt, welcher durch die unregelmässige Anordnung im SiO_2 Gefüge hervorgerufen wird. Außerdem sind die Lichtwellenleiter anfällig für sog. **Biegeverluste**, also Verluste die durch den Leitungsverlauf beeinflusst werden.

5.7.1 Monomode Faser

Ist der Kerndurchmesser lediglich einige Vielfache der Wellenlänge des Licht(-strahls) breit, so spricht man von einer **Monomode-Faser** (engl. Single-mode fiber, SMF). Zur Definition gehören noch ein paar andere Randbedingungen, aber die lassen wir mal beiseite. (Das gehört in eine Optik-Vorlesung)



Abbildung 5.21: Monomode Faser

Die Lichtstrahlen in Monomode-Fasern werden quasi geradlinig durch die Faser geleitet.

5.7.2 Multimode Faser

Ist der Kerndurchmesser im Vergleich zur Wellenlänge des Licht(-strahls) sehr breit, so spricht man von sog. Multimode-Fasern.

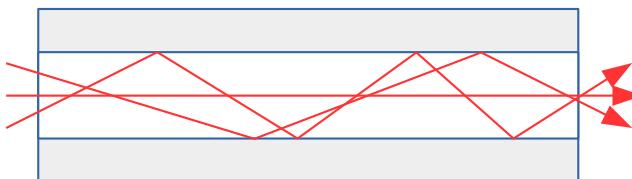


Abbildung 5.22: Multimode Faser

Die Lichtstrahlen in Multimode-Fasern werden unterschiedlich reflektiert. Das heisst, dass ein Signal (z.B. Ein Rechtecksignal) wird über die gesamte Länge verschmiert, da einige Lichtstrahlen länger unterwegs sind als andere. Dieser Effekt wird **Dispersion** genannt. Dieser begrenzt die Bandbreite und erschwert die Detektion des Empfangssignals.

Gradientenindexfaser

Um den Dispersions-Effekt in Multimode-Fasern zu reduzieren, wurden sog. Gradientenfasern entwickelt. Der Übergang vom Kern zum Mantel sind in Gradientenindexfasern verschmiert. Das heisst, dass der Brechungsindex vom Mantel nach aussen fliessend zunimmt.

Der Dispersionseffekt der Gradientenindexfaser ist aber immer noch *grösser* als einer Monomode Faser.

5.7.3 POF

Polymer optische Fasern kurz. **POF** wurden als kostengünstige Alternative zu Glasfaserkabeln entwickelt. Der Kern besteht aus einem Polymethylmethacrylat (PMMA) und der Mantel aus fluoriertem PMMA. Im Vergleich zu den Glasfaserkabel weisen sie eine weitaus höhere Dämpfung auf. Ebenfalls ist der Dispersions-Effekt stark ausgeprägt.

Dadurch eignen sich POF heutzutage nur für Entfernnungen bis ca. 100 m ... 150 m. Dafür sind sie flexibler und die Steckverbindungen sind billiger. Damit sind sie gut für die Gebäudeverkabelung geeignet. «Der Preisunterschied von POF und Glasfaser ist aber so weit gesunken, das POF wohl schon bald verschwinden wird.»

5.7.4 Zusammenfassung

- Lichtwellenleiter sind das derzeit geeignete Medium für breitbandige und weitläufige Datenverbindungen
- Lichtwellenleiter sind auch weitaus das teuerste Medium für Datenverbindungen
- POF ist eine billige Alternative für die Gebäudeverkabelung mit LWL. (Aber immer weniger gefragt ⇒ Kostenunterschied)
- Auch LWL weisen eine Dämpfung auf. Sie wird in dB/km angegeben
- Der Biegeradius von LWL ist immer einzuhalten! Die LWL sind einfach zu beschädigen!
- Lichtwellenleiter sind von Natur aus galvanisch getrennt!
- Keine Beeinflussung benachbarter Lichtleiter (Cross-Talk)
- Keine Beeinflussung von elektromagnetischen Störer (EMV)

5.8 Ausbreitungsgeschwindigkeit

Haben Sie sich jemals gefragt wie schnell sich ein Signal in einem Leiter eigentlich fortbewegt?
Um ein wenig Spannung aufzubauen : Es ist langsamer als gedacht, aber doch immer noch sehr schnell!

Nice to Know

Elektrische und auch elektromagnetische (Funk) Signale breiten sich im Vakuum mit Lichtgeschwindigkeit aus.
Überall sonst aber langsamer

In einem realen Leiter sind die Signale durch die parasitären Induktivitäten und Kapazitäten **gebremst**.

$$v = \sqrt{\frac{1}{L' \cdot C'}} \quad (5.9)$$

$$v = \frac{1}{\Delta t}$$

Für ein Koaxialkabel kann man dies gut ausrechnen. Ich gebe hier aber einfach schon die Lösung:

$$v_{coax} \approx \frac{1}{\sqrt{\mu \cdot \epsilon}} = \frac{c}{\sqrt{\mu_r \cdot \epsilon_r}} \quad (5.10)$$

Die Ausbreitungsgeschwindigkeit im Koaxialkabel ist in etwa 66% der Lichtgeschwindigkeit $v_{coax} \approx 66\% \cdot c$.

In den Datenblättern ist manchmal auch «nur» der Delay (engl. für Verzögerung) Δt in ns/m oder ns/100m angegeben.

Auch Lichtwellenleiter sind von einem Delay betroffen.

In den Datenblättern finden Sie häufig die Werte des Propagation Delay oder des Brechungsindex n_{core} des Fasermaterials. Aus dem Brechungsindex lässt sich wie folgt der Propagation Delay ausrechnen:

$$\Delta t = \frac{n}{c} \quad (5.11)$$

Grösse	Formelzeichen	Einheit
Signal-Geschwindigkeit	v	m/s
Lichtgeschwindigkeit	c	m/s
Delay	Δt	s/m
Brechungsindex	n	1

Tabelle 5.4: Formel-Zeichen / -Einheiten : Ausbreitungsgeschwindigkeit

Aufgabe 5.36 : Signallaufzeit in einem Leiter

Frage

Um unsere Kontinente mit einem Kommunikationsnetz auszustatten gibt es diverse Seekabel. Früher waren diese als Koaxialkabel ausgeführt (heute als Lichtwellenleiter) Ein solches Kabel verbindet Irland mit New-York und ist etwa $s = 5536$ km lang.

Berechnen Sie nun die Laufzeit eines Signals welches über dieses Kabel gesendet wird. Wir nehmen an es handelt sich um ein Koaxialkabel mit einem Delay von $\Delta t = 530 \frac{\text{ns}}{\text{100m}}$

Antwort

$$s = v \cdot t$$

$$t = \frac{s}{v} = \frac{s}{\frac{1}{\Delta t}} = s \cdot \Delta t$$

$$t = 5536 \cdot 10^3 \text{ m} \cdot 5.3 \cdot 10^{-9} \frac{\text{s}}{\text{m}} = 29.03 \text{ ms}$$

Aufgabe 5.37 : Signallaufzeit in einem Leiter (2)

Frage

Gleiche Distanz wie in *Aufgabe 5.36*

Nun aber mit einem Lichtwellenleiter mit einem Brechungsindex von $n = 1.49$

Antwort

$$\Delta t = \frac{n}{c}$$

$$\Delta t = \frac{1.49}{300 \cdot 10^6 \frac{\text{m}}{\text{s}}} = 5 \frac{\text{ns}}{\text{m}}$$

$$t = s \cdot \Delta t$$

$$t = 5536 \cdot 10^3 \text{ m} \cdot 5 \cdot 10^{-9} \frac{\text{s}}{\text{m}} = 27.68 \text{ ms}$$

Wie Sie sicherlich ein wenig überrascht sind : Die Signallaufzeiten sind nahezu identisch! Mein Bauchgefühl hätte erwartet dass das Licht durch die Glasfaser viel schneller unterwegs wäre.

Das Licht in der Glasfaser wird aber stark gebremst!

5.8.1 Einfluss auf die verschiedenen Kabelarten

allg. Zweidrahtsystem Da sich die parasitären Eigenschaften je nach Lage verändern, so verändert sich auch die Ausbreitungsgeschwindigkeit mit der Kabelführung

Twisted Pair Ausbreitungsgeschwindigkeit ist konstant. Nur vom eingesetzten Dielektrikum abhängig. Geringer Einfluss der Verlegungs-Lage.

Koaxial Kabel Ausbreitungsgeschwindigkeit ist konstant und je nach Material erreicht das Koaxial-Kabel die höchste Ausbreitungsgeschwindigkeit mit elektrischen Signalen

Lichtwellenleiter EM-Wellen werden vom Glasfaser-Material kaum beeinflusst, daher ist ein Lichtwellenleiter nahezu Immun gegenüber elektromagnetischen Störern. Der Lichtwellenleiter weist bei weitem die kleinste Dämpfung pro Längeneinheit auf.

5.9 Fazit

Mithilfe dieses Kapitels können Sie nun folgendes:

- Sie können die Dämpfung von einem realen Leiter aus dem Datenblatt lesen und die Dämpfung auf einen spezifisch langen Leiter extrapoliieren
- Sie kennen den Einfluss von parasitärer Induktivität und Kapazität auf ein reales Leitungssystem
- Sie kennen den Einfluss des Skin-Effekts auf die Dämpfung eines realen Leitungssystems
- Sie wissen nun, was es für Kabelarten gibt und was die Vor- und Nachteile dieser Kabelarten sind

6

Wireless / Drahtlos

Inhalt des Kapitel

6.1	Was ist Wireless?	94
6.1.1	Verhalten unterschiedlicher Trägerfrequenzen	94
6.2	Wellenarten	95
6.3	Was ist eigentlich eine Antenne?	96
6.4	Freiraum Übertragungsleistung	96
6.4.1	Freiraumdämpfung	97
6.4.2	Antennengewinn / Antenna-Gain	98
6.4.3	äquivalente isotrope Strahlungsleistung	98
6.4.4	Freiraum Sende-Distanz	99
Emfpänger-Empfindlichkeit	99	
6.4.5	Zuverlässigkeit	100
6.5	WLAN	100
6.5.1	industrial, scientific and medical frequency band	100
6.5.2	WLAN Standards	101
6.6	RFID	102
6.7	NB-IoT / M2M	102
6.7.1	LoRa	103
6.7.2	Sigfox	103
6.7.3	Weightless	103
6.7.4	Cat-M1 / Cat-NB1	103
6.7.5	Bluetooth 5: Long Range	103
6.8	Zusammenfassung	104

Zitat:

I just wondered how things were put together

Ich habe mich nur gefragt, wie die Dinge zusammen gesetzt sind

Claude Shannon : Mathematiker / Elektroingenieur

Zum Inhalt In diesem Kapitel werden wir tiefer in die Materie der Drahtlos-Kommunikation eingehen. Je länger je mehr, werden Daten drahtlos übermittelt. Es ist daher sinnvoll die Grenzen der Drahtlos-Kommunikation zu kennen und zu begreifen, dass die Drahtlos-Kommunikation mit vielen Problemen umzugehen hat.

Eine Drahtlos-Kommunikation muss mit unzuverlässigen Datenverbindungen, Nationalen Bestimmungen, physikalischen Phänomenen und schlussendlich anderen Teilnehmern klar kommen.

6.1 Was ist Wireless?

Als Wireless wird die Signalübertragung ohne Leiter bezeichnet, oder einfach «Drahtlos-Verbindung» oder «Funk» genannt.

Damit man ein Signal überhaupt drahtlos übertragen kann, muss man das elektrische Signal vom Leiter abstrahlen. Dies geschieht über Antennen, das sind im Grunde genommen sog. «nicht abgeschlossene»-Leiter. Was das genau ist, besprechen wir im *Kapitel 7 : Leitungstheorie*.

Durch die Form der Antenne können gewisse Ausbreitungs-Charakteristika erzielt werden.

6.1.1 Verhalten unterschiedlicher Trägerfrequenzen

Das Verhalten des abgestrahlten Signals ist dabei stark von der Frequenz des Signals abhängig.

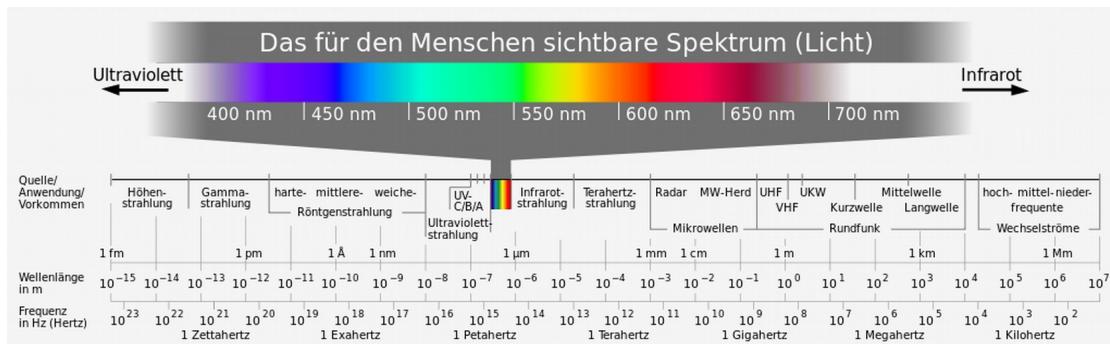


Abbildung 6.1: EM Spektrum

https://upload.wikimedia.org/wikipedia/commons/thumb/1/15/Electromagnetic_spectrum_c.svg/1024px-Electromagnetic_spectrum_c.svg.png

In Abbildung 6.1 sind EM-Wellen verschiedener Frequenz abgebildet und wie wir diese auffassen. Wir sehen zum Beispiel im Bereich um 500 nm das für das menschliche Auge sichtbare EM-Spektrum. Licht ist eigentlich gar nicht so verschieden zu Funksignal im UKW Bereich. Es hat einfach eine andere Frequenz. Für die Drahtlos-Kommunikationstechnik sind die Frequenzen ab dem *low frequency* (LF $f \approx [30 \text{ kHz}, \dots, 300 \text{ kHz}]$) Band bis zu dem *extremely high frequency* (EHF $f \approx [30 \text{ GHz}, \dots, 300 \text{ GHz}]$) Band, interessant. Abgestrahlte Funkwellen, werden auch *elektromagnetische Wellen*, oder *EM-Wellen* genannt. Unter $f \approx 30 \text{ kHz}$ sind die EM-Wellen kaum mehr abzustrahlen. (Die Antennen werden einfach zu gross) Über $f \approx 300 \text{ GHz}$ ist der Übergang zu sichtbaren Licht. Auch dieses lässt sich noch zur Datenübertragung nutzen, ist aber wieder an ein Medium gebunden (Siehe *Kapitel 5.7 : Lichtwellen-Leiter LWL*). Allgemein, kann man sagen das sich EM-Wellen bei tiefen Frequenzen ähnlich wie Schall- und Wasserwellen verhalten. Sie reflektieren, aber umfliessen auch Objekte.

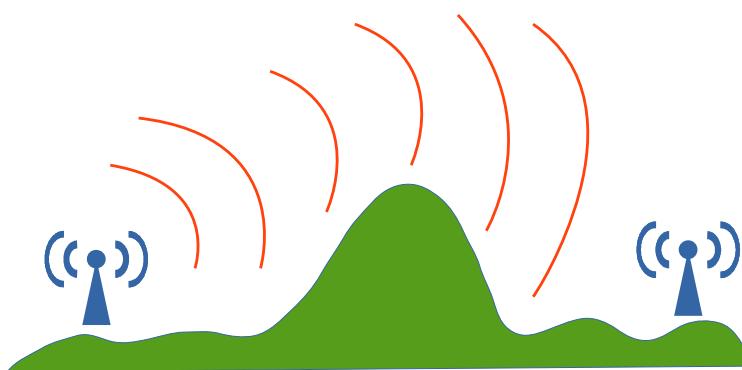


Abbildung 6.2: Strahlungsverhalten von EM-Wellen bei tiefen Frequenzen

Erhöhen wir die Frequenz, verhalten sich die EM-Wellen immer mehr wie Licht. Sie reflektieren mehr, sind «gerichtet» und werden mehr und mehr von Objekten absorbiert.

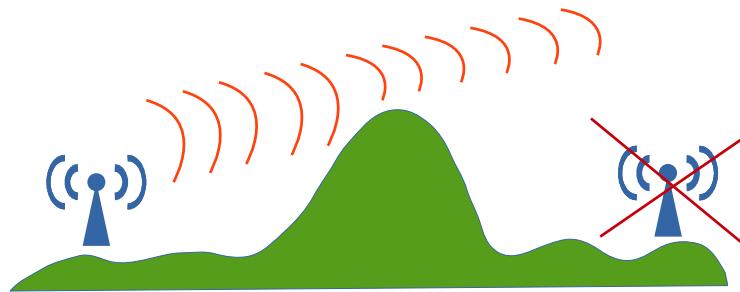


Abbildung 6.3: Strahlungsverhalten von EM-Wellen bei hohen Frequenzen

Bevor unser Planet von einem Satellitennetz umgeben war, nutzte man eine interessante Eigenschaft der Ionosphäre (ein Teil unserer Atmosphäre) für Übertragungen über sehr weite Strecken.

Die Ionosphäre beginnt bei einer Höhe von ca. 80 km und läuft danach in den interplanetaren Raum aus. (in ca. 1000 km Höhe)

Die Ionosphäre besteht aus mehreren Schichten mit unterschiedlicher Elektronendichte und kann zeitlich variieren (auch «verschwinden»). Sie wird durch verschiedene Stäbe (Sonnenstrahlen, Röntgenstrahlung etc) ionisiert.

Das interessante dabei ist, dass die Ionosphäre für EM-Wellen mit ca. $f = [3 \text{ MHz}, \dots, 30 \text{ MHz}]$ wie ein Spiegel erscheint. Damit können wir die EM-Signale an der Ionosphäre gezielt ablenken und auch weit entfernte Funkstationen erreichen.

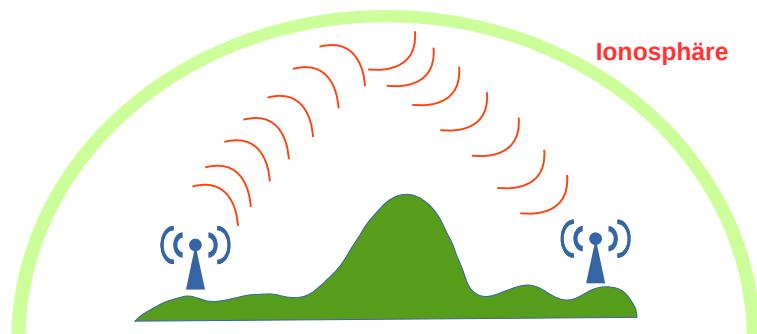


Abbildung 6.4: Strahlungsverhalten von EM-Wellen welche von der Ionosphäre reflektiert werden

Falls Sie Militärdienst geleistet haben, haben sie vielleicht auch schon Langwellen Funksysteme angetroffen welche heute noch die (Not-) Funkverbindungen zur KFOR in den rund 1200 km entfernten Kosovo bereitstellt.

6.2 Wellenarten

EM-Wellen werden in zwei Kategorien eingeteilt. Eine EM-Welle welche auf direkten weg zum Sender ausbreiten und daher nahezu parallel zum Boden ausbreitet, wird **Bodenwelle** oder auch **direkte Welle** genannt. Eine EM-Welle welche zunächst in den Raum abgestrahlt wird und danach an einem Objekt gespiegelt wird (zB. Ionosphäre) wird **Raumwelle** genannt.

6.3 Was ist eigentlich eine Antenne?

Eine Antenne ist ein «elektrotechnisches» Gebilde, welches Signale an den Raum abstrahlt. Man kann sich dies als eine Art LC-Schwingkreis vorstellen.

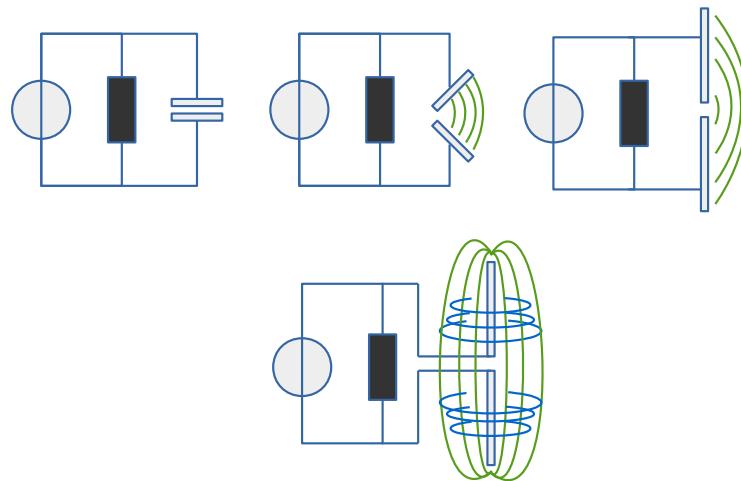


Abbildung 6.5: Antenne = Aufgeklappter Schwingkreis

Wir klappen die Kapazität auf. Das elektrische Feld wird dabei mit aufgezogen und strahlt nun gegen den Raum. Wenn wir das Gebilde nun in Resonanz betreiben, haben wir sehr grosse Ströme durch die Kondensatoren «Arme», welche uns ein magnetisches Streufeld erzeugen. Sind die beiden \vec{E} und \vec{B} Streufelder gerade 90° Phasenverschoben lösen sich die Wellen von der Platte und breiten sich in den Raum aus.

6.4 Freiraum Übertragungsleistung

Wir wollen nun abschätzen wie weit wir mit einer EM-Welle senden können. Hierzu müssen wir uns überlegen wie sich die Sendeenergie über die Distanz verhält.

Für unsere Überlegung nehmen wir an, dass sich die (Sende-) Antenne wie ein **Isotroper-Strahler** verhält. Damit ist gemeint, dass die gesamte Sende-Energie gleichmäßig über eine Kugelform verteilt wird. Wir wissen, dank Albert Einstein und seiner relativitäts Theorie, dass sich eine EM-Welle idealerweise mit Lichtgeschwindigkeit ausbreitet. Somit nimmt der Radius der Kugel mit Lichtgeschwindigkeit zu. Die Kugel bläht sich auf.

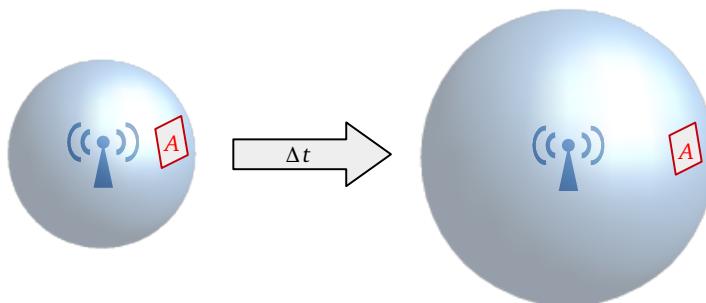


Abbildung 6.6: Energieverteilung einer sich ausbreitenden EM-Welle

Wir sehen in Abbildung 6.6 die EM-Energie-Kugel zu zwei Zeitpunkten. Aus dem Energieerhaltungssatz folgt, dass zu beiden Zeitpunkten gleich viel Energie auf der Oberfläche der Kugel verteilt sein muss. Wir nehmen an, eine Empfangsantenne würde immer die Energie der Fläche A absorbieren können.

So absorbiert die Antenne im linken Beispiel wesentlich mehr Energie aus der Kugel, als in der rechten Darstellung. Zusätzlich wissen wir, dass sich die Kugeln mit nahezu Lichtgeschwindigkeit ausdehnen. Somit können wir die Energie zu Distanz Abhängigkeit berechnen. Ich fasse mich kurz: Hier ist die Lösung:

$$\begin{aligned} P_{rx} &= P_{tx} \left(\frac{\lambda}{4\pi \cdot d} \right)^2 & [P] = W \\ \lambda &= \frac{c}{f} & [\lambda] = m \end{aligned} \quad (6.1)$$

Wir haben die Sendeleistung bei der Sendeantenne P_{tx} gegeben und können über die Wellenlänge λ des Sendesignals und den Abstand d von Sende- und Empfangsantenne, die Empfangsleistung bei der Empfängerantenne P_{rx} bestimmen.

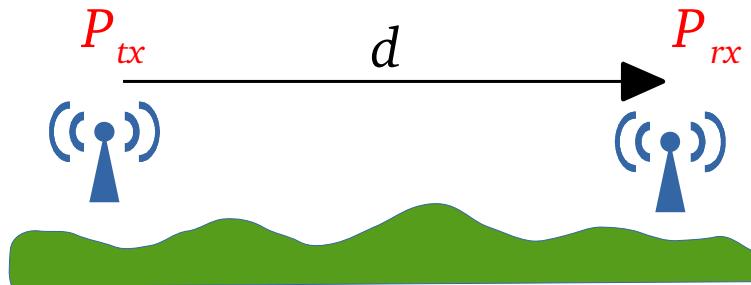


Abbildung 6.7: Freiraum Übertragungsleistung

6.4.1 Freiraumdämpfung

Gebräuchlicher ist die Verwendung der sog. Freiraumdämpfung F (engl. Path-Loss). Die Änderung der Signalleistung über die Distanz kann als Dämpfung aufgefasst werden:

$$\begin{aligned} A_F &= \frac{P_{tx}}{P_{rx}} = \left(\frac{4\pi \cdot d}{\lambda} \right)^2 \\ A_{dB F} &= 10 \log_{10} (F) = 20 \log_{10} \left(\frac{4\pi \cdot d}{\lambda} \right) & [A_F] = 1 \\ &= 20 \log_{10} \left(\frac{4\pi \cdot d \cdot f}{c} \right) = 20 \log_{10} \left(\frac{4\pi}{c} \right) + 20 \log_{10} (d \cdot f) & [A_{dB F}] = dB \\ &= -147.55 dB + 20 \log_{10} (d) + 20 \log_{10} (f) \end{aligned} \quad (6.2)$$

Wichtig

Aus der Formel sehen wir zwei wichtige Dinge:

- $F_{air} \sim \frac{1}{\lambda^2} = \frac{f^2}{c^2}$
- $F_{air} \sim r^2$
- Die Freiraumdämpfung nimmt **quadratisch mit Frequenz zu**. Das heisst: Je grösser unsere Sendefrequenz, desto weniger weit können wir senden.
Das ist einer der Gründe wieso im Mobilfunk die Upload Frequenzen (Mobiltelefon zu Basisstation) unter der Download Frequenzen gewählt wurden.
- Die Freiraumdämpfung nimmt **quadratisch mit der Distanz zu**. Interessanterweise haben wir in (5.4) die Dämpfung eines Kabels mit $A \sim 10^d$ zunimmt. (also Exponentiell anwächst). Auf grosse Distanzen weist eine Drahtlosverbindung weniger Dämpfung auf, als das beste Kabel der Welt!

6.4.2 Antennengewinn / Antenna-Gain

Je nach Antennen Bauform unterscheidet sich die Verteilung des EM-Feldes um die Antenne. Durch eine gezielte Kopplung kann dies konzentriert werden, so dass die Antenne in genau einer Richtung gut empfängt / sendet. Im Gegenzug wird der Empfangs- / Sendeleistung in alle anderen Richtungen schlechter. Damit man nun verschiedene Antennen miteinander vergleichen kann, wird die Strahlungscharakteristik immer auf einen **Isotropen Strahler** bezogen.

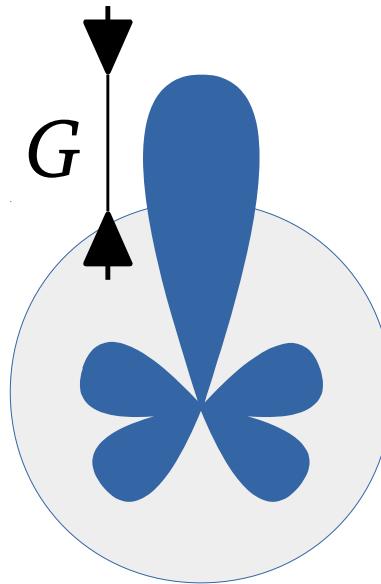


Abbildung 6.8: einfaches Beispiel einer Antennen-Strahlungscharakteristik

Der Gewinn in der Direktionsrichtung wird als Antennengewinn bezeichnet.

$$G = \frac{P_{Antenne}}{P_{iso}} \quad [G] = 1$$

$$G_{dB} = 10 \cdot \log_{10} \left(\frac{P_{Antenne}}{P_{iso}} \right) = 10 \cdot \log_{10}(G) \quad [G_{dB}] = dBi \quad (6.3)$$

6.4.3 äquivalente isotrope Strahlungsleistung

Die *äquivalente isotrope Strahlungsleistung* engl. *equivalent isotropically radiated power* oder kurz **EIRP** genannt. Dieser Wert gibt die effektive (Signal-) Feldstärke in der Luft an. Dieser Wert wird häufig als Grenzwert in div. Übertragungsstandards genannt.

Die Berechnung ist einfach :

$$EIRP = P \cdot G \quad [EIRP] = W$$

$$EIRP_{dB} = P_{dB} + G_{dB} \quad [EIRP_{dB}] = dB \quad (6.4)$$

6.4.4 Freiraum Sende-Distanz

Wir brauchen nun noch eine Angabe und können abschätzen wie weit wir mit einem bestimmten Sender senden können.

Wir müssen wissen : wie viel Signal-Leistung muss am Empfangssystem ankommen?

Empfänger-Empfindlichkeit

Diese Grösse nennt sich **Empfänger-Empfindlichkeit**, in englisch **Receiver-Sensitivity** oder **Rx-Sensitivity**. Diese wird in dBm/Hz angegeben.

Zum jetzigen Zeitpunkt merken Sie sich folgendes:

Wichtig

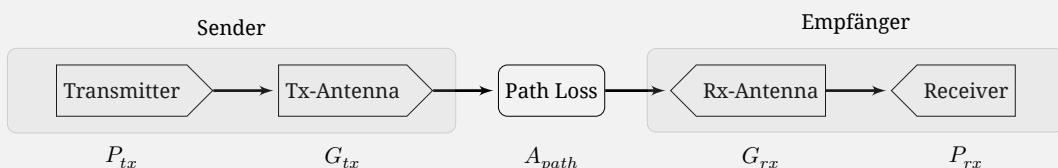
Die Receiver-Sensitivity gilt nur für eine bestimmte Datenrate.
Mehr Datenrate = mehr Bandbreite = weniger weit senden.

Aufgabe 6.38 : max Sendedistanz

Beispiel

Gegeben ist ein Sendesystem mit einem Sender, welcher eine Sendeleistung von $P_{tx} = 10 \text{ mW}$ und eine Sendeantenne mit $G_{tx} = 1$ bei 433 MHz aufweist. Der Empfänger weist eine Antenne mit $G_{rx} = 2.5 \text{ dBi}$ und einer Empfangsempfindlichkeit von $P_{rx} = -80 \frac{\text{dBm}}{20 \text{ kHz}}$ auf. Wie weit können Sie die Sende und Empfangsantenne voneinander entfernen, so dass das System noch funktioniert?

Antwort



$$P_{tx} = 10 \log_{10}(10 \text{ mW}) + 30 = 10 \text{ dBm}$$

$$G_{tx} = 10 \log_{10}(1) = 0 \text{ dBi}$$

$$P_{rx} = P_{tx} + G_{tx} + G_{rx} - A_{path}$$

$$\begin{aligned} A_{path} &= P_{tx} + G_{tx} + G_{rx} - P_{rx} \\ &= 10 \text{ dBm} + 0 \text{ dBi} + 2.5 \text{ dBi} - (-80 \text{ dBm/Hz}) = 92.5 \text{ dB} \end{aligned}$$

Das heisst : Wir können auf unserem Signal-Pfad 132.5 dB dämpfen und unser Wireless System funktioniert gerade noch.

Unter der Annahme, dass unser Sender und Empfänger eine «Sichtverbindung» haben, können wir die Berechnung der Freiraumdämpfung verwenden, um die Distanz abzuschätzen.

$$\begin{aligned} A_{dB_F} : -147.55 \text{ dB} + 20 \log_{10}(d) + 20 \log_{10}(f) &= 92.5 \text{ dB} \\ 20 \log_{10}(d) &= 92.5 \text{ dB} + 147.55 \text{ dB} - 20 \log_{10}(433 \text{ MHz}) \\ &= 67.3 \text{ dB} \\ d &= 10^{67.3/20} = 2.3 \text{ km} \end{aligned}$$

Diese Abschätzung ist wirklich eine **best-offer** Abschätzung. In der Praxis nehmen Sie besser 1/3 dieses Wertes oder noch weniger...

Grösse	Formelzeichen	Einheit
Sendeleistung (Transmitter)	P_{tx}	W
Empfangsleistung (Receiver)	P_{rx}	W
Distanz	d	m
Frequenz	f	Hz
Wellenlänge	λ	m
Freiraumdämpfung	A_F	1
Antennengewinn	G	1
Empfänger-Sensitivität	S_{rx}	$\frac{W}{Hz}$

Tabelle 6.1: Formel-Zeichen / -Einheiten : Wireless- / Drahtlos-Kommunikation

6.4.5 Zuverlässigkeit

Wenn wir Daten über eine Wireless-Verbindung übertragen, müssen wir immer mit einer reduzierten Zuverlässigkeit der Verbindung rechnen.

Die Wireless-Signale können an Objekten reflektiert oder absorbiert werden, bei sich bewegenden Hindernissen haben wir zusätzlich zu den sich ändernden Reflexionseigenschaften noch einen sog. **Doppler-Effekt** welcher unser Sendesignal moduliert. Oder kurz ausgedrückt : Bei Wireless kann immer alles schief gehen.

Die heutigen Wireless-Systeme verwenden alle Tricks der modernen Ingenieurskunst, damit wir die Verbindung so gut wie möglich hin kriegen!

Wir werden in den Kapiteln *Kapitel 10 : Fehlererkennung* / *Kapitel 13 : Fehlerkorrektur* und *Kapitel 16 : Flussteuerung* ein paar Strategien kennen lernen, wie wir solch eine Verbindung absichern können.

Nice to Know

Der Rest dieses Kapitels ist nicht Prüfungsrelevant

6.5 WLAN

Wireless Local Area Network WLAN ist das Pendant zum Ethernet LAN. In der Regel wird WLAN mit dem Standard der IEEE-802.11 Familie assoziiert.

WLAN wird in dem sog. industrial, scientific and medical (ISM) Frequenzband betrieben.

6.5.1 industrial, scientific and medical frequency band

Um im ISM-Frequenzband senden zu dürfen, muss man keine Lizenzen des Standortlandes erwerben. (**royalty free**) Aber man muss sich an die nationalen Bestimmungen halten. In der Schweiz ist dies im Nationalen Frequenzzuweisungsplan [O2] [NaFZ](#) der BAKOM festgehalten.

Eine (nicht geschlossene) Auflistung bekannter ISM Frequenbänder:

Anwendung	Frequenzbereich	(cumulative) Bandbreite
RFID	13.553, ..., 13.567 MHz	14 kHz
Short-Range-Devices (1)	433.050, ..., 434.790 MHz	1.74 MHz
Short-Range-Devices (2)	863.0, ..., 870.0 MHz	7.0 MHz
WLAN 2.4 GHz	2.400, ..., 2.4835 GHz	83.5 MHz
WLAN 5 GHz	5.150, ..., 5.875 GHz	725 MHz

Tabelle 6.2: ISM Frequenbänder

6.5.2 WLAN Standards

Derzeit wird WLAN in den folgenden Standards verwendet:

Standard	Frequenzband	Modulation	MIMO	Brutto-Datenrate	Netto-Datenrate
802.11	2.4 GHz	DBPSK / DQPSK	-	2 Mb/s	0.9 Mb/s
802.11a	5 GHz	QAM64	-	54 Mb/s	23 Mb/s
802.11b	2.4 GHz	DQPSK	-	11 Mb/s	4.3 Mb/s
802.11g	2.4 GHz	QAM64	-	54 Mb/s	19 Mb/s
802.11n	2.4 GHz / 5 GHz	QAM64	2x2 / 3x3 / 4x4	600 Mb/s	240 Mb/s
802.11h	5 GHz	QAM64	-	54 Mb/s	-
802.11ac	5 GHz	QAM256	3x3 / 8x8	1.3, 2.6, 6.9 Gb/s	0.66, 3.5 Gb/s
802.11ad	60 GHz	QAM64	-	< 6.7 Gb/s	-
802.11ah	0.9 GHz	OFDM	4x4	347 Mb/s	-
802.11ax	2.4/5.0 GHz	OFDM	MU-MIMO	< 12 Gb/s	-

Tabelle 6.3: WLAN Standards

Wir sehen in *Tabelle 6.3*, dass zur Zeit eine ganze Reihe Standards im gleichen Frequenzbereich existieren und auch nebeneinander eingesetzt werden.

Die 5 GHz Standards verbreiten sich zur Zeit zunehmend in Europa. Dieser Frequenzbereich wurde in der EU erst seit ungefähr 2012 für das ISM Band freigegeben. (Vorher war dieser Frequenzbereich für eine andere Anwendung im Nationalen Frequenzzuweisungsplan eingetragen) Die angegeben Brutto- / Netto-Datenraten sind **best-offer** und werden in der Realität nur selten erreicht. Der 802.11ac Standard wird zusätzlich in Wave 1 / Wave 2 unterteilt und die ersten Geräte die diesen Standard unterstützen wurden 2015 vorgestellt.

Der 802.11ax Standard wird voraussichtlich mitte 2019 fertig gestellt. Erste Geräte welche bis etwa ≈ 10 Gb/s erreichen sind derzeit bereits erhältlich.

Der 802.11ad Standard bietet zwar sehr hohe Datenraten, aber die sehr hohe Trägerfrequenz sorgt dafür, dass das Sendesignal von fast allen festen Materialien absorbiert wird und ist damit nur für sehr kurze Distanzen geeignet.

MIMO Multiple Input Multiple Output (MIMO) bezeichnet ein Sende- / Empfangsverfahren, bei dem mehrere Antennen eingesetzt werden um das gesendete Signal zu empfangen. Durch korrelieren der einzelnen Empfangssignale kann das Rauschen besser unterdrückt werden, und somit ein besser S/N heraus gerechnet werden, womit auch die erreichbare max. Datenrate erhöht werden kann.



Abbildung 6.9: Ein TP-LINK Router mit 4x4 MIMO Technologie
https://static.digitecgalaxus.ch/Files/6/7/1/3/8/9/4/AD7200_un_V1_1185_large_2.00_20160125105911.jpg?fit=inside%7C464:368&output-format=progressive-jpeg

In *Abbildung 6.9* ist ein WLAN Router mit 4x4 MIMO Technologie dargestellt. Sie sehen die charakteristischen 8 Antennen, welche es für 4x4 MIMO zwingend benötigt.

Modulation Die angegeben Modulationen, *differential binary phase-shift keying DBPSK* und *differenti- al quadrature phase-shift keying DQPSK* gehören zur Kategorie die Phasenumtast-Modulationen [W54]. Die *quadrature amplitude modulation QAM-xxx* [W56] ist eine Weiterentwicklung der DBPSK / DQPSK Modulationen, welche bei gleicher Bandbreite noch höhere Datenraten erlaubt. Die QAM Modulation ist dabei das modernste Modulationenverfahren welche ein hohe Spektrale-Effizienz erreicht.

6.6 RFID

RFID [W57] ist wieder eine englische Abkürzung und steht für **Radio-Frequency Identification**. Ein RFID System besteht aus einem *Reader* (Lesegerät) und diversen *Tags* (Funketiketten / Transponder). Jeder RFID Tag besitzt im minimum eine 96 Bit Kennnummer und meist noch ein wenig Speicher. RFID sollte den Barcode durch eine Funketikette ersetzen.

Wir wissen hat dies RFID bis heute noch nicht geschafft. Aber RFID wird zB. beim Swiss-Pass eingesetzt, beim markieren von Haustieren, zur Mauterfassung von LKW's und zur Materialverwaltung in vollautomatischen Hochregallagern.

Die Tags gibt es in 3 Ausführungen:

- **passiv** ohne Batterie, zieht Energie aus dem Sendesignal des Readers. Übertragungsdistanz bis 30 m
- **semi-passiv** ohne Batterie, aber Sendesignal wird über eine **backscatter** Modulation aus dem Sendesignal des Readers erzeugt. Übertragungsdistanz bis 100 m
- **aktiv** mit Batterie. Übertragungsdistanz bis 1000 m

RFID wird unter anderem in den folgenden Frequenzbändern verwendet:

Anwendung	Frequenzband
Tierimplantate	125 kHz
Smart-Tags	13.56 MHz
Wide Range Tags	433/863.0 MHz

Tabelle 6.4: RFID Frequenzbänder

Der sog. NFC-Standard, welcher unter anderem für das *berührungslose Bezahlen* mit Kreditkarte oder Smartphone verwendet wird ist ein Substandard von RFID.

6.7 NB-IoT / M2M

Die sog. NB-IoT Technologien sind schmalbandige (Narrow-Band) Funk-Protokolle für das Internet-of-Things. In anderen Worte : Sehr langsame Funkverbindungen die dafür geschaffen sind sehr viele «low-power Nodes» anzubinden welche nur wenige Daten versenden.

Ein weiterer Begriff für die gleiche Art Technologien ist M2M : Machine-to-Machine Communication.

NB-IoT	Frequenzband	Reichweite	Datenrate
LoRa	169/433/868/915 MHz	10 – 15 km	1.2 – 300 kbit/s
Sigfox	868/902 MHz	3 – 10 km	100 – 600 bit/s
Weightless	433/868/915/923 MHz	≈ 2 – 5 km	0.2 – 100 kbit/s
Cat-M1	LTE Band	-	1 Mbit/s
Cat-NB1	LTE Band	-	250 kbit/s
Bluetooth 5 Long Range	2.4 GHz	400 – 1000 m	0.250 – 2 Mbit/s

Tabelle 6.5: NB-IoT Protokolle / Technologien

6.7.1 LoRa

LoRa [W41] ist ein Akronym für **Low-Power Wide Range**.

Bei LoRa ist die Leitungscodierung Patent geschützt und nur Semtech stellt derzeit Chipsätze her.

LoRa hat ausserdem ein LPWAN Protokoll definiert, so dass eine LPWAN LoRa Node mit jeder LoRa LPWAN Basistation kommunizieren kann. Ein Vorteil der Leitungscodierung von LoRa ist, dass diese sehr gut in Gebäuden empfangen wird. Die ersten Wissenschaftlichen Untersuchungen zeigen allerdings, dass die Technologie nicht sehr gut skaliert. Damit ist gemeint, dass eine einzelne LoRa Basisstation nicht sehr viele (1000-2500) Nodes bedienen können. Als Concentrator muss man mehr Basisstationen aufstellen um mehr Nodes zu verwalten.

Ein Endanwender kann entweder über einen Concentrator Daten versenden oder selbst eine Basisstation aufstellen. Derzeitige Daten-Abo's werden in «Transaktionen pro Tag» abgerechnet.

6.7.2 Sigfox

Sigfox [W63] ist eine Firma in Frankreich und Verwalter des gleichnamigen SIGFOX Protokolls. Das Protokoll zeichnet sich dadurch aus, dass eine einzelne Sigfox-Basisstation sehr viele Nodes verwalten kann. Der Hersteller spricht von bis zu 1 Mio Nodes pro Basisstation.

Ebenfalls speziell ist, dass pro Nachricht nur 12 Bytes Payload und max. 140 Nachrichten pro Tag versenden werden können. Eine Downlink Nachricht kann max. 8 Bytes Payload versenden.

Für Basisstation betreiben fallen Lizenzgebühren für Sigfox an und für Endabnehmer muss bei seinem Concentrator ein Obulus pro Device pro Jahr gezahlt werden.

6.7.3 Weightless

Weightless [W74] ist ein offenes Protokoll und wurde von der *Weightless SIG* spezifiziert. Obwohl es als offen deklariert wurde, müssen alle Geräte von der Weightless SIG qualifiziert werden.

Weightless-N (ein Substandard) ist sehr ähnlich wie Sigfox.

Speziell am Standard ist die Payload Länge von gerade 48 Bytes

Der Vorteil von Weightless ist der offene Standard. Aber als Endanwender muss man auch die Basisstation stellen, da diese derzeit kaum verbreitet von einem Concentrator eingesetzt wird.

6.7.4 Cat-M1 / Cat-NB1

Cat-M1 [W11] und Cat-NB1 sind eigentlich LTE-Modems.

Die Hersteller der LTE-Modem haben erkannt, dass es einen Markt für schmalbandige Datenübertragung gibt. Und das dieser in direkter Konkurrenz zu ihrem Geschäftsfeld aufgezogen wird. Daher haben sie pro aktiv ihre LTE Modem soweit abgespekt, so dass diese die LTE Infrastruktur mitbenutzen aber mit viel geringerer Datenrate.

Dies spielt vor allem den etablierten Telekomunternehmen in die Hände. Diese müssen nur ein Software-Update auf Ihre Basisstationen aufspielen.

Als Endanwender brauchen Sie für jedes Device eine SIM-Karte und ein Datenabo. (Die Datenabo sind aber einiges günstiger, als Beispielsweise für ihr Handy) Ein gewichtiger Vorteil ist, dass ein Cat-M1 oder Cat-NB1 keine tägliche Beschränkung der Datenmenge aufweist. Sie können so viel Daten senden und empfangen bis ihr Guthaben aufgebraucht ist. Daher eignet es sich für Anwendungen die vor allem kurzzeitig viel Daten zu versenden haben.

6.7.5 Bluetooth 5: Long Range

Ein wenig aus der Rolle fällt Bluetooth 5: Long Range [W10].

Eigentlich gehört Bluetooth 5 LR nicht in dieses NB-IoT Kapitel. Die Reichweite ist mit 400 – 1000 m sehr viel geringer als bei der Konkurrenz.

Im Gegensatz zu den anderen Technologien ist Bluetooth 5 nach dem Kauf aber wirklich kostenfrei!

BL5 unterstützt auch sog. Mesh Netzwerke womit sich die Reichweite massiv erhöhen kann. Ausserdem kann BL5 bei guten Bedingungen mit sehr hohen Datenraten senden und empfangen.

Wir haben keine Beschränkungen der Upload und Download Datenmenge. Es eignet sich somit für nicht so weit verteilte Netze welche aber moderate stetige Datenmengen zu versenden haben.

Verbreitung Schweiz / Europa

Die Schweiz setzt in den NB-IoT Technologien derzeit auf LoRa. Der Rest von Europa vor allem auf Cat-M1 / Cat-NB1.

6.8 Zusammenfassung

Sie haben nun einen kurzen Abriss über die Drahtlos-Kommunikation erhalten und kennen nun die wichtigsten Grössen eines Wireless Systems und einige davon können Sie ausrechnen.

Sie haben ausserdem erfahren, dass Sie beim BAKOM Informationen erhalten und das der National Frequenzzuweisungsplan wichtige Informationen enthält.

Desweiteren wurden ein paar heute verwendete Wireless-Standards vorgestellt, welche Sie in Ihrem Berufszweig schon mal gehört haben sollten.

7

Leitungstheorie

Inhalt des Kapitel

7.1	Geschichte	106
7.2	Leiter Impedanz / Admittanz	107
7.2.1	abgeschlossene Leitung	108
7.2.2	offene Leitung	109
7.2.3	kurzgeschlossene Leitung	110
7.2.4	beliebig abgeschlossene Leitung	110
7.3	Reflexionsfaktor	112
7.3.1	Video : Theorie Reflexionen	113
7.4	Time-Domain Reflektometer	114
7.5	Elektrische Länge eines Leiters	115
7.5.1	Splitter	115

Zum Inhalt

In diesem Kapitel besprechen wir die sog. Leitungstheorie. Wie in den Kapiteln zuvor angedeutet, werden wir hier erfahren, was ein abgeschlossener Leiter ist und wieso dieser in der Kommunikationstechnik so wichtig ist.

7.1 Geschichte

Die Telegrafie hat eine lange Geschichte. 1809 wurde der erste elektrochemische Telegraf in München entwickelt. 1835 folgte die Entwicklung eines Nadeltelegrafen. Federführend bei der Verbreitung der Telegrafentechnik waren die Eisenbahnunternehmen zu dieser Zeit. Bereits 1870 war ein Grossteil der Erde schon verkabelt. Die Kabel wurde in der Regel neben den Eisenbahnschienen aufgestellt. Zu dieser Zeit, waren dies wohl die längsten von Menschen hergestellten Kabeln. Und auch die damit verbundenen technischen Probleme tauchten auf.

Was waren das für Probleme?

Textnachrichten werden mit Morsecodes «verschlüsselt» und mit einem Telegrafen über weite Strecken übermittelt.

Mit dem Telegrafen wird eine unter Spannung stehendes Leiterpaar kurzzeitig kurzgeschlossen, die Empfängerseite hat einen Lautsprecher an diesen Kabeln montiert und piepst bei einer Spannungsänderung.

Bei sehr langen Telegrafen Verbindungen hörte man **Echos** der bereits gesendeten Signale.

Was noch interessanter war, manchmal erschienen diese Echos ebenfalls bei eher kurzen Leitungen, welche zB. durch einen See verlegt wurde. Aber woher kamen diese Echo's?

Vermutlich war es William Thomson, 1. Baron Kelvin, welcher die sog. Telegrafengleichung aufstellte und die damit hergehenden Probleme mit den Unterseekabeln des Telegrafennetzes beschrieb. Oliver Heaviside hat die Telegrafengleichung auf ein langes geradliniges Zweileitersystem angewendet und die partiellen Differentialgleichungen angepasst. Damit konnte nun der Strom- und Spannungsverlauf auf einer Leitung zu jedem Zeitpunkt an jedem Ort berechnet werden.

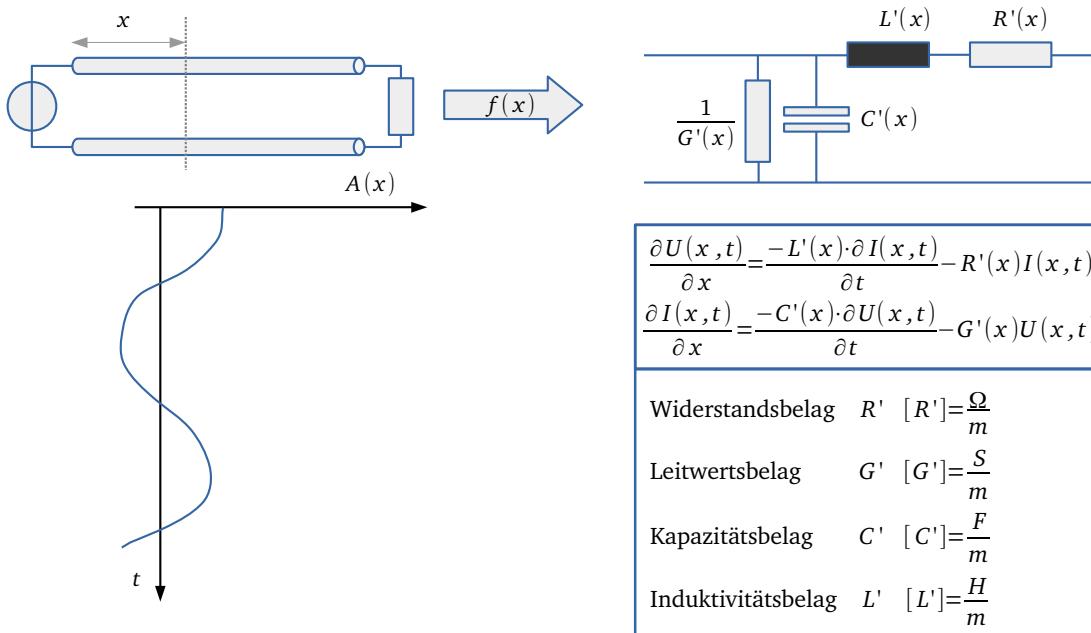


Abbildung 7.1: Telegrafen-Modell und -Gleichung

Wir sehen in Abbildung 7.1 unser Leitungsmodell (oben rechts). Links sehen wir unser Leitungssystem. Gleich darunter ist die Spannung am Punkt x in Abhängigkeit der Zeit t abgebildet. Dies soll nur veranschaulichen, dass wir an Punkt x «mathematisch in den Leiter» schauen können.

Die eigentliche Telegrafengleichung ist rechts mit den zwei partiellen Differentialgleichungen abgebildet. Sie sehen, dass diese Gleichung unheimlich kompliziert ausschaut. Sie lässt sich auch nur für Spezialfälle berechnen.

Gleich unter der Gleichung steht nun die genaue Bezeichnung unserer Symbole!

R' Widerstandsbelag **G' Leitwertsbelag** **C' Kapazitätsbelag** **L' Induktivitätsbelag**

Wir erhalten die Belag-Werte, wenn wir einen ganzen Leiter in kleine Stückchen der Länge Δx zerteilen.
Genug zur Erklärung: Was kann man damit anstellen?

7.2 Leiter Impedanz / Admittanz

Mit der Telegrafengleichung lässt sich aus unserem Leitungsmodell einen komplexen Widerstand berechnen, welcher Stellvertretend alle Eigenschaften der Leitung representiert. Diesen komplexen Widerstand nennt man **Impedanz** Z und das Inverse davon nennt man die **Admittanz** Y :

$$\begin{aligned} Z &= \sqrt{\frac{R' + j\omega L'}{G' + j\omega C'}} \\ Z &\sim \sqrt{\frac{L'}{C'}} \quad \begin{matrix} [Z] = \Omega \\ [Y] = \Omega \end{matrix} \\ R' \ll \omega L', G' \ll \omega C' \\ Y &= \frac{1}{Z} \end{aligned} \tag{7.1}$$

Wir sehen in *Formel (7.1)* ein interessantes Phänomen: Die Impedanz entspricht bei einem DC-Strom nur gerade der Wurzel aus dem Widerstandsbelag und dem Leitwertsbelag. (da $\omega = 0$) Aber wenn wir eine sinusartiges Signal auf den Leiter anlegen, besitzt er eine Impedanz welche nur vom Induktivitäts- und Kapazitätsbelag abhängt.

Wichtig

Die Impedanz lässt sich **nicht** mit dem Ohm-Meter messen!

Wie wir noch sehen werden, ist die Impedanz wichtig für die Kommunikationstechnik. Hierfür wurden ebenfalls diverse Standards definiert:

Anwendung	Impedanz
Koaxial-Kabel (RF)	50Ω
Koaxial-Kabel (CATV)	75Ω
Twisted-Pair (Profibus)	120Ω
Twisted-Pair (Telefon)	600Ω

Tabelle 7.1: Impedanz Standards

7.2.1 abgeschlossene Leitung

Wir lernen nun den Begriff der **abgeschlossenen Leitung** kennen.

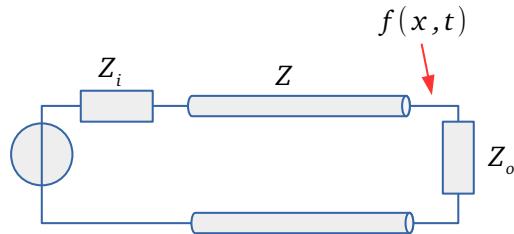


Abbildung 7.2: Schema : abgeschlossene Leitung

In Abbildung 7.2 sehen wir unseren Messaufbau. Wir haben eine Quelle mit einem definierten Innenwiderstand Z_i . Außerdem haben wir eine Leitung, welche ebenfalls eine Impedanz Z aufweist. Und zu guter Letzt haben wir noch einen Abschlusswiderstand Z_0 .

Jedes Kommunikationssystem können wir auf solch ein einfaches Modell herunterbrechen.

Wichtig

Eine Leitung wird als abgeschlossene Leitung bezeichnet, wenn sie am Anfang und am Ende mit der eigenen Impedanz abgeschlossen wird.

$$Z_i = Z_0 = Z$$

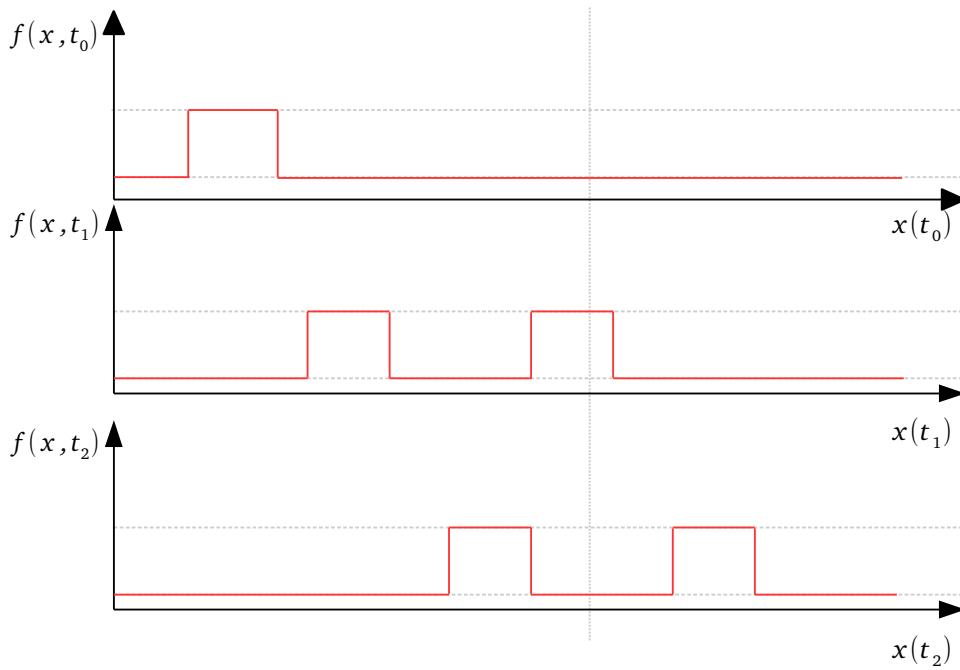


Abbildung 7.3: Signalform : abgeschlossene Leitung

Mithilfe der Telegrafengleich können wir nun «in die Leitung» hineinsehen. Genauer gesagt an die Übergang zwischen Z und Z_0 . (rot markiert in Abbildung 7.2)

Wir haben in Abbildung 7.3 eine vertikale gestrichelte Linie. Das ist genau der Übergang von Z in Z_0 . Links davon sehen wir in das Leitungsstück Z und rechts in den Abschlusswiderstand Z_0 .

Von der Quelle werden zwei **rechteckige Pulse** generiert. Wir sehen wie die Pulse von der Leitung her

in den Abschlusswiderstand überlaufen. Alles ohne Probleme.
Das ist der Ideal-Fall und wenn immer möglich versuchen wir eine Leitung abzuschliessen.

7.2.2 offene Leitung

Als nächstes sehen wir uns einen Extremfall an. Der Abschlusswiderstand ist unendlich gross $Z_0 = \infty$.
(Quasi nichts angeschlossen)

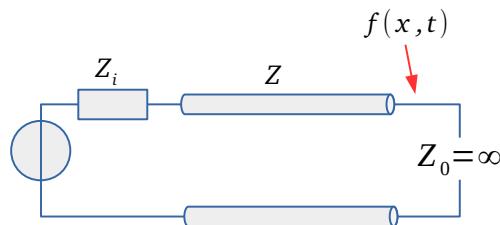


Abbildung 7.4: Schema : offene Leitung

In Abbildung 7.4 ist die Ersatzschaltung abgebildet.

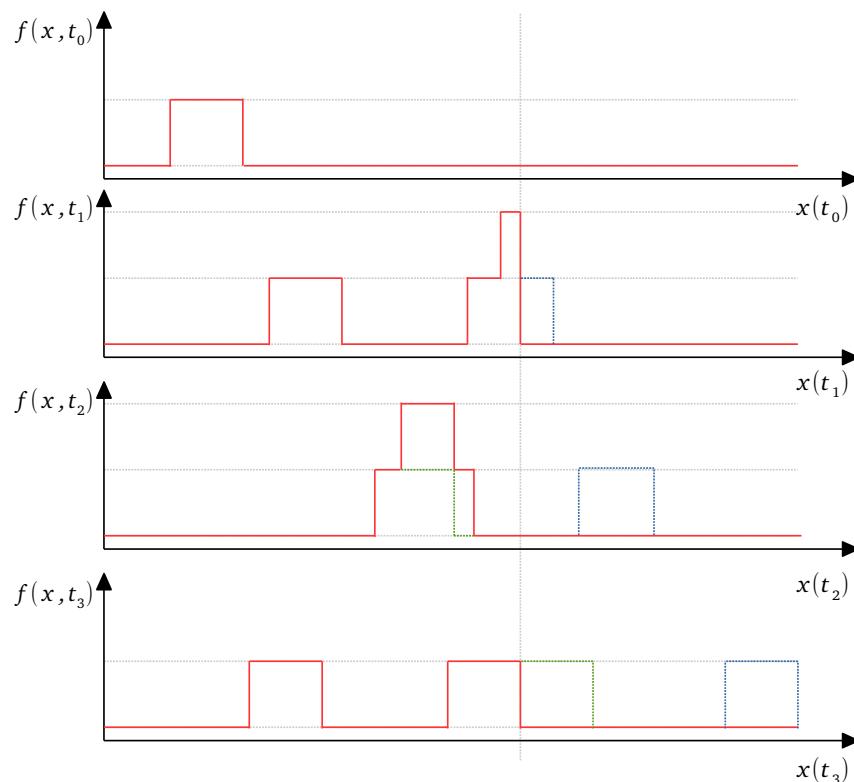


Abbildung 7.5: Signalform : offene Leitung

Interessanterweise verhält sich das System zu beginn recht ähnlich. Wenn wir eine genügend lange Leitung verwenden, sieht die Quelle nämlich nur die Kabelimpedanz und das gar kein Verbraucher angeschlossen ist, merkt diese das gar nicht.

Zu beginn ist in Abbildung 7.5 alles beim alten, die Pulse werden von der Quelle erzeugt und traversieren durch den Leiter. Am Übergangspunkt von Z und Z_0 geschieht jetzt etwas interessantes : Der Puls wird am Ende der Leitung **reflektiert** und überlagert sich mit dem **vorwärtslaufenden** Puls.

In der Grafik ist **grün** und **blau** das Verhalten abgebildet, wenn die Leitung abgeschlossen wäre. (Dies aber nur zur Visualisierung)

7.2.3 kurzgeschlossene Leitung

Sehen wir uns nun noch den zweiten Extremfall an. In diesem Fall ist unser Lastwiderstand kurzgeschlossen $Z_0 = 0 \Omega$.

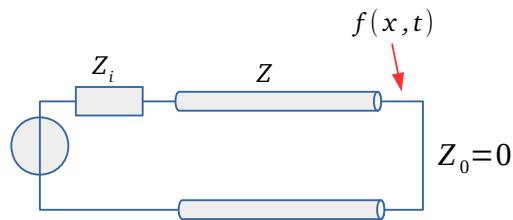


Abbildung 7.6: Schema : kurzgeschlossene Leitung

In Abbildung 7.6 ist die Ersatzschaltung für den Kurzschluss-Fall abgebildet.

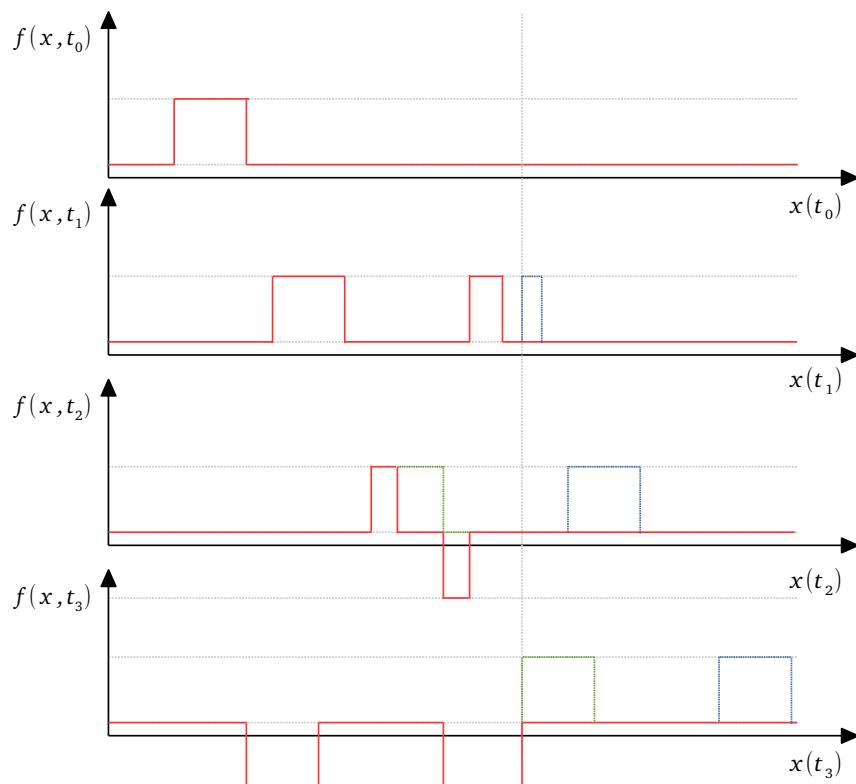


Abbildung 7.7: Signalform : kurzgeschlossene Leitung

Die Quelle sieht bei einem genügend langen Leiter noch immer nur die Leitungsimpedanz und hat keine Ahnung vom Abschlusswiderstand. Wieder ist zu begin alles recht ähnlich.

Die zwei Pulse werden wie gehabt erzeugt und traversieren durch den Leiter. Am Übergangspunkt von Z zu Z_0 geschieht nun folgendes: Der Puls wird am Ende der Leitung mit der **negativen Amplitude reflektiert** und überlagert sich mit dem **vorwärtslaufenden** Puls.

In der Grafik ist grün und blau das Verhalten abgebildet, wenn die Leitung abgeschlossen wäre. (Dies aber nur zur Visualisierung)

7.2.4 beliebig abgeschlossene Leitung

Normalerweise stimmen die Impedanzen nie ganz genau überein. An jeder Stelle an der sich die Impedanzen der Leitung unterscheiden entstehen Reflexionen.

Wir beenden nun das Thema indem wir einen allgemeinen Fall besprechen. Eine *beliebig* abgeschlossene Leitung.

In diesem Beispiel ist der Abschlusswiderstand leicht grösser als die Leiterimpedanz $Z_i = Z$, $Z_o > Z$.

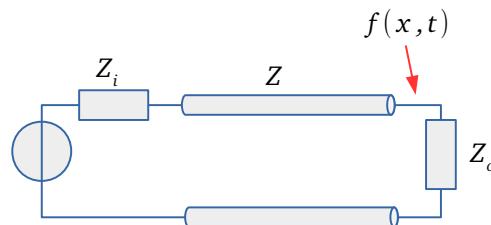


Abbildung 7.8: Schema : beliebig abgeschlossene Leitung

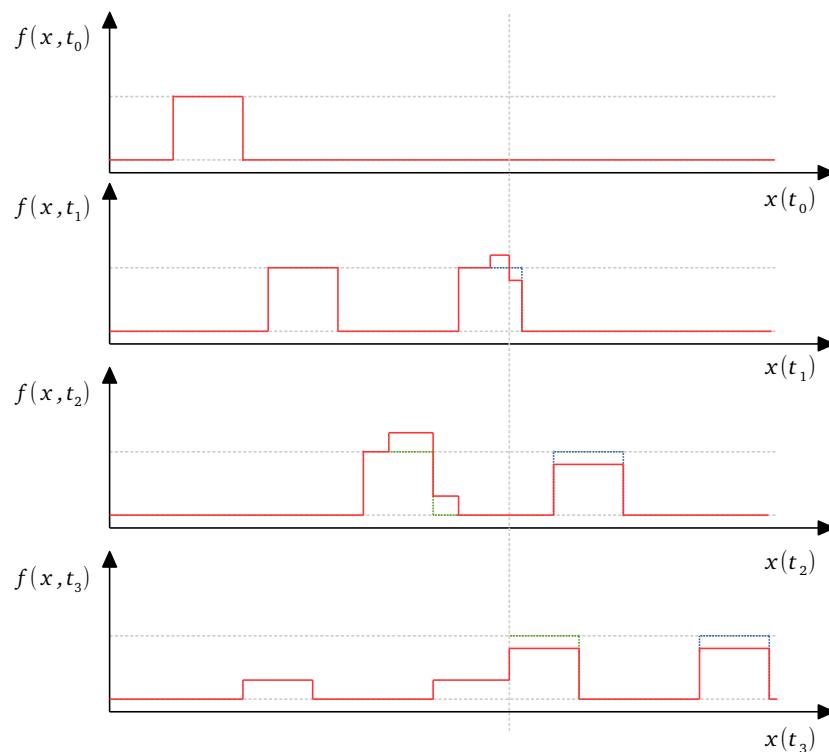


Abbildung 7.9: Signalform : beliebig abgeschlossene Leitung

Da der Abschlusswiderstand leicht grösser als die Leiterimpedanz ist, können wir schon jetzt sagen, dass die Reflexion mit pos. Amplitude reflektiert wird.

So betrachten wir noch Abbildung 7.9 : Die Signale werden wie immer generiert und traversieren durch die Leitung. Am Übergangspunkt wird ein kleiner Teil des Sendesignals reflektiert und überlagert sich mit den vorwärtslaufenden Pulsen. Der Rest traversiert in den Abschlusswiderstand und wird dissipiert.

Wichtig

Ich hoffe Sie erinnern sich noch an Aufgabe 5.34. Wir haben ein beliebiges Zweidraht-System ausgerechnet und gesehen, dass die parasitären Eigenschaften stark mit der Leitungsleitung variieren. Jetzt wissen wir auch wieso dies für High-Speed Kommunikationssysteme überhaupt nicht gut ist. Die ständigen Änderungen der parasitären Eigenschaften und damit der Impedanz führen zu Signalreflexionen.

7.3 Reflexionsfaktor

Wie wir im vorherigen Kapitel 7.2 : Leiter Impedanz / Admittanz gesehen haben, ist das Signalverhalten vom Widerstandsverhältnis des Leitungs- und des Abschlusswiderstandes abhängig. Wir verallgemeinern diesen Zusammenhang nun noch ein wenig. Reflexionen entstehen nämlich immer, wenn zwei unterschiedlich **Impedanzbeläge** zusammengeschlossen werden.

Mit einem sog. *Richtkoppler* lassen sich die *Vorwärtlaufende Welle* und *Rücklaufende Welle* separat messen. Aus dem Spannungs- $\frac{U_R}{U_V}$ resp. Stromverhältnis $-\frac{I_R}{I_V}$ der vor- und zurücklaufender Welle lässt sich der Reflexionsfaktor Γ berechnen:

Wichtig

Die Impedanz wird über das Verhältnis : Vorlaufende Welle / Rücklaufende Welle gemessen.

Entweder mit einem *Richtkoppler* oder mit einem *Netzwerkanalyzer*

$$\Gamma = \frac{U_R}{U_V} = -\frac{I_R}{I_V} = \frac{Z_2 - Z_1}{Z_2 + Z_1} \quad [\Gamma] = 1 \quad (7.2)$$

$$|\Gamma| \leq 1$$

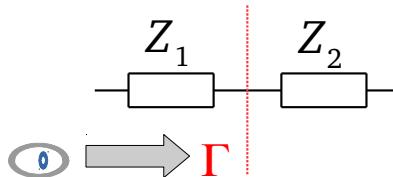


Abbildung 7.10: Schema : Reflexionsfaktor

In Formel (7.2) ist die Berechnung des Reflexionsfaktors dargestellt. Je nachdem von welcher Seite man die Reflexion betrachtet, ist diese positiv oder negativ. Es kommt also auf die **Sichtweise** an! In Abbildung 7.10 ist die Sichtweise abgebildet.

Der Reflexionsfaktor ist so konstruiert, dass er bei einer vollständigen Reflexion am offenen Ende einer Leitung den Wert $\Gamma = 1$ annimmt. Während er bei einer vollständigen Reflexion einer Kurzgeschlossenen Leitung den Wert $\Gamma = -1$ annimmt. Sind die beiden Widerstände nahezu identisch, nimmt der Reflexionsfaktor einen Wert von $\Gamma = 0$ an. Man spricht dabei von zwei angepassten Widerständen.

Anpassung	Reflexionsfaktor	Impedanzen
Optimal angepasst	$\Gamma = 0$	$Z_1 = Z_2$
Vollständige Reflexion (offene Leitung)	$\Gamma = 1$	$Z_1 \ll Z_2 \sim \infty$
Vollständige Reflexion (kurzgeschlossene Leitung)	$\Gamma = -1$	$Z_1 >> Z_2 \sim 0 \Omega$

Tabelle 7.2: Reflexionsfaktor : Interpretation und Wertebereich

In der Hochfrequenztechnik, lassen sich viele Effekte nur durch die Spannungsverhältnisse von vor- und rücklaufenden Wellen charakterisieren. Damit ist der Reflexionsfaktor ein essentieller Qualitätsfaktor. Außerdem wird mit dem Reflexionsfaktor und einer normierten Systemimpedanz, die Impedanzwerte von zB. Kabeln oder anderen Bauteilen bestimmt.

Sind Reflexionen aus irgendwelchen Gründen unumgänglich, haben sie die verzwickte Situation, dass sie die Leistung ihres Signals nicht bis zum Verbraucher schicken können. Auch wenn dieser mit dem Ohmmeter nur einen kleinen Widerstand aufweist, wird bei einem Hochfrequenzsignal die Leistung einfach reflektiert.

Praxis Tipp

Wenn Sie mit Hochfrequenz-Signalen hantieren, können am Ausgang eines Signalgenerators Reflexionen entstehen!
 Bei offenem Signalausgang muss der Ausgang die doppelte Spannung aushalten können.
 Bei kurzgeschlossenen Signalausgang muss der Ausgang die einfache negative Spannung aushalten können.

Grösse	Formelzeichen	Einheit
Widerstands-Belag	R'	$\frac{\Omega}{m}$
Leitwerts-Belag	G'	$\frac{S}{m}$
Kapazitäts-Belag	C'	$\frac{F}{m}$
Induktivitäts-Belag	L'	$\frac{H}{m}$
Impedanz	Z	Ω
Admittanz	Y	S
Spannung der vorwärtslaufenden Welle	U_V	V
Spannung der rückwärtslaufenden Welle	U_R	V
Strom der vorwärtslaufenden Welle	I_V	A
Strom der rückwärtslaufenden Welle	I_R	A
Reflexionsfaktor	Γ	1

Tabelle 7.3: Formel-Zeichen / -Einheiten : Reflexionsfaktor / Impedanz

7.3.1 Video : Theorie Reflexionen

Wie sie nun gehört haben, sind Reflexionen und ihre Eigenschaften schon lange bekannt. Ich habe hier noch ein Youtube Video einer alten Lehrsendung aus den Bell-Labs. Das interessante an diesem Video ist, dass die Reflexionen an einem mechanischen System vorgeführt werden. Schlussendlich ist es egal, ob elektrische- oder mechanische-Reflexion, beide Systeme werden über die gleichen Differentialgleichungen beschrieben.

[AT&T Archives: Similiarities of Wave Behavior \(Bonus Edition\)](#)

Wer hats bemerkt?

Im Video bei min. 4:53 erzählt der Kommentator, dass die Natur konsistent ist und dass bei einem elektrischen System mit «open-circuit» (also einer Abschlusswiderstand mit $Z_0 = \infty$) die elektrische Welle mit ungleichem Vorzeichen reflektiert wird.

Das ist gerade umgekehrt wie in meinen Unterlagen. Wie kann das sein?

Strom und Spannung verhalten sich gerade inverse. Im Video wird aus der Sicht vom Strom argumentiert. In meinen Unterlagen vom Standpunkt der Spannung.

7.4 Time-Domain Reflektometer

Für was können wir die Leitungstheorie noch gebrauchen?

Heutzutage gibt es interessante Messgeräte. Ein ganz bestimmtes, erzeugt Pulse und misst die Amplitude sowie die Laufzeit des reflektierten Signal. Diese Messgeräte nennt man **time-domain Reflektometer**.

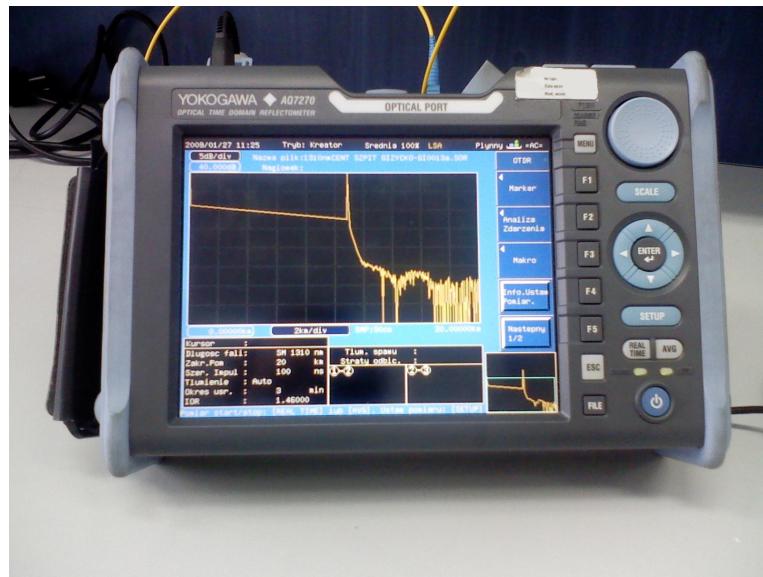


Abbildung 7.11: Time-Domain Reflektometer SQ7270

[https:](https://upload.wikimedia.org/wikipedia/commons/0/08/OTDR_-_Yokogawa_AQ7270_-_1.jpg)

[/upload.wikimedia.org/wikipedia/commons/0/08/OTDR_-_Yokogawa_AQ7270_-_1.jpg](https://upload.wikimedia.org/wikipedia/commons/0/08/OTDR_-_Yokogawa_AQ7270_-_1.jpg)

Mit diesen kann man Störstellen in Kabeln nicht nur feststellen, man kann sogar bestimmen **wie weit** diese vom Einspeisepunkt entfernt sind. Eingesetzt werden diese zum Beispiel in der Telekommunikationsbranche um Defekte in einem vergrabenem Kabel zu eruieren.

So das man weiß, wo man das Kabel ausgraben muss, damit man es reparieren kann.

Damit wir uns das ganze ein wenig besser vorstellen können, hier nochmals ein Video:

Achtung : Das Video ist ein Werbevideo eines Hersteller. Weder bewerbe ich diesen Hersteller noch
nehme ich Zahlungen dessen entgegen.
[TDR - Change the world of Time Domain Reflectometry measurement](#)

In Geheimdienst kreisen wird mit einem time-domain Reflektometer (TDR) überprüft, ob eine Telefonleitung physikalisch abgezweigt wird. In der Aviatik wird mit der TDR die Verkabelung überprüft. Mit der sog. *spread-spectrum time-domain Reflektometrie* kann dies sogar während dem Betrieb durchgeführt werden.

Nach dem gleichen Schema, aber mit Funk, kann mit der time-domain Reflektometrie, zB. die Dichte von Beton und deren Struktur bestimmt werden.

7.5 Elektrische Länge eines Leiters

Für die Praxis ist es wichtig zu wissen, ab wann wir auf Reflexionen Rücksicht nehmen müssen. Hierzu gibt es eine Faustregel:

empirische Faustregel

- Die Leitungstheorie ist relevant sobald eine Leitung elektrisch «**lang**» ist.
- Ein Leitung ist elektrisch «**lang**» sobald $l_{el} \geq \frac{\lambda}{10}$.
- Die Wellenlänge λ entspricht der höchsten Übertragungsfrequenz auf dem Leiter.

Wie in der Faustregel angegeben ist, brauchen wir zum einen die Wellenlänge λ des Signals auf dem Leiter und die elektrische Länge des Leiters l_{el} .

Wichtig!

Ein Leiter ist mechanisch *kürzer* als seine elektrische Länge!

$$l_{el} = l_{mech} \cdot \sqrt{\epsilon_r} \quad (7.3)$$

7.5.1 Splitter

Wenn Sie mehrere Kabel angepasst miteinander verbinden möchten, können Sie diese nicht Reflexionsfrei einfach so zusammenhängen.

Sie brauchen dafür einen sog. **Splitter** welcher an allen Anschlüssen die gleiche Impedanz aufweist.

Splitter

Ein Splitter muss an allen Anschlüssen mit der Bezugs-Impedanz abgeschlossen sein damit er richtig funktioniert.

Offene Anschlüsse werden mit einem Abschluss abgeschlossen. (Einen Abschluss nennt man auch Terminator). Beachten Sie : Ein Splitter hat pro zusätzlichen Anschluss eine Dämpfung von mind. $A_{Splitter} > (n - 1) \cdot 3 \text{ dB}$

Man kann auch Splitter bauen, welche an den Anschlüssen unterschiedliche Impedanzen aufweisen.

Zusammenfassung

- Sie haben in diesem Kapitel gehört, dass jeder Leiter eine Impedanz aufweist
- Ebenfalls haben Sie gehört, dass wenn wir ein (Wechsel-) Signal auf eine Leitung senden, sollte diese abgeschlossen sein, ansonsten generieren wir Reflexionen
- Abgeschlossen ist eine Leitung, wenn diese am Anfang und am Ende die «eigene» Impedanz **sieht**
- Sie haben gelernt was der Reflexions-Faktor ist und wie man diesen berechnet
- Sie haben gelernt, dass wenn Sie einen Signalanschluss nicht richtig abschliessen, an diesem entweder die doppelte oder die neg. Signalamplitude anliegen kann
- Kabel sind mechanisch «**kürzer**», als ihre elektrische Eigenschaft
- Mit time-domain Reflektometer können sie Störstellen finden und deren Abstand bestimmen
- Mit einem «normalen» Reflektometer können Reflexionen gemessen werden und damit auf Fehler oder die Kabel-Impedanz geschlossen werden
- Ein gut angepasster Verbraucher macht keine Reflexionen
- Je höher die Übertragungsfrequenzen, desto eher haben Sie Probleme mit Reflexionen

8

UART (2)

Inhalt des Kapitel

8.1 Rekapitulation	118
8.2 Leitungs-Abschluss	118
8.3 Fehlererkennung / Error Detection	119
8.3.1 Parity Bit	119
8.4 UART im OSI-Modell	121
8.5 Zusammenfassung	121

Zitat:

Air conditioner speaks serial, just like everything else...

Die Klimaanlage spricht «serial», fast wie alles andere auch...

Lewin Day : Autor @ Hackaday.com

Zum Inhalt Wie haben im *Kapitel 3 : Die serielle Schnittstelle RS-232 / UART* die «Serielle Schnittstelle» kennen gelernt.
Wir werden in diesem Kapitel noch ein wenig tiefer in diesen Standard eintauchen und weitere Themen der Kommunikationstechnik exemplarisch einführen.

8.1 Rekapitulation

Zur Rekapitulation hier nochmals kurz, was wir bereits über RS-232 / UART wissen:

- Wir schicken die Daten seriell über ein Kabel
- Das Kabel hat zwei Pegel, low und high und stehen für die *Informations-bits*
- Ein Nachrichtenpacket besteht aus *start-bit*, *Daten-bits*, *stop-bit* (ev. *idle*, ev. *parity-bit*)
- Man braucht kein zusätzliches Taktsignal
- Es gibt einen Hin-Leiter und (optional) einen Rück-Leiter

Zur Erinnerung, hier nochmals die Abbildung 3.11:

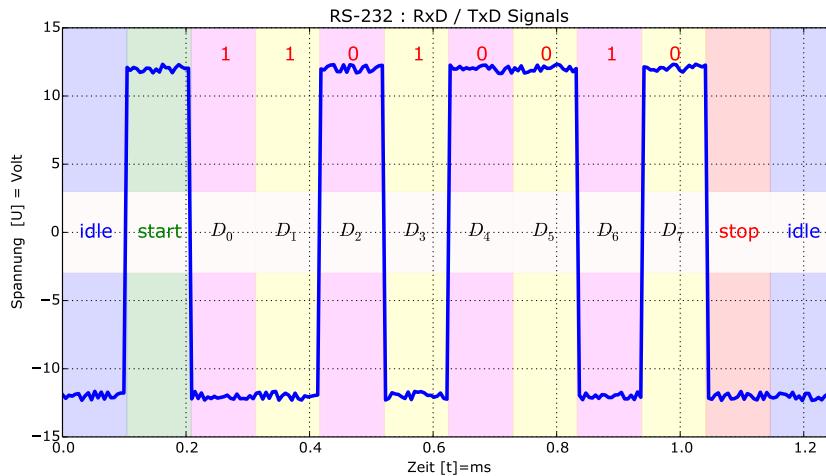


Abbildung 3.11: RS-232 Leitungscode

Was Sie noch nicht wissen können : Dieser Leitungscode wird NRZI-Polar (Non-Return-to-Zero Inverse Polar) genannt. Wir werden im Kapitel 9.3 : Non-Return-to-Zero NRZ noch genauer darauf eingehen.

8.2 Leitungs-Abschluss

Wir haben im Kapitel 7 : Leitungstheorie gesehen, dass wir Leitungen abschliessen sollten.

Die RS-232 / UART Schnittstelle ist eine alte Schnittstelle, welche man für kurze Strecken entwickelt hat. Daher wird diese nicht abgeschlossen. Auf längeren Leitungen finden wird recht ansehnliche Reflexions-Artefakte.

Info

RS-232 eignet sich nicht für längere Leitungen. Unter anderem, da die Leitung bei RS-232 nicht abgeschlossen wird.

Info 17: RS-232 : Leitungsabschluss / Reflexionen

Man versuchte dazumals die Signalqualität mit viel Sende-Leistung hoch zu halten. Daher wird bei RS-232 auch solch hohe Signalpegel von $[-15 \text{ V}, \dots, +15 \text{ V}]$ verwendet.

8.3 Fehlererkennung / Error Detection

Wie wir bereits im *Kapitel 6 : Wireless / Drahtlos* gehört haben, kann es vorkommen, dass eine Nachricht auf dem Weg zum Empfänger durch Störungen verändert wird. Eine Veränderung wäre zB. dass durch zu viel Rauschen ein Bit falsch detektiert wird. Oder das durch eine starke Stromänderung neben einem Empfangskabel, eine so grosse Störspannung induziert wird, dass der Empfänger gleich ein paar Bits falsch detektiert.

8.3.1 Parity Bit

Wir haben in *Kapitel 3.4.2 : Parity Bit* einmal gehört, dass es ein sog. **Parity-Bit** gibt. Dieses Parity-Bit dient dazu, dass wir als Empfänger einer Übertragung erkennen können, ob unsere Nachricht auf dem Weg zu uns verändert wurde.

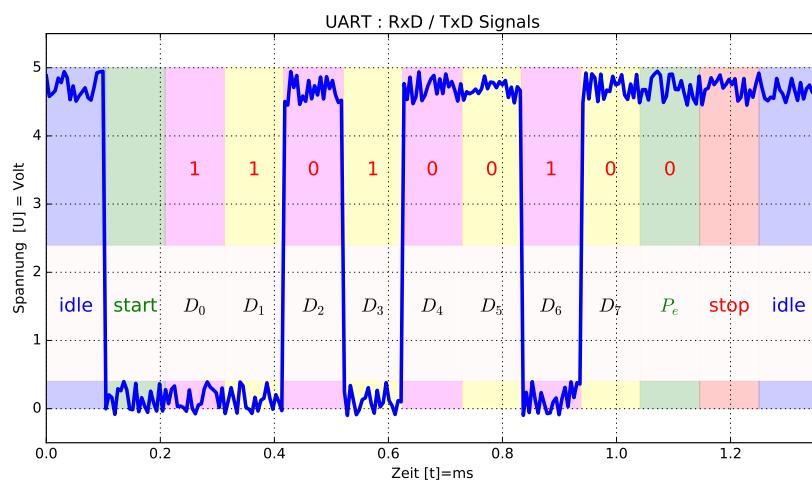


Abbildung 8.1: even Parity-Bit

Sie sehen in Abbildung 8.1 einen Leitungscode mit einem sog. **even parity-bit**. Dadurch, dass der Empfänger ebenfalls das gleiche *parity-bit*, durch die gleiche mathematische Vorschrift bilden kann, können wir nun das *parity-bit* mit dem Empfangsdaten berechnen und mit dem empfangenen *parity-bit* vergleichen. Stimmen die beiden *parity-bit* **nicht** überein haben wir einen Bit-Fehler empfangen.

Berechnung Wir sehen in Abbildung 8.1 ein sog. *even Parity-bit*. Das ist die *reduzierte binäre Quersumme* der Daten. Oder einfach ausgedrückt: Man summiert alle Datenbits, wenn ungerade : $P_e = 1$, wenn gerade : $P_e = 0$

$$x = \sum_i D_i$$

$$P_e = \begin{cases} x \stackrel{?}{=} 2n - 1 & : P_e = 1 \\ x \stackrel{?}{=} 2n & : P_e = 0 \end{cases} \quad (8.1)$$

Theorie zur Fehlererkennung Wir werden im *Kapitel 13 : Fehlerkorrektur* noch genauer auf das Thema eingehen.

Hier eine kleine Einführung:

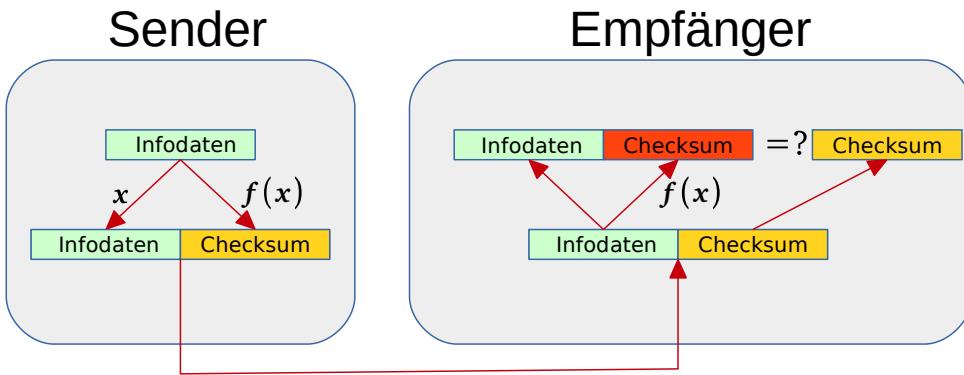


Abbildung 10.4 : Theorie zur Fehlerdetektion

Der Sender nimmt die Daten und berechnet die Checksumme. Die Checksumme wird danach an die Daten angehängt und beide zusammen dem Empfänger geschickt.

Der Empfänger trennt die Checksumme und die Daten. Er berechnet aus den Daten eine eigene Checksumme und vergleicht diese mit der empfangenen Checksumme.

Wichtig

Sind die *empfangene* und die *berechnete* Checksumme **identisch**, sind die Daten (mit grosser Wahrscheinlichkeit) **unverändert** empfangen worden.

Sind die *empfangene* und die *berechnete* Checksumme **nicht identisch**, ist die Nachricht **verändert** empfangen worden.

Info 18: Theorie : Checksummen Vergleich

Die Checksumme wird über eine mathematische Funktion gebildet. Diese ist je nach Standard unterschiedlich. Es gibt kurze Checksummen wie das Parity-Bit aber auch längere und kompliziertere wie zB. ein Cycle Redundancy Check, welcher mehr Fehler erkennen kann.

8.4 UART im OSI-Modell

Wir haben nun viel über die UART Schnittstelle und das OSI-Modell gehört. Es gilt nun die beiden Theorien zu vereinen!

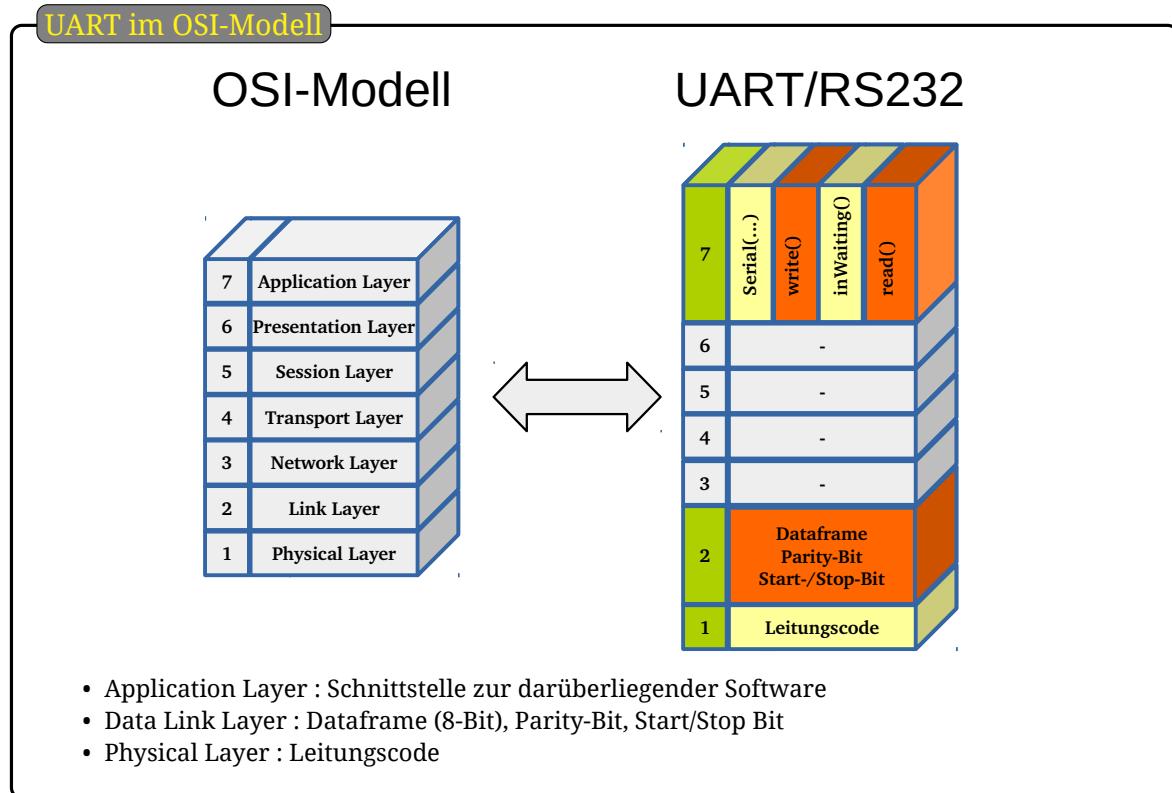


Abbildung 8.2: UART im OSI-Modell

8.5 Zusammenfassung

- Wir haben gesehen wie sich der UART Standard ins OSI-Modell einfügt
- Außerdem wissen wir nun für was das Parity-Bit steht und dass wir noch bessere Methoden kennen lernen werden, wie wir korrupte Nachrichten erkennen können
- Wir haben mit das Wissen aus dem Kapitel 7: *Leitungstheorie* gesehen, dass UART nicht abgeschlossene Leitungssysteme spezifiziert und damit schlecht für schnelle und weitläufige Verbindungen geeignet ist

9

Leitungscodes

Inhalt des Kapitel

9.1	Grundlagen	124
9.1.1	Takt-Synchronisierung	124
Asynchron	124	
Synchron	125	
Takt-Rückgewinnung	125	
9.1.2	Gleichspannungsfreiheit	126
9.2	Übersicht Leitungscodes	127
9.3	Non-Return-to-Zero NRZ	128
9.3.1	Unipolar	128
9.3.2	Polar	128
9.3.3	NRZI	129
9.4	Return-to-Zero RZ	129
9.5	Alternate Mark Inversion (AMI)	130
9.6	High Density Bipolar 3 (HDB3)	130
9.7	Manchester	132
9.8	weitere Codes	134
9.8.1	2B1Q	134
9.8.2	MLT-3/ 4B5B	134
9.8.3	8B10B	135
9.8.4	64B66B	135

Zum Inhalt

In diesem Kapitel besprechen wir verschiedene Leitungscodes und was die jeweiligen Vor- und Nachteile sind. Wir werden die Leitungscodes auf die Eigenschaften : «Taktrückgewinnung» und «Synchronisierung» untersuchen. Außerdem werden wir noch ein paar neue Begriffe der Schaltungstechnik kennen lernen.

9.1 Grundlagen

Wir haben mit dem Shannon-Theorem gesehen, dass die Kanalkapazität resp. Die maximale Bitrate bei idealer Leitungscodierung vom **Signal-zu-Rausch-** Verhältnis abhängt und von der **Bandbreite**.

Die folgenden Leitungscodierungen unterscheiden sich schlussendlich in **Signal-zu-Rausch-** Verhältnis und in der **Bandbreite**.

Die verschiedenen Leitungscodierungen unterscheiden sich ausserdem darin, ob sie sich zur **Taktrückgewinnung** eignen und wie gross der **Gleichspannungsanteil** des Signals ist.

9.1.1 Takt-Synchronisierung

Es gibt zwei Varianten, wie man Kommunikationssysteme synchronisieren kann.

Wir haben exemplarisch in *Kapitel 3.1.2 : parallele Datenübertragung* und *Kapitel 3.1.3 : serielle Datenübertragung* bereits beide gesehen. Entweder wir verwenden am Sender und Empfänger ähnliche Taktquellen, oder wir benutzen eine zusätzliche Signalleitung für die Taktsynchronisierung und synchronisieren alle Teilnehmer auf eine gemeinsame Taktquelle.

Asynchron

Benutzen wir zwei unterschiedliche Taktquellen zum Senden und beim Empfänger zum Abtasten der Daten, so nennen wir dies eine **asynchrone** Verbindung.



Abbildung 9.1: Einfache Darstellung einer Asynchronen Verbindung

Asynchron deshalb, da die beiden Taktquellen nie genau die selbe Taktfrequenz aufweisen.

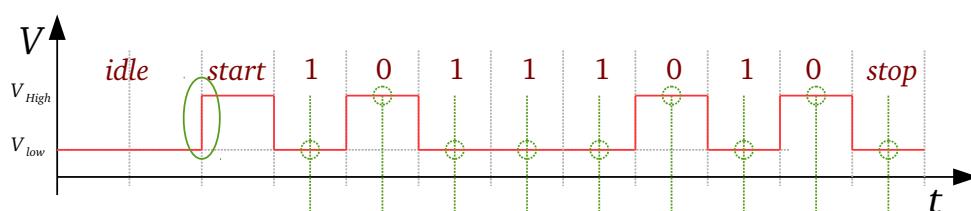


Abbildung 9.2: Timing-Diagramm einer ideal synchronisierten Übertragung

In Abbildung 9.2 ist ein RS-232 Timing Diagram abgebildet. Die *grünen* Striche zeigen die Abtastzeitpunkte des Empfängers. In diesem Beispiel sind die Taktquellen perfekt synchronisiert. Es wird immer in der Mitte von einem Symbol abgetastet.

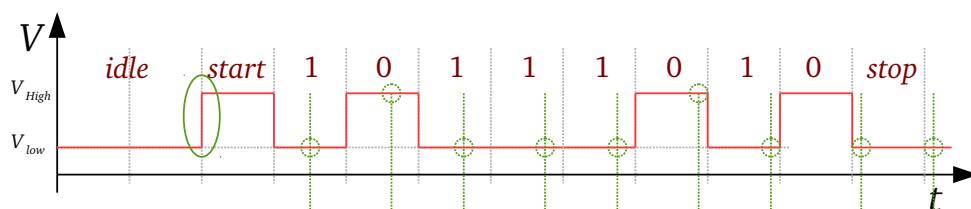


Abbildung 9.3: Timing-Diagramm einer unsynchronisierten Übertragung (fatal)

In Abbildung 9.3 unterscheiden sich die Taktquellen. Wir beobachten wie der Abtastzeitpunkt des Empfängers sich immer weiter verschiebt. Das letzte Bit wird in diesem Beispiel nun falsch abgetastet.

Wichtig

Bei einer asynchronen Verbindung werden die Taktquellen immer weiter auseinanderlaufen, bis alle Daten falsch abgetastet werden.
Es ist lediglich eine Frage der Zeit.

Synchron

Benutzen wir eine gemeinsame Taktquellen für alle Teilnehmer, so nennen wir dies eine **synchrone** Verbindung.

Wir nehmen hier nochmals eine alte Grafik hervor:

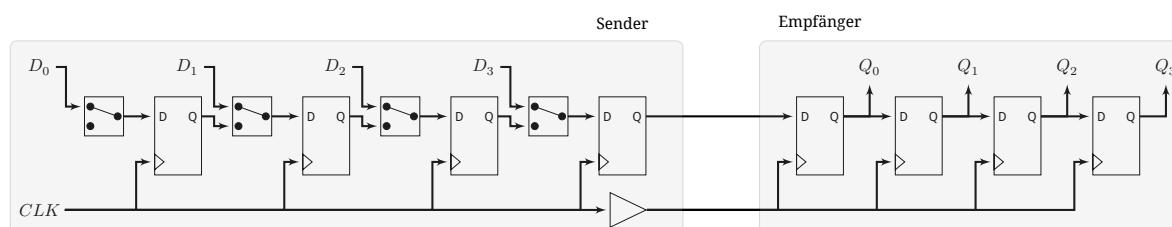


Abbildung 3.4: serielle Datenübertragung

Bei dieser «Allgemeinen» Seriellen Verbindung wurde das Taktsignal mit einer separaten Leitung mitgeführt. Die beiden Sende- und Empfangssystem laufen *synchron*. Anders als bei der asynchronen Verbindung, können wir hier beliebig lange Nachrichten übermitteln, ohne das ein Bit «verloren» geht. Es kann durchaus Sinn ergeben eine separate Takteleitung zu verlegen. Kommunizieren Sie mit einfachen Sensoren, sind diese meist so billig wie möglich, so dass auch ein interner Takt-Generator zu teuer ist. Hochsynchrone Systeme wie zB. NMR Spektrometer sind darauf angewiesen, dass einzelne Subsysteme keine einzige Taktflanke verpassen. Hier ist der höhere Verkabelungsaufwand gerechtfertigt.

Takt-Rückgewinnung

Aber wie können wir grössere Datenmengen zB. über ein WLAN versenden?

Wir haben ja gehört, dass eine **asynchrone** Verbindung immer dazu führt, dass irgendwann die Daten falsch abgetastet werden. Und bei WLAN haben wir nur einen Kanal zur Verfügung!

Wir müssen das Taktsignal also irgendwie **aus den Daten** herausbekommen!

Wichtig

Mit einer Takt-Rückgewinnung, kann der Empfänger-Takt aus den gesendeten Daten korrigiert werden. Hierzu müssen die empfangenen Daten möglichst viele log. **0 / 1** Übergänge aufweisen. (Flanken)

Die Takt-Rückgewinnung ist eine recht komplexe Angelegenheit. Wir können dies leider nur oberflächlich behandeln.

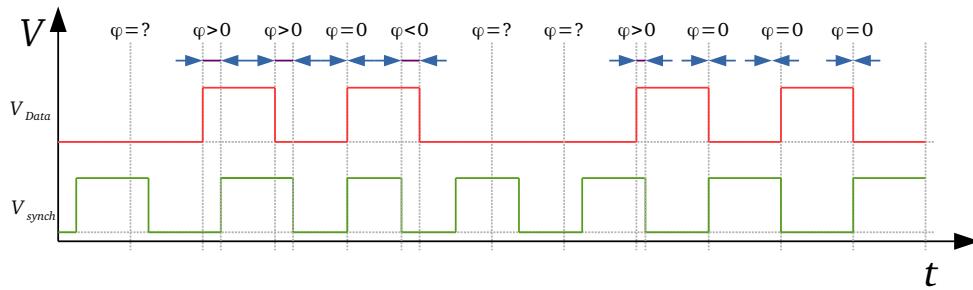


Abbildung 9.4: Takt-Rückgewinnung

Abbildung 9.4 ist stark vereinfacht. V_{Data} ist unser Empfangssignal und V_{Sync} ist das interne synchronisierte Takt-Signal. Wir erkennen, dass mit jeder Flanke des Empfangssignals (egal ob positiv oder negativ) wir wieder die Möglichkeit haben die Flanken des Synchronisationssignals mit dem Eingangssignal zu vergleichen. Enthält das Empfangssignal zu wenig Flanken, kann das Synchronisationssignal nicht verglichen werden und driftet langsam weg. Wir sehen dies in der Grafik. Zunächst haben wir ein paar Flanken und können unser Synch-Signal angleichen, aber bei der Nullserie in der Mitte driftet das Synch-Signal weg. Es muss neu aufsynchronisiert werden.

Wir werden sehen, dass es Leitungscodierungen gibt, welche extra viele Flanken in den Bitstrom einführen, damit die Taktrückgewinnung genügend Flanken erhält.

9.1.2 Gleichspannungsfreiheit

Reale Empfängerschaltungen koppeln eine Leitung meist über Koppelkondensatoren oder über Transformatoren (Baluns) ein. Damit wird die Eingangsschaltung vor externen Spannungsspitzen geschützt und differentielle Signale symmetriert.

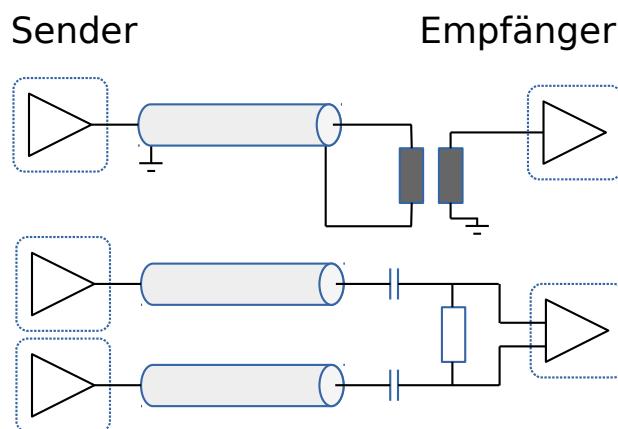


Abbildung 9.5: Empfänger Kopplung

Der Nachteil dieser Ankopplung ist, dass **nur Wechsel signale** über die Kopplungen übertragen werden. Wie sie sich sicherlich erinnern, werden bei Transformatoren **nur Wechsel signale** von der Primärwicklung auf die Sekundärwicklung übertragen. Die Koppel-Kondensatoren wiederum koppeln nur über das Elektrische Feld ein. Auch hier brauchen wir ein Wechsel signal, damit es auf die Sekundärseite des Kondensators koppelt.

Wichtig

Daraus folgt, dass wenn wir ein System mit solch einer Ankopplung verwenden möchten, unser Datensignal möglichst Gleichspannungsfrei sein muss.

Das heisst, dass unser Datensignal **möglichst viele** Spannungs-/Stromänderungen aufweisen sollte. Ausserdem sollte das Signal Bipolar übertragen werden, so dass keine sog. Schwebungen auf dem Leiter entstehen. Schwebungen überlagern unser Nutzsignal und treiben die Kopplungselemente in Sättigung.

Selbststudium / Repetition

Die Funktionsweise eines Transformators ist nicht Umfang dieses Kurses.
Wird aber vorausgesetzt

9.2 Übersicht Leitungscodes

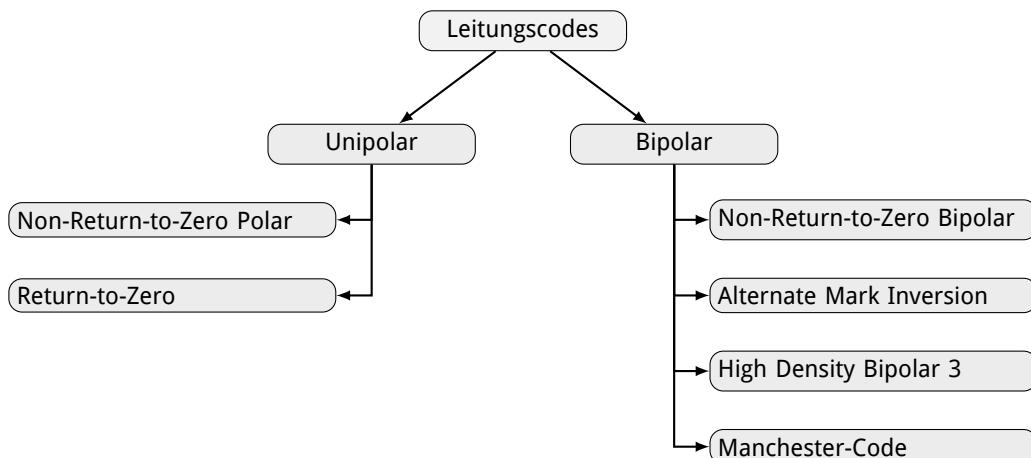


Abbildung 9.6: Leitungscode - Übersicht

Wir werden nun diverse Leitungscodes betrachten. Wir werden sehen, dass jeder seine Vor- und Nachteile hat. Interessant für Sie dürften vor allem der NRZ und der Manchester Code sein, da diese in Profibus / CAN und Ethernet verwendet werden. Der HDB3 zeigt auf, das man die Eigenschaften Taktrückgewinnung und Gleichspannungsfreiheit nur durch «Komplexität» erreichen kann.

9.3 Non-Return-to-Zero NRZ

Der non-return-to-zero (NRZ) Code ist so ziemlich das einfachste was man sich in der digitalen Übertragungstechnik vorstellen kann. Es gibt den Code in 2 Varianten. Der Code ist nicht besonders gut für die Taktrückgewinnung oder für Gleichspannungsfreiheit geeignet.

9.3.1 Unipolar

Es werden zwei Pegel verwendet:

$$V = \begin{cases} \log. 0 : 0 \text{ V} \\ \log. 1 : V_{high} \end{cases} \quad (9.1)$$

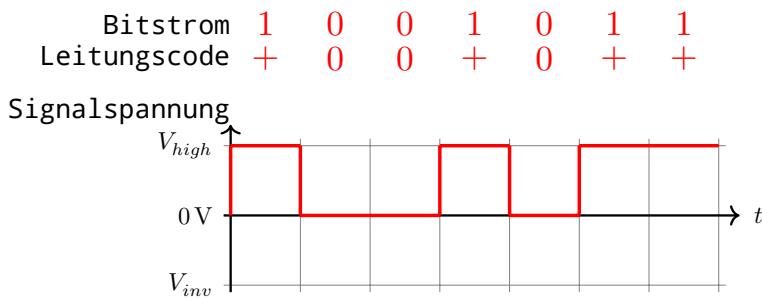


Abbildung 9.7: NRZ Unipolar

NRZ Unipolar ist der rauschbehafteste Code überhaupt. In der Regel wird das NRZ Unipolar bei einfachen Systemen verwendet, die wenig Rauschen aufweisen, aber vor allem einfach und billig sein müssen, zB. Centronic Parallel Port, SPI, I2C.

9.3.2 Polar

Der NRZ Polar Code berücksichtigt den grössten Kritikpunkt des NRZ Unipolar. Anstatt $\log. 0 = 0 \text{ V}$ wird $\log. 0 = V_{inv}$ verwendet.

$$V = \begin{cases} \log. 0 : V_{inv} = -V_{high} \\ \log. 1 : V_{high} \end{cases} \quad (9.2)$$

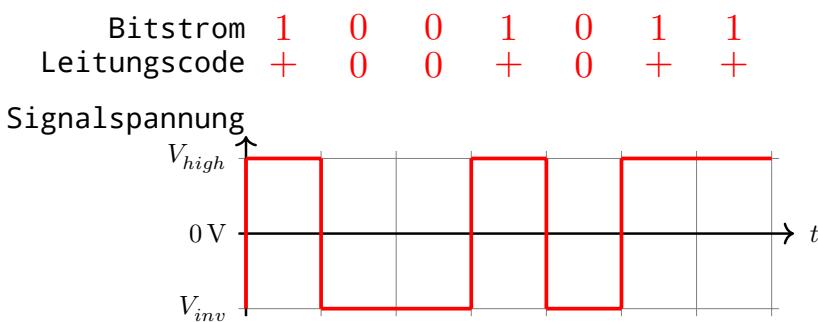


Abbildung 9.8: NRZ (Bi-) Polar

Wie wir in Abbildung 9.8 sehen, sind die beiden Pegel für log. 0 / 1 symmetrisch um 0 V gewählt. Durch die symmetrische Austeuerung kann Gleichspannungsfreiheit erreicht werden. Damit kann auf der Empfängerseite eine Kopplung mit Trafo / Balun / Kondensator realisiert werden um den Empfänger zu schützen. Damit ist der NRZ-Polar besser für längere Verbindungen geeignet als der NRZ Unipolar.

Der NRZ Polar Code wird in der Invers Variante für RS-232 eingesetzt.

9.3.3 NRZI

Der NRZ Inverse Polar (NRZI) Leitungscode wird bei RS-232 eingesetzt. Das **I** steht für Inverse und heisst, dass für eine log. **0** der Pegel V_{high} und für log. **1** der Pegel V_{inv} verwendet wird.

Aufgabe 9.39 : Galvanische-Trennung mit NRZ /NRZI Polar Leitungscode

Frage

Kurze Zwischenfrage : Welche dieser Leitungscodes könnte man mit einer Galvanischer-Trennung mit Transformatoren einsetzen und wieso?
Kennen Sie ein System welches dies so umsetzt?

Antwort

Der NRZ- /NRZI-Polar. Mit dem Polar-Leitungscode verhindern wir ein magnetische Sättigung des Transformators. Allerdings garantiert der Leitungscode keine Gleichspannungsfreiheit, somit ist eine Sättigung des Transformators immer noch wahrscheinlich.

Die Antwort ist somit : **Keiner** dieser Leitungscodes ist geeignet und **Nein** Sie kennen keine Systeme welche dies so umsetzen.

Wird dennoch eine Galvanische-Trennung für solch ein System gefordert, wird dies heute über Digitale-Isolatoren [O3] realisiert.

9.4 Return-to-Zero RZ

Das Ziel des **return-to-zero** Code war eine Verbesserung der Taktrückgewinnung im Vergleich zum NRZ Unipolar. Dies wird erreicht, indem nach jedem log. **1** auf log. **0** zurückgesprungen wird. So wird bei langen log. **1** Sequenzen eine Taktrückgewinnung möglich. Allerdings nicht bei einer log. **0** Sequenz.

$$V = \begin{cases} \log. 0 : 0 \text{ V} \\ \log. 1 : V_{high} \Rightarrow \log. 0 \end{cases} \quad (9.3)$$

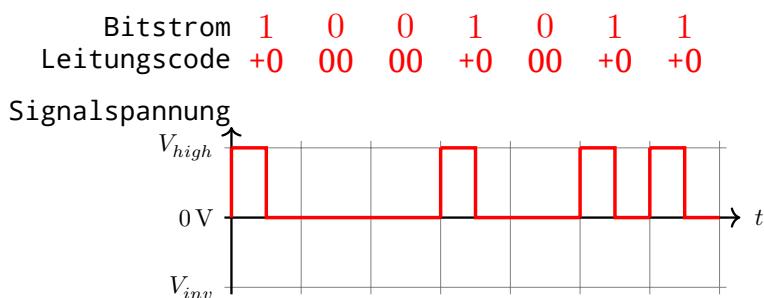


Abbildung 9.9: RZ Unipolar

Das ganze hat aber einen Nachteil. Vergleichen wir Abbildung 9.9 und Abbildung 9.7. Bei beiden Signalen ist die Bitrate die selbe. Auch die Baudrate ist die selbe. Aber der Leitungscode muss schneller zwischen zwei Pegeln wechseln als der NRZ Code. Deshalb braucht der RZ Code im direkten Vergleich doppelt so viel Bandbreite. Wir können also sagen : Wir erkaufen die Funktionalität «Taktrückgewinnung» durch mehr Bandbreitenbedarf. Der **RZ Unipolar** wird zB. bei der IrDa Schnittstelle verwendet.

9.5 Alternate Mark Inversion (AMI)

Der AMI Bipolar Code ist ein **tertiärer** (dreiwertiger) Code. Das heisst, dass unser Leitungscode 3 Spannungspiegel annehmen kann:

$$V = \begin{cases} \log. 0 : 0 \text{ V} \\ \log. 1 : [V_{high}, V_{inv}] \end{cases} \quad (9.4)$$

Auch ein wenig gewöhnungsbedürftig ist die Bezeichnung für eine log. **1** = *Mark* und für log. **0** = *Space*. Der **AMI Bipolar** Code wurde entwickelt um gleichzeitig *Gleichspannungsfreiheit* und *Taktrückgewinnung* zu realisieren.

Der AMI Code **alterniert** immer bei einer log. **1** Sequenz. Somit kann aus einer langen log. **1** Sequenz der Takt extrahiert werden.

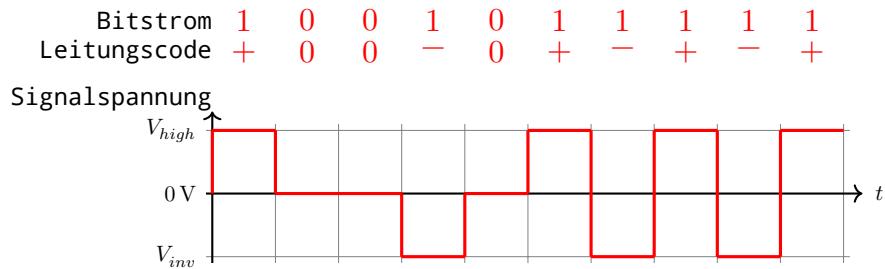


Abbildung 9.10: AMI Bipolar

Der **AMI Bipolar** Code hat aber immer noch eine Schwäche. Aus längeren log. **0** Sequenzen lässt sich der Takt nicht zurückgewinnen. Wir haben die Eigenschaften *Gleichspannungsfreiheit* und *Taktrückgewinnung* in diesem Fall durch (Technische-) Komplexität des Senders / Empfängers realisiert.

Der AMI Bipolar wird im WAN Bereich und im ISDN S0-Bus verwendet.

9.6 High Density Bipolar 3 (HDB3)

Um den AMI Bipolar Code noch zu verbessern fehlt uns noch eine Eigenschaft. Und zwar soll auch der Takt bei einer längeren log. **0** Sequenz wieder zurückgewonnen werden.

Hierzu wurde der **High Density Bipolar 3** Code entworfen. Dieser wird auch *Modified AMI Bipolar* Code genannt.

Grundsätzlich funktioniert der HDB3-Code gleich wie der AMI-Code, aber sollen 4 aufeinander folgenden log. **0** übertragen werden, wird vorsätzlich eine Codeverletzung eingefügt. Dadurch sind genügend Signalfanken für die Taktrücksynchronisation vorhanden, selbst bei einer längeren log. **0** Sequenz.

Was ist eine Codeverletzung? Wir haben gesehen, dass der AMI Bipolar Code bei einer log. **1** immer zwischen V_{high} , V_{inv} alterniert.

Eine Codeverletzung haben wir, wenn zwei mal hintereinander der gleiche Pegel (und somit nicht alternierend) ausgesteuert wird.

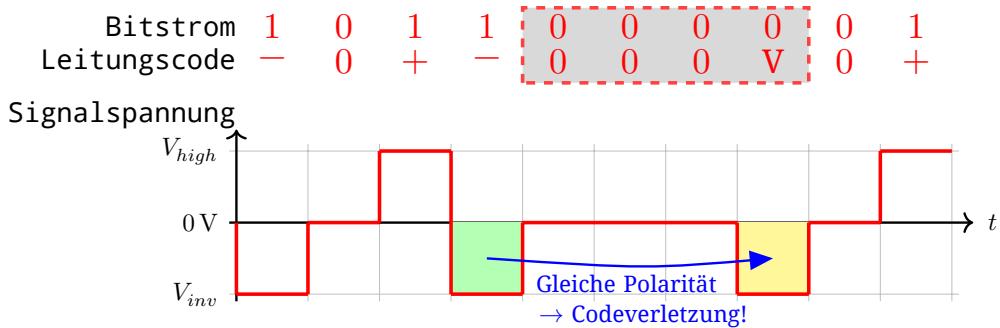


Abbildung 9.11: HDB3 (odd parity = [000V])

Wie man in Abbildung 9.11 erkennt, wird nach 3 Nullen eine Codeverletzung eingefügt. Diese Codeverletzung wird auch als eine [000V] Sequenz bezeichnet. Das heisst 3 Nullen und eine Codeverletzung. Die [000V] Sequenz wird eingesetzt, wenn die Anzahl der vorangegangenen log. 1 Bits ungerade ist. Man nennt dies eine ungerade Parität. $\text{mod}_2(\sum_i |L_i|) = \text{ungerade} \Rightarrow [000V]$

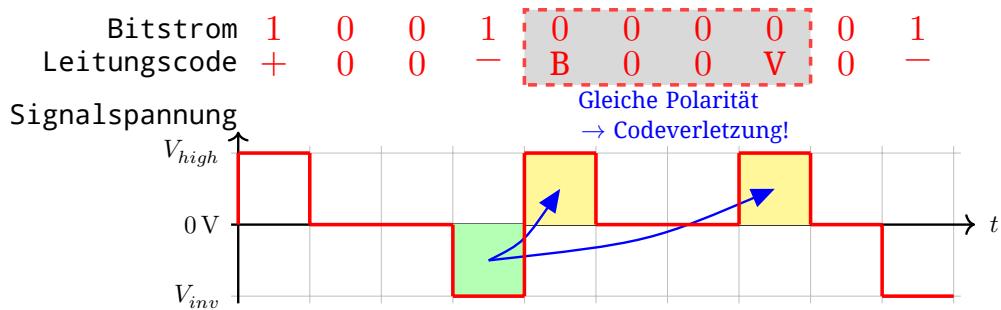


Abbildung 9.12: HDB3 (even parity = [B00V])

In Abbildung 9.12 ist eine Codeverletzung eingezeichnet, für den Fall das die Anzahl vorangegangener log. 1 Bits gerade ist. Diese Sequenz wird [B00V] bezeichnet. Also eine Codeverletzung, zwei Nullen und wieder eine Codeverletzung. Aber die Codeverletzung ist invers zum letzten gesendeten log. 1 Leitungscode. Die [B00V] Sequenz wird eingesetzt, wenn die Anzahl der vorangegangenen log. 1 Bits gerade ist. Man nennt dies auch eine gerade Parität. (Auch eine [000V] Codeverletzung wird mitgezählt).

$\text{mod}_2(\sum_i |L_i|) = \text{gerade} \Rightarrow [B00V]$ Wird eine längere mind. 8 stellige Null Sequenz übertragen, wechselt die Polarität der Codeverletzung.

Durch die [000V], [B00V] Sequenz und den Polaritätswechsel haben wir auch bei langen log. 0 Sequenzen einen geringen Gleichspannungsanteil und es lässt sich eine Taktrückgewinnung realisieren.

Der HDB3 Code wird unter anderem in der Digital-Telefonie eingesetzt : PCM-30

Aufgabe 9.40 : HDB3 Sequenz

Frage

Machen wir hierzu gleich ein Beispiel. Es soll die folgende Bitsequenz in HDB3 übertragen werden : Das erste log. 1 ist ein «+»

Data : 10110000000010000

Antwort

Data : 10110000000010000
Zwischenschritt : 1011000VB00V1000V
Leitungscode : +0-+000+-00-+000+

9.7 Manchester

Kommen wir nun zum **Manchester-Code**. Dieser arbeitet fundamental anders.

Anstatt die Information in den Signalpegel zu stecken, entscheidet die Taktflanke ob ein log. **0** oder ein log. **1** gesendet wurde. Der Signalpegel ist Bipolar.

$$V = \begin{cases} \text{log. 0 : } \uparrow \\ \text{log. 1 : } \downarrow \end{cases} \quad (9.5)$$

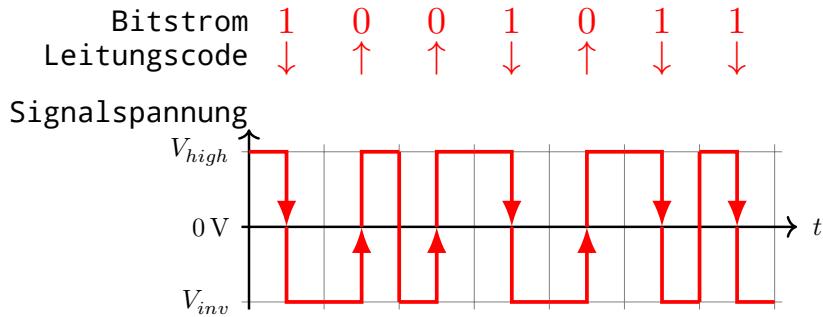


Abbildung 9.13: Manchester Code

Sehen wir uns *Abbildung 9.13* genauer an. Wir erkennen, das immer die Taktflanke in der Mitte des Symbols entscheidend ist. Wir brauchen auch beim Manchester Code daher eine **gute synchronität** mit dem Empfänger. Aber durch die vielen Signalfanken, lässt sich eine gute Taktrückgewinnung implementieren!

Auch hier ist der Bandbreitenbedarf doppelt so hoch, als in einem NRZ codierten Signal mit gleicher Baudrate.

Der Manchester-Code ist der einzige Leitungscode welcher eine sehr gute Gleichspannungsfreiheit und Taktrückgewinnung ohne komplexe Decodierung, Codeverletzungen oder Scrambler realisieren lässt.

(Scrambler : Pseudorandom Generator, welcher den Bitstrom überlagert → Taktrückgewinnung)

Der Manchester-Code wird beim 10 Mbit Ethernet eingesetzt.

Aufgabe 9.41 : AMI Bipolar Sequenz

Frage

Schreiben sie die folgende Sequenz im AMI Bipolar Leitungscode.
Benutzen Sie die Zeichen +, -, 0 für die Pegel und beginnen Sie mit einem positiven Pegel

Data : 00101000011101110000

Antwort

Leitungscode : 00+0-0000+-+0+-+0000

Aufgabe 9.42 : HDB3 Sequenz

Frage

Schreiben sie die folgende Sequenz im HDB3 Leitungscode.

Benutzen Sie die Zeichen +, -, 0 für die Pegel und beginnen Sie mit einem positiven Pegel

Data : 00101000011101110000

Antwort

Data : 00101000011101110000

Zwischenschritt : 00101B00V1110111000V

Leitungscode : 00+0-+00+-++0-+-000-

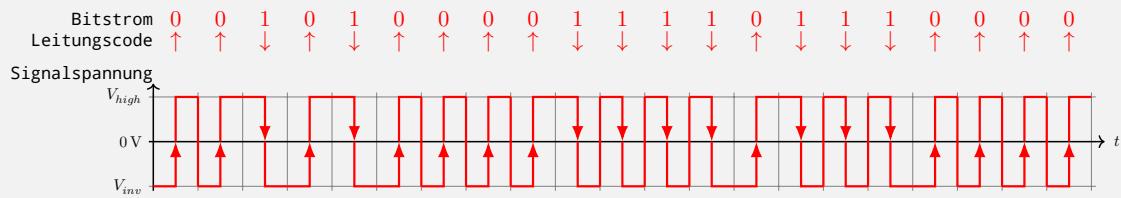
Aufgabe 9.43 : Manchester Code

Frage

Zeichnen sie die folgende Sequenz im Manchester Leitungscode.

Data : 00101000011101110000

Antwort



Vergleich der diskutierten Codes

	NRZ Unipolar	NRZ Polar	RZ	AMI	HDB3	Manchester
Robustheit	-	+	-	+	+	+
Bandbreite	+	+	-	+	+	-
Taktrückgewinnung	-	-	0.5	0.5	+	+
Gleichspannungsfreiheit	-	-	-	+	+	+

Tabelle 9.1: Vergleich der diskutierten Codes

9.8 weitere Codes

Wir haben nun ein paar bekannte Leitungscodierungen diskutiert. Sehen wir uns noch ein neuere Leitungscodes an. Aber wir gehen nicht all zu sehr ins Detail, da dies den Rahmen dieser Vorlesung sprengen würde.

9.8.1 2B1Q

Der **2B1Q** Code ist ein sog. quarternärer (vierwärtiger) Code und wurde im ISDN Netz eingesetzt. Das Design-Ziel war die Reduktion der Bandbreite bei gleichbleibender Datenrate.

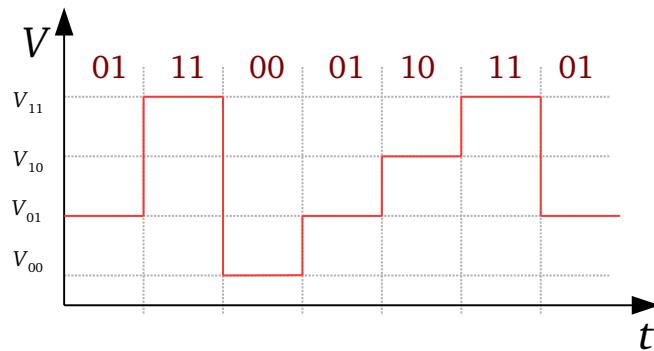


Abbildung 9.14: 2B1Q Code

9.8.2 MLT-3/ 4B5B

Nach dem 10 Mbit Standard (Manchester Code) fragte man sich, wie man die Übertragung noch effizienter gestalten kann. Der Manchester Code braucht für die Taktrückgewinnung doppelt soviel Bandbreite wie ein NRZ-Code mit gleicher Baudrate und für die Gleichspannungsfreiheit ist das Signal Bipolar ausgelegt.

Aber was wäre, wenn man anstatt nur an den Spannungspiegeln herumzuschrauben auch an den Datenbits geschraubt wird? Das man gewisse Signal-Zustände einfach nicht zulässt?

Dies wird mit dem **4B5B** und der **MLT-3** Codierung für 100 Mbit Ethernet erreicht. Wir verwerfen hier klar das OSI-Schichtenmodell und verbinden OSI Layer 1 und 2 untrennbar miteinander!

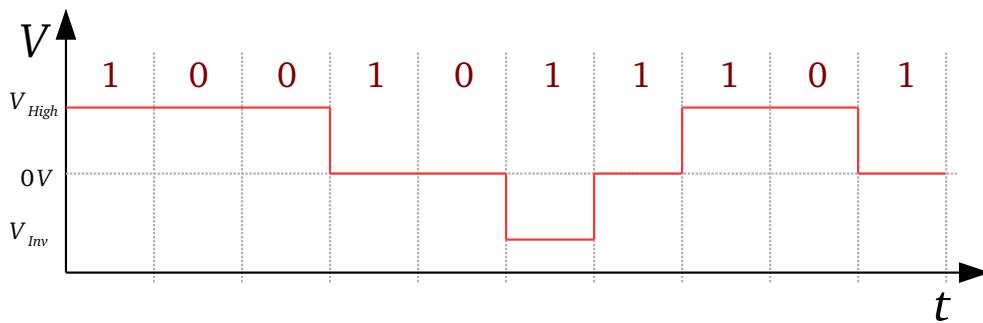


Abbildung 9.15: MLT-3 Codierung

Die MLT-3 Leitungscodierung ist wieder ein Bipolar Code und ändert den Pegel nur bei einem log. **1**. Bei einer log. **0** bleibt das Signal auf dem derzeitigen Spannungspiegel. Der Code braucht eine halb so grosse Bandbreite wie ein vergleichbarer NRZ-Code bei gleicher Baudrate.

Der MLT-3 ist weder gut in der Taktrückgewinnung (log. **0** Sequenzen geben keine Signalflanken) und auch nicht ideal für die Gleichspannungsfreiheit. (log. **0** Bitsequenzen können eine erzeugen Offset Spannung).

Aber wie realisieren wir trotzdem die Eigenschaften Tacktrückgewinnung und Gleichspannungsfreiheit?

Sehen wir uns den Bitstrom genauer an!

Der **4B5B** Leitungscode nimmt 4 Datenbits auf und sendet 5 Bits weiter.

Mit den 4 Datenbits können wir 16 verschiedene Bitmuster darstellen. Mit den 5 Bits können wir 32 Bitmuster darstellen.

Wir suchen uns jetzt alle Sendemuster mit vielen log.1 aus!

Diese weisen wir nun den verschiedenen Daten-Bitmuster zu. Damit können wir sicherstellen, dass genügend Flanken im Signal enthalten sind.

Der 4B5B Code ist folgendermassen definiert:

Bezeichnung	4B	5B	Funktion	Beschreibung	4B	5B	Funktion
0	0000	11110	Data 0	C	1100	11010	Data 12
1	0001	01001	Data 1	D	1101	11011	Data 13
2	0010	10100	Data 2	E	1110	11100	Data 14
3	0011	10101	Data 3	F	1111	11101	Data 15
4	0100	01010	Data 4	Q	-	00000	Quit
5	0101	01011	Data 5	I	-	11111	Idle
6	0110	01110	Data 6	J	-	11000	Start #1
7	0111	01111	Data 7	K	-	10001	Start #2
8	1000	10010	Data 8	T	-	01101	End
9	1001	10011	Data 9	R	-	00111	Reset
A	1010	10110	Data 10	S	-	11001	Set
B	1011	10111	Data 11	H	-	00100	Halt

Tabelle 9.2: 4B5B Codierung

Man erkennt in *Tabelle 9.2*, dass die Daten von 4 Bit auf 5 Bit aufgeblasen werden. Damit nimmt die Netto-Datenrate um 25% ab. Aber wir können die 5B Codewörter nun so aufbauen, dass wir den **MLT-3** Leitungscode, die **Gleichspannungsfreiheit** und **Taktrückgewinnung** unterstützen. Ausserdem können wir weiter Codesymbole einsetzen, welche wir nicht als Datensymbole, sondern als Steuersignale einsetzen können. Mit dem 4B5B Code eröffnen wir jede Daten-Übermittlung mit einem **JK** Symbol, dass ist das Start-Symbol des **4B5B** Encoders. Und am Abschluss wird ein **TR** Symbol angehängt, das Stop-Symbol.

Wir sehen also, dass wir mit der Verzahnung von Leitungscode und Bitstromcodierung ein effizienteres System aufbauen können. Der Bandbreitenbedarf ist 4 mal kleiner als beim Manchestercode, bei gleicher Symbolrate! Und dennoch können wir die Tecktrückgewinnung und Gleichspannungsfreiheit realisieren.

Das System ist aber «komplexer» geworden.

Wie bereits erwähnt wird der **4B5B** Code bei 100 Mbit Ethernet (100BASE-TX) eingesetzt.

9.8.3 8B10B

Der 8B10B Code ist eine Weiterführung des 4B5B Code Prinzips und wird in 1 Gbit Ethernet (1GBASE-TX) und USB 3.0 eingesetzt. Die Einbusse bei der Netto-Datenrate beträgt ebenfalls 25%.

9.8.4 64B66B

Bei **64B66B**, sie ahnen es bereits, ist die Weiterführung des **8B10B** Codes. Die Einbusse der Netto-Datenrate beträgt nur 3.125%. **64B66B** wird bei 10 Gbit (10GBASE-SR/LR/LRM/ER) / 100 Gbit Ethernet (100GBASE-xx) und Thunderbolt eingesetzt, Varianten wie die **128B130B** Codierung werden bei PCIe 3.0 und der **128B132B** Code wird bei USB 3.1 eingesetzt.

Aufgabe 9.44 : NRZI-Polar

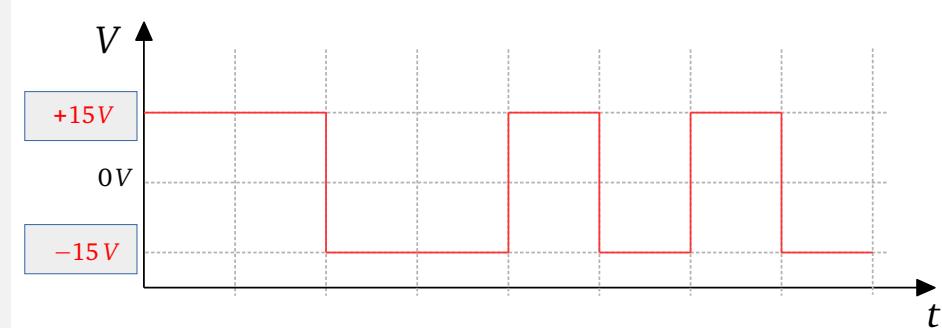
Frage

Zeichnen sie die folgende Sequenz im geforderten Format ein.
Falls nötig, zeichen Sie ebenfalls das Taktsignal ein.

- NRZI-Polar
- $V_{High} = 15 \text{ V}$, $V_{Low} = -15 \text{ V}$
- asynchron

Data : 00110101

Antwort



Aufgabe 9.45 : NRZ-Unipolar

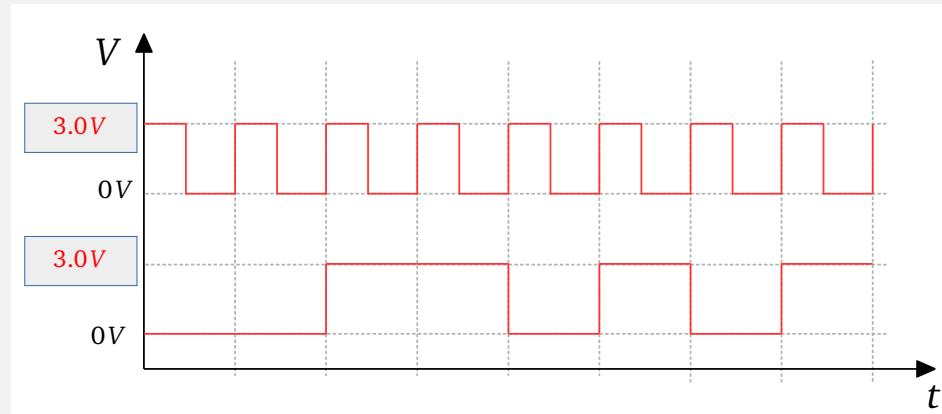
Frage

Zeichnen sie die folgende Sequenz im geforderten Format ein.
Falls nötig, zeichen Sie ebenfalls das Taktsignal ein.

- NRZ-Unipolar
- $V_{High} = 3.0 \text{ V}$, $V_{Low} = 0 \text{ V}$
- synchron

Data : 00110101

Antwort



10

Fehlererkennung

Inhalt des Kapitel

10.1 Kanalmodell	138
10.2 Grundlagen	139
10.2.1 Fehlerarten	139
single-bit-error	139
burst-error	139
10.2.2 Checksumme	140
10.3 Repetition Code	141
10.4 Parity-Bit	142
10.5 Fehlerhypothese	144
10.5.1 Coderate	144
10.6 Cycle Redundancy Check	145
10.6.1 Binäre Polynom Division	145
10.6.2 Generatorpolynom	146
10.6.3 Standards	146
10.6.4 Eigenschaften	147
10.6.5 Hardware-Umsetzung	147
10.6.6 Varianten	148
10.6.7 Vorteil	148
10.6.8 Nachteil	148

Zum Inhalt

Wir lernen in diesem Kapitel, wie wir Fehler in einem Datenpaket erkennen können. Zunächst werden wir die Theorie des Binär-symmetrischen-Kanals kennen lernen und danach die häufigsten Fehlererkennungs-Mechanismen erarbeiten.

Information, Redundanz etc...

Damit wir in diesem Kapitel kein durcheinander kriegen:

- Information / Infodaten : Sind die Daten, welche wir Übertragen wollen
- Checksumme : zusätzliche Daten, welche an die Infodaten angehängt werden
- Redundanz : Die Checksumme ist **redundant** zur Information
- Nachricht : Ist das gesamte Datenpaket mit Infodaten und Checksumme

10.1 Kanalmodell

In der Kommunikationstechnik haben wir leider den Fall, dass unsere Nachrichten während einer Übertragung gestört werden können. Meistens durch elektromagnetische oder leitungsgebundene Störungen aber auch bei schlechter Signalqualität (kleines SNR) durch das Rauschen.

Wenn wir dies auf einzelne Bits herunter brechen, haben wir den Fall, das z.B. eine log. **0** gesendet wird, aber eine log. **1** empfangen wird, oder umgekehrt. Dies nennen wir einen **Fehler**.

In der Informations-Theorie wird diese Eigenschaft, das generieren von Fehlern, dem Kanal zugeschrieben. Ein solcher Kanal für Binäre Daten wird dabei als sog. **Binär symmetrischen Kanal** bezeichnet.

Hierzu wird in der Informations-Theorie das sog. **Binär symmetrische Kanal Modell** verwendet (**binary symmetric channel model, BSK Model**)

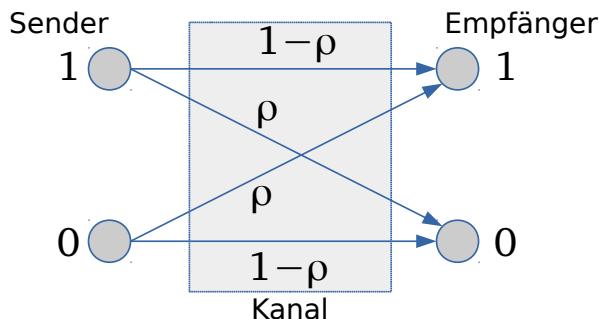


Abbildung 10.1: BSK Model : Binary Symmetric Channel Model

Wir haben dabei den Sender, welcher ein binäres Signal überträgt. Mit der Wahrscheinlichkeit $1 - \rho$ wird dieses richtig übertragen. Mit der Wahrscheinlichkeit ρ wird das Signal falsch empfangen.

$$\begin{aligned} \rho = 0 &: \text{Fehlerfreier Kanal (optimal channel)} & \rho \in [0, \dots, 1] \\ \rho = 0.5 &: \text{Zufälliger Kanal (random channel)} \end{aligned} \quad (10.1)$$

Das BSK Modell, (sowie auch ein realer Übertragungskanal) ist **gedächtnislos (memoryless)**. Damit ist gemeint, dass wenn Sie mehrere Bits über den Kanal senden, jedes Bit einzeln, unabhängig von den anderen, ein Fehler aufweisen könnte.

Grösse	Formelzeichen	Einheit
Fehler-Wahrscheinlichkeit	ρ	1

Tabelle 10.1: Formel-Zeichen / -Einheiten : Binär symmetrische Kanal Modell

Anmerkung Gerade diese Annahme ist in der Realität nicht ganz richtig. Erinnern wir uns an die sog. Gleichspannungsfreiheit eines Leitungscodes. Ein nicht Gleichspannungsfreier Code kann zu einer Offsetspannung führen und damit den Kanal verändern.

Wenn wir uns an Abbildung 2.8 erinnern : Wir heben die Spannung an, der low-Pegel wir kaum noch zuverlässig erkannt. Damit könnte man auch sagen : Der Kanal «erinnert» sich...

Was ist nun so interessant an einem gedächtnislosen Binär symmetrischen Kanal? Das BSK Modell entstammt eigentlich aus der Spieltheorie und wurde schon vor langer Zeit von Mathematikern untersucht. Die daraus entstandenen Fehlererkennungs- und Fehlerkorrektur-Verfahren basieren auf diesem Modell. Wir gehen nun nicht weiter auf das Modell ein, ich möchte das Sie wissen woher die Grundannahmen stammen und wie ein Bit-Fehler überhaupt definiert ist.

Claude Shannon erarbeitete Mithilfe des Kanalmodells heute unentbehrliche Grundlagen. Die genaue Berechnung möchte ich Ihnen ersparen, aber Schlussendlich konnte er Beweisen, dass man (abhängig von der Rate und vom S/N) eine Modulations- / Demodulations-Funktion definieren kann, die das ursprüngliche gesendete Signal wieder aus dem verrauschten Kanal errechnen kann. Mit diesem Beweis in der Hand hat man nun versucht möglichst ideale Modulation- / Demodulations-Funktionen zu finden. Damit war der Grundstein für die sog. **Quellen-Kodierer (engl. source-encoder)** gelegt. Heute wird dies als **forward error correction (FEC)** bezeichnet. Damit lassen sich nicht nur Fehler erkennen, sondern sogar korrigieren. Dies besprechen wir in Kapitel 13 : Fehlerkorrektur. Sind wird nur daran interessiert **Fehler zu erkennen** und bei zu vielen Fehlern die Daten erneut anzufordern, so nennt man dies **feedback error control** und gehört zur **Flusssteuerung**. Feedback error control ist in der Regel simpler umzusetzen. Die Flusssteuerung werden wir im Kapitel 16 : Flusssteuerung betrachten.

10.2 Grundlagen

10.2.1 Fehlerarten

single-bit-error

Wie wir bereits gesehen haben, ist der häufigste Grund für Fehler das Rauschen. Das BSK Modell bildet dies wie besprochen als Normalverteilte-Zufallsvariable ρ ohne Gedächtnis ab.



Abbildung 10.2: Single-Bit Error

Für uns heißt das, dass einzelne Bits **unkorrielt** invertiert empfangen werden. Wir nennen dies **single-bit-errors**.

burst-error

Eine unkorrelierte Fehler-**Sequenzen**, also mehrere Bits welche nach einem *unkorriierten Ereignis hintereinander falsch empfangen* werden, nennt man einen **burst-error**. Burst-error entstehen durch zufällige Störereignisse (zB. Umschalten Relais-Kontakt, Blitz, etc.) welche aber gleich mehrere Bits nacheinander stören.



Abbildung 10.3: Burst Error

Burst-error treten vor allem bei Drahtlosverbindungen aber auch bei leitungsgebundenen Störungen auf.

Info

Unsere Fehlerdetektion muss also *single-bit-* und *burst-error* erkennen können.

Wir widmen uns nun der interessanten Frage :

Wichtig

Wie können wir Fehler detektieren?

10.2.2 Checksumme

Wir übertragen auf unserem Kommunikationssystem (Info-) Daten hin und her. Diese sind im Grunde ja nichts einderes als eine einzelne grosse binäre Zahl. Wir gehen nun davon aus, dass unsere Infodaten jeden Wert in dieser Zahl annehmen kann.

Damit wir als Empfänger überhaupt erkennen können, ob die empfangenen Daten **nicht** verändert wurden, brauchen wir eine Checksumme, welche wir mit den Infodaten **gegenprüfen** können. Diese Checksumme wird über eine mathematische Funktion aus den Infodaten generiert und der Sender hängt die Checksumme beim Senden zusätzlich an die Infodaten an.

Vorgehen

In der Praxis wird dies wie folgt umgesetzt:

- **Sender**
 - Berechnet die Checksumme aus den Infodaten
 - Bildet aus Infodaten und Checksumme eine Nachricht
 - Versendet die Nachricht
- **Empfänger**
 - Trennt aus der empfangenen Nachricht die Infodaten und *empfangene Checksumme*
 - Berechnet Checksumme aus den empfangenen Infodaten
 - Vergleicht *berechnete Checksumme* mit *empfangener Checksumme*
 - Stimmen die Checksummen überein : alles okay!

Falls nicht, dann wurden entweder die Infodaten oder Checksumme bei der Übertragung verändert

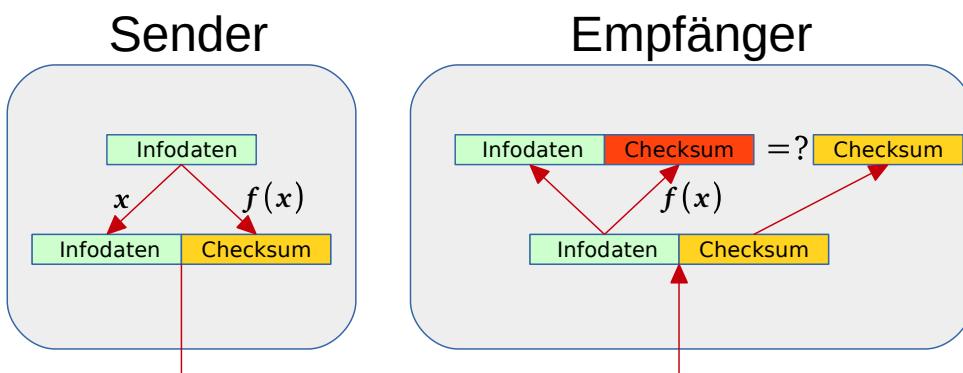


Abbildung 10.4: Theorie zur Fehlerdetektion

In Abbildung 10.4 sehen wir das Vorgehen als Grafik abgebildet.

Konsequenz

Die Fähigkeit Fehler zu detektieren bringt folgende Konsequenzen mit:

- Unsere Nachricht wird grösser, da wir neben den Infodaten auch die Checksumme übertragen.
- Der Sender, sowie der Empfänger kennen die *Mathematische Funktion* mit welcher die Checksumme berechnet wird und müssen diese für die Sende- / Empfangsdaten berechnen.
- Wir müssen entscheiden, was zu tun ist, wenn die Checksummen nicht übereinstimmen! (Siehe Kapitel 16 : Flusssteuerung)

10.3 Repetition Code

Die älteste Art und Weise und so ziemlich die schlechteste Variante ist die mehrfache Übertragung der Infodaten. Übertragen wir die Daten mehrfach können wir diese miteinander Vergleichen, sind sie nicht identisch ist ein Fehler aufgetreten.

Aber wir vergrössern die Datenmenge erheblich! Um ganzzahliges Vielfache der eigentlichen Infodaten. Der Repetition Code gehört übrigens in die Kategorie der **Fehlerkorrigierenden** Codes.

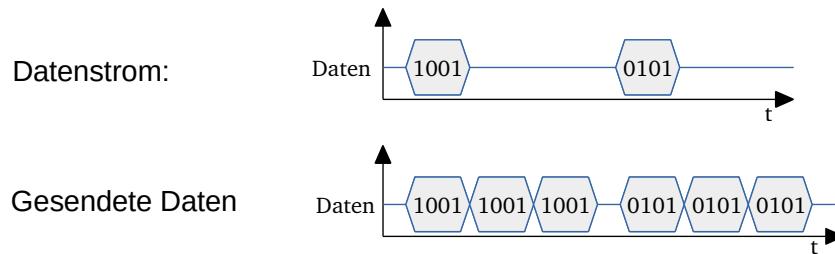


Abbildung 10.5: Beispiel : Repetition Code : Sendesignal

In Abbildung 10.5 ist zunächst der Datenstrom (Bits) abgebildet. Es sind die Infodaten die wir senden wollen. Danach sind die mit dem RepCode(3) gesendeten Daten abgebildet. Beim RepCode(r) werden die Infodaten r mal hintereinander übertragen.

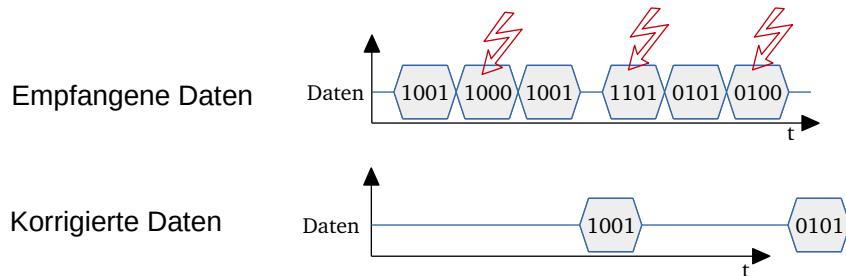


Abbildung 10.6: Beispiel : Repetition Code : Empfangssignal

In Abbildung 10.6 ist nun das Empfangssignal dargestellt. Wir haben mehrere Fehler beim Übertragen auf geschnappt. Bei der ersten Sequenz sehen wir das die [1001] Sequenz zwei mal übertragen wurde. Wir können also davon ausgehen, das mit hoher Wahrscheinlichkeit eine [1001] gesendet wurde. Bei der zweiten Zahl wurden zwei Fehler aufgeschnappt. Wir können aber Bit für Bit vergleichen und doch noch die zwei einzelnen Bitfehler bestimmen. Hier hatten wir gerade Glück das es gut geklappt hat. Wie bereits erwähnt bläht dieser Code die Datenmenge erheblich an.

Wir brauchen eine effizientere Variante Fehler zu detektieren!

Anmerkung

Verwenden Sie nie den Repetition Code! Es gibt bei weitem besserer Varianten. Er dient lediglich als Beispiel für die Fehler-Detektion (und für die spätere Fehlerkorrektur)

10.4 Parity-Bit

Eleganter ist das Mitsenden eines «Paritäts-Bit» (engl. Parity-Bit [W51]). Das Parity-Bit enthält die (binäre) Quersumme der gesendeten Daten.

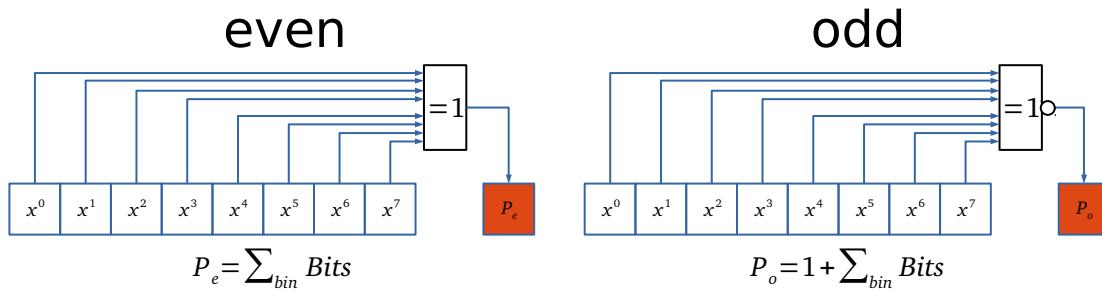


Abbildung 10.7: Parity-Bit : even / odd

Das Parity-Bit lässt sich leicht erzeugen und vergleichen. Aber es versagt bei einer geraden Anzahl von Fehlern.

	Data	P_e
original	0010 1001	1
received	0010 1001	1
calced Parity	0010 1001	1

	Data	P_e
original	0010 1001	1
received	0010 1101	1
calced Parity	0010 1101	0

	Data	P_e
original	0010 1001	1
received	1010 1101	1
calced Parity	1010 1101	1

Abbildung 10.8: Beispiel : even parity-bit (inkl. Versagen)

Die Parity-Bit Methode krankt daran, dass wir nur 1 Bit zur Überprüfung haben. Die Idee liegt nahe doch einfach mehr Parity-Bits anzufügen. Das funktioniert, wir nennen dies später einen Hamming-Code. Aber wir müssen eine andere *Mathematische Funktion* verwenden. Nur die Quersumme reicht nicht aus.

Vorteil

Einfach zu berechnen, schnell

Nachteil

Erkennt nur eine ungerade Anzahl Fehler. Gerade Anzahl Fehler bleiben unentdeckt.

Praxis Tipp

Merken Sie sich nur die Berechnung des Even-Parity Bit.
Und invertieren Sie das Ergebnis, falls das Odd-Parity gefragt ist.

Aufgabe 10.46 : Parity-Bit

Frage

Rechnen Sie die Parity-Bits der folgenden Bitsequenzen. Jeweils in even und odd Parity
[1001, 10001011, 10111001, 111001, 1, 0, 10, 1111111]

Antwort

even : 0, 0, 1, 0, 1, 0, 1, 0

odd : 1, 1, 0, 1, 0, 1, 0, 1

Aufgabe 10.47 : Parity-Bit

Frage

Die folgenden Daten sind jeweils 8 Bit lang und als LSB ist ein odd Parity Bit angefügt. Bei welchen Sequenzen wurden ein Einbit-Fehler eingeführt?

[100110011, 011100010, 101111010, 100111000, 111000000]

Antwort

[100110011, **011100010**, **101111010**, **100111000**, 111000000]

10.5 Fehlerhypothese

Wenn wir die Fehlererkennung wie bei dem Parity-Bit genauer betrachten, sollte uns bewusst werden, dass wir eigentlich «nur» eine Hypothese prüfen.

Beim Parity-Bit wissen wir, das wir nur eine ungerade Anzahl Fehler erkennen. Von allen möglichen Fehlervariationen können wir demnach nur die Hälfte abdecken. Für Kommunikationssysteme bei denen wir nur mit sehr weniger Fehlern rechnen, ist dies ausreichend. Aber was, wenn wir häufiger mit Fehler rechnen müssen? Wenn wir nun andere Verfahren für die Generierung der Checksumme verwenden, wie gut ist unsere Hypothese?

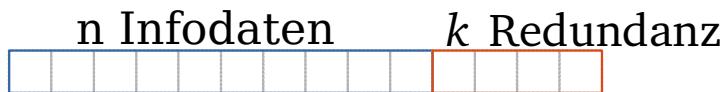


Abbildung 10.9: Bitstrom mit Infodaten und Checksumme / Redundanz

Wir betrachten einen (allgemeinen-) Bitstrom, bestehend aus Infodaten und der Checksumme. Wir haben genau n Bits Infodaten und k Bits für die Checksumme. Bei einer Checksumme mit k Bits Redundanz haben wir im besten Fall eine Wahrscheinlichkeit von $\rho_{error} = 2^{-k}$, dass die empfangenen Daten trotz Fehler als gültig erkannt werden. Man nennt dies auch einen **false-positive**.

Anders ausgedrückt heisst das auch : Mit einer Checksumme der Länge k können wir max. k Fehler im Datenwort (Infodaten und Checksumme) detektieren.

Eigenschaften der Checksummen Funktion

Wir haben eine gute Checksummen-Funktion gefunden wenn Sie die folgenden Eigenschaften aufweist:

- k single-bit-error erkennen können
- Einen burst-error der Länge k erkennen können
- Es muss egal sein ob die Fehler im Checksummen- oder Infodaten-Bereich auftreten
- Einfach in Software / Hardware implementierbar sein

Ob eine Funktion diese Eigenschaften aufweisen ist schwer zu beweisen.

Glücklicherweise haben Mathematiker sich an diesem Thema bereits ausgetobt! In der Mathematik hat man diese Funktionen mit sog. Galois-Feldern [W24] beschrieben. Wir werden dies vor allem im nächsten Kapitel antreffen.

10.5.1 Coderate

Wie wir gesehen haben, hängen wir unseren Datenbits n zusätzliche Redundanzbits k an. Unsere Nachricht als ganzes wird grösser.

$$\begin{aligned} n &:= \text{Anzahl Datenbits} \\ k &:= \text{Anzahl Redundanzbits} \\ n + k &:= \text{Anzahl Nachrichtenbits} \\ R_C &= \frac{n}{n+k} \end{aligned} \tag{10.2}$$

Das Verhältnis von der Anzahl Datenbits zur Gesamtzahl der gesendeten Bits, nennt man die **Coderate**.

10.6 Cycle Redundancy Check

Bei dem **cycle redundancy check** (CRC [W12]) wurde als mathematischen Funktion die *binäre Polynom Division* gewählt.

10.6.1 Binäre Polynom Division

Die Daten werden zunächst um ein Padding erweitert und durch ein sog. *Generator-Polynom* dividiert. Der *mathematische Rest der Division* ist unsere gesuchte Prüfsumme.

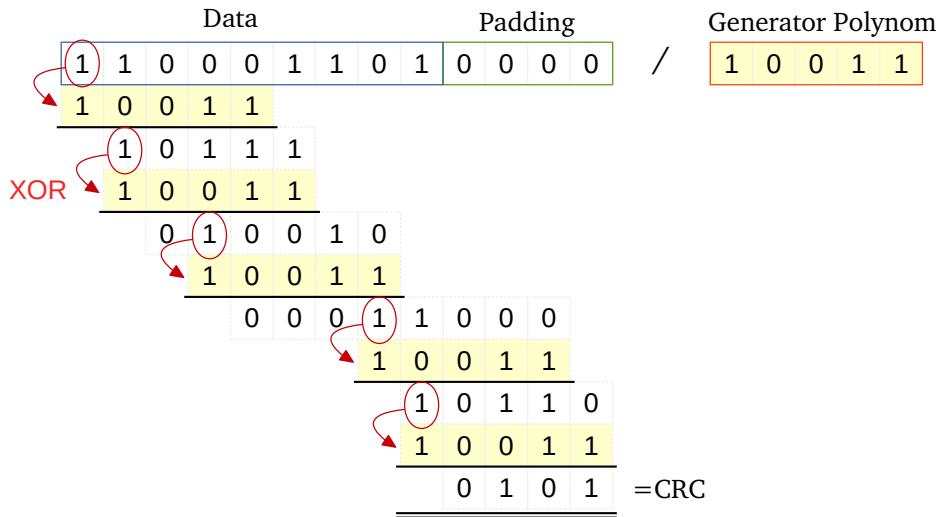


Abbildung 10.10: CRC Berechnung : CRC-4 mit $g(x) = x^4 + x^1 + 1$

In Abbildung 10.10 ist eine CRC Berechnung eines CRC-4 mit einem Generatorpolynom $g(x) = x^4 + x^1 + 1$ abgebildet. Unser Datenwort ist der **Dividend**, das Generatorpolynom ist der **Divisor** und unser CRC ist der **Rest**. Ein CRC wird mit dem **Generatorpolynom** beschrieben.

Wieso als Polynom? Die Mathematiker haben, wie bereits erwähnt, die Grundlagen des CRC's erarbeitet. Sie haben dies mit Galois-Felder getan. Diese werden als Polynome dargestellt.

CRC-Länge Ein CRC hat immer eine bestimmte Länge : Die Checksumme des CRC-4 hat eine Länge von 4 Bit. Die Checksumme eines CRC-32 hat eine Länge von 32 Bit.

Zero-Padding Das Padding des CRC ist immer gleich lang wie der CRC selbst. Bei einem CRC-4 also 4 Bit.

Generatorpolynom Man muss nun wissen: Das Generatorpolynom ist immer um 1 Bit länger als der CRC und hat an erster Stelle immer eine **1**. Wir sehen in Abbildung 10.10 das Polynom $g(x) = x^4 + x^1 + 1$. Man muss dies folgendermassen interpretieren: Das LSB (**1**) ist eine log. **1**, an der zweiten Stelle (x^1) ist eine log. **1**, und an der fünften Stelle (x^4) ist eine log. **1**. Alle anderen Stellen sind eine log. **0**.

Grad des Polynoms Mathematisch wird die Länge eines Polynoms mit dem «Grad» eines Polynoms angegeben. $\deg(y)$.

Bei $g(x) = x^4 + x^1 + 1$ ist der Grad $\deg(x^4 + x^1 + 1) = 5$

Selbststudium / Repetition

Die Eigenschaften eines Polynoms, die Polynomdivision und der Grad des Polynoms $\deg(f(x))$ sind im Selbststudium / Repetition Mathematik zu absolvieren!

10.6.2 Generatorpolynom

Das Generatorpolynom wird in einem Standard oder sonstigen Dokumenten meist in der Hexadezimal-Form dargestellt. Davon gibt es 3 Darstellungsformen die man vorfindet:

$$\begin{array}{lll} \text{normal : } g(x)_{hex} = 0x3 & = 0b0011 \Rightarrow x^4 + [0x^3 + 0x^2 + 1x^1 + 1] & (MSB - First) \\ \text{inverse : } g(x)_{hex} = 0xC & = 0b1100 \Rightarrow [1 + 1x^1 + 0x^2 + 0x^3] + x^4 & (LSB - First) \\ \text{inverse reziprok : } g(x)_{hex} = 0x9 & = 0b1001 \Rightarrow [1x^4 + 0x^3 + 0x^2 + 1x^1] + 1 & (\text{Koopman}) \end{array}$$

Da beim Generatorpolynom der höchstwertige Koeffizient immer eine log. **1** sein muss, wird er in der *normalen*- und *inversen*-Form nicht angegeben.

Bei der *inverse reziprok*-Form wird der niederwertigste Koeffizient nicht angegeben (dieser ist meist auch log. **1**).

In der *inversen*-Form wird das Polynom vom höchstwertigsten Koeffizienten MSB) an begonnen.

Wieso so viele Varianten? Wie wir noch sehen werden, kann ein CRC auf unterschiedliche Art und Weise in Hardware gebaut werden.

Die normal / inverse und inverse-reziprok Form ist jeweils für eine Architektur ausgelegt.

Wichtig

Sie werden nie selbst ein CRC Polynom erstellen müssen. Die CRC-Polynome sind standardisiert und normalerweise in der Hexadezimal-Form angegeben.

10.6.3 Standards

Nicht jedes beliebige Generatorpolynom hat die Eigenschaften, welche wir uns für die Checksumme wünschen. Und wie bereits erwähnt, sind diese schwer zu beweisen. Daher gibt es eine ganze Reihe normierten Generatorpolynome für unterschiedlich Längen.

Bezeichnung	Verwendung	Polynom		
		normal	invers	invers reziprok
CRC-5-USB	USB token packets	0x05	0x14	0x12
CRC-6-ITU	G.704	0x27	0x39	0x33
CRC-6-CDMA2000-A	CDMA2000	0x03	0x30	0x21
CRC-16-CCITT	X.25,V.41,Bluetooth	0x1221	0x8408	0x8810
CRC-32	ITU-T,V.42, Ethernet, MPEG-2, PNG, Gzip	0x04C11DB7	0xEDB88320	0x82608EDB

Tabelle 10.2: Verschiedene CRC-Generatorpolynome [W12]

10.6.4 Eigenschaften

Wir können anhand des CRC-Polynoms dessen Eigenschaften ablesen.

single-bit-error Enthält das Polynom mehr als 2 Koeffizienten, kann immer ein *single-bit-error* erkannt werden.

double-bit-error Es kann gezeigt werden, dass wenn $g(x) \neq p(x) \cdot (x^r + 1)$ für $r = [1, n+k]$, dann kann der CRC auch zwei unabhängige *bit-error* erkennen

odd-bit-error Ist nicht einfach zu beweisen, aber sofern sich $g(x) = p(x) \cdot (x + 1)$ ausklammern lässt, werden alle *odd-bit-error* erkannt.

burst-error Ein CRC mit der Länge k erkennt immer *burst-error* der Länge $r \leq k$ falls $g(x) \in x^0$ enthält. (ist normalerweise der Fall)

Prüfungsrelevanz

Sie müssen an der Prüfung nicht die Eigenschaften eines CRC-Polynoms bewerten können

Wir können also froh sein, haben sich bereits Mathematiker an diesem Thema die Zähne ausgebissen und «gute» CRC-Polynome standardisiert.

10.6.5 Hardware-Umsetzung

Der CRC wurde so entworfen das man in gut in Hardware umsetzen kann.



Abbildung 10.11: CRC Hardware-Umsetzung

Etwas was man in Digital-Hardware gut umsetzen kann sind Schieberegister und XOR Schaltungen. Der CRC basiert auf einem rückgekoppelten Schieberegister, mit XOR Verknüpfungen und lässt somit sehr einfach in Hardware umzusetzen.

In Abbildung 10.11 sehen wir einen CRC-16, genauer ein CRC-16-CCITT:

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

Die Boxen sind die Schieberegister und die Kreuz-Symbole sind die XOR. Die XOR sind dem Generator-Polynom entsprechend angeordnet.

1. Es werden alle Register am Anfang auf log. 0 gesetzt
2. Danach wird mit jedem Takt ein Bit der Infodaten in den CRC-Generator geschoben
3. Dies wird wiederholt, bis alle Infodaten in den CRC-Generator geschoben wurden
4. In den CRC-Schieberegistern ist nun unser gesuchter CRC vorhanden

10.6.6 Varianten

In Abbildung 10.11 haben wir nur eine Normal-Form Variante in Forwärtsrichtung gesehen. Wir können den CRC aber auch wie folgt umsetzen:

- Die Infodaten am Eingang x^0 einfügen.
- Das Schieberegister mit **1** initialisieren \Rightarrow reziprok
- Die Infodaten mit LSB-First hineinschieben \Rightarrow inverse-Form
- Die CRC-Checksumme ganz am Ende nochmals alle Bits invertiert werden

Praxis-Tipp

All diese Varianten können in der Praxis vorkommen und sind meist nur mit einer Fussnote in eine Standard erwähnt.

Passen Sie also auf, welche Variante von Ihnen gefordert wird!

10.6.7 Vorteil

Der CRC erkennt *burst-error* bis zu der Länge k Bits. *Single-bit* und meist auch *double-bit-error* werden erkannt. Je nach Generatorpolynom werden auch *odd-bit-error* erkannt. Der CRC ist einfach in Hardware implementierbar und ist schnell.

10.6.8 Nachteil

Die Wahrscheinlichkeit eines *false-positive* liegt bei 2^{-k} .

Single-bit-error $r > 3$ für $r = 2n, n \in \mathbb{R}_+$ werden nicht erkannt. Damit kann die Anforderung « k single-bit-error erkennen können» nicht erfüllt werden.

Aufgabe 10.48 : CRC-5-USB

Frage

Berechnen Sie den CRC aus dem nachfolgenden Bitstrom mit dem CRC-5-USB Generatorpolynom :

$$g(x)_{hex} = 0x05 \\ [100100101]$$

Antwort

Data	Padding	Generator Polynom
1 0 0 1 0 0 1 0 1	0 0 0 0 0 0	/ 1 0 0 1 0 1
1 0 0 1 0 1		
0 0 0 0 1 1 0 1 0 0		
1 0 0 1 0 1		
1 0 0 0 1 0		
1 0 0 1 0 1		
0 0 1 1 1 0 0	=CRC	

Aufgabe 10.49 : CRC-4**Frage**

Wir haben folgenden Bitstrom erhalten. Er enthält einen 4 Bit CRC am Ende (rechts). Das Generatorpolynom lautet $g(x)_{hex} = 0x09$. Wurde die Sequenz richtig übertragen oder sind Fehler im Bitstrom enthalten?

[1101110111110]

Antwort

Data	Padding	Generator Polynom
1 1 0 1 1 1 0 1 1 1	0 0 0 0	/ 1 1 0 0 1
1 1 0 0 1		
0 0 1 0 1 0 1		
1 1 0 0 1		
1 1 0 0 1		
0 0 0 0 0 0 0	=CRC	

Die

Sequenz wurde nicht richtig empfangen. Die CRC Prüfsummen stimmen nicht überein!

11

Topologie

Inhalt des Kapitel

11.1 Netzwerk Topologie	152
11.1.1 Bus	153
11.1.2 Stern	154
11.1.3 Linien	155
11.1.4 Ring	155
11.1.5 Baum	156
11.1.6 Masche	157
11.1.7 Hybride Formen	157
Beispiele	157
11.2 Anisotropie / Richtungsabhängigkeit	158
11.2.1 Simplex	158
11.2.2 Halb-Duplex	158
11.2.3 (Voll-) Duplex	159
11.2.4 Shared-Medium	160
Shared-Simplex	160
Shared-Halb-Duplex	161
Shared-Duplex	161
11.3 Netzwerkausdehnung	162
11.4 Fallbeispiele	162
11.4.1 I2C	162
11.4.2 RS-485	163

Zum Inhalt

In diesem Kapitel lernen wir die verschiedenen Netzwerk-Topologien kennen. Ausserdem lernen wir die Begriffe, welche in der Netzwerk-Technik verwendet werden um die Richtungsabhängigkeit einer Nachrichtenschnittstelle zu beschreiben. Und als kleine Ergänzung lernen wir die Begriffe für die Beschreibung einer Netzwerk-Ausdehnung kennen.

11.1 Netzwerk Topologie

Wir haben nun alle Grundsteine gelegt um ein Kommunikationssystem aufzubauen. Besprechen wir nun, was wir daraus aufbauen können.

Info

Verbinden wir mehrere Kommunikationsteilnehmer zusammen, so nennen wir dieses Gebilde ein **Netzwerk**.

Dieses kann auf verschiedene Arten miteinander verbunden werden.

Wir nennen dies eine **Netzwerk-Topologie**

Im Englischen wird ein Kommunikationsteilnehmer auch als **client** bezeichnet.

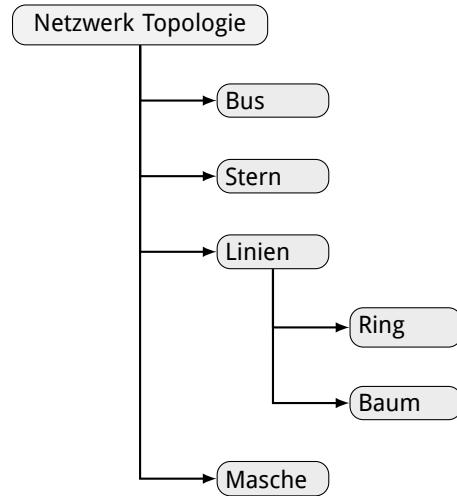


Abbildung 11.1: Netzwerk Topologie

In *Abbildung 11.1* sind die verschiedenen Netzwerk-Topologien abgebildet. Wir werden diese einzeln besprechen.

11.1.1 Bus

Die **Bus**-Topologie ist die einfachste Netzwerk Topologie. Alle Clients hängen am gleichen Übertragungsmedium. Dieser wird auch der **Bus** genannt.

Jeder Client kann die versendeten Daten mithören, welche auf dem Bus ausgetauscht werden. Damit die einzelnen Clients wissen, welche Nachricht für Sie bestimmt ist, werden die Daten mit mind. einer Empfangsadresse versehen. Nur wenn der Client mit der seiner Adresse angesprochen wird, nimmt dieser die Daten entgegen, sonst werden die Bus-Daten einfach ignoriert. Wie die Adressverwaltung realisiert wird, ist im Protokoll definiert.

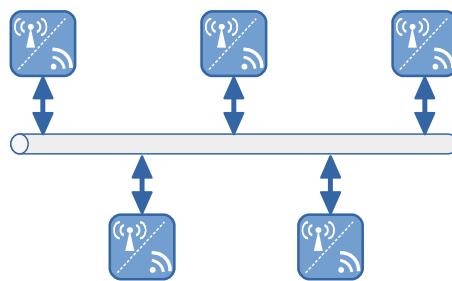


Abbildung 11.2: Bus Topologie

Vorteil

Alle Client hängen am gleichen Bus. Hat man Zugang zum Bus, hat man Zugang zu allen Daten auf dem Bus. Die Verkabelung ist einfach. Für einen neuen Client muss lediglich der Bus erweitert werden.

Nachteil

Hat man unbefugt Zugriff auf den Bus, kann man alle Daten abfangen. Das Protokoll muss dafür sorgen, dass sich die Clients nicht untereinander stören und den Bus nicht blockieren. Wird der Bus wegen eines defektes gestört, fällt das gesamte Netzwerk aus.

Voraussetzung

Die Treiber Schaltungen der einzelnen Clients müssen **deaktivierbar** oder **Kurzschlussfest** sein, so dass keine Kurzschlüsse auftreten können.

Somit können nur *Tristate-Treiber* oder *Open-Collector Treiber* für ein Bus-System verwendet werden.

11.1.2 Stern

Die **Stern**-Topologie hat im Zentrum einen zentralen-Teilnehmer, auch **Hub** genannt. Alle Clients sind mit nur diesem Hub verbunden.

Anmerkung

Mit Hub ist nicht ein Netzwerk Hub/Repeater gemeint

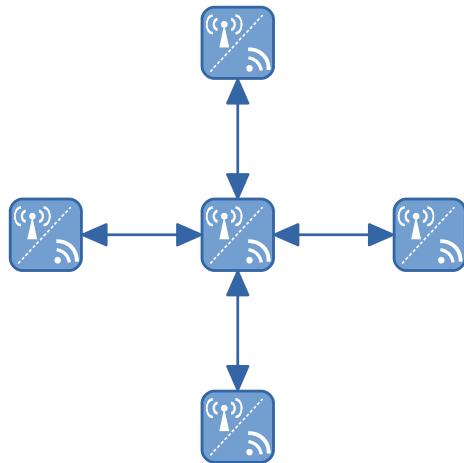


Abbildung 11.3: Stern Topologie

Vorteil

Der Verlust eines Client führt im *Rest* des Netzwerks zu keinen Problemen. In der Regel können Clients einfach angehängt und abgehängt werden. Die Vernetzung ist einfach. Nur der Hub hat Zugriff auf alle Daten.

Nachteil

Die Verkabelung ist aufwändig, da jeder Client mit dem Hub verbunden sein muss. Fällt der Hub aus, fällt das gesamte Netzwerk aus. Der Hub hat Zugriff auf alle Daten und ist dementsprechend ein sich lohnendes «Angriffs-Ziel».

Voraussetzung

Keine speziellen Anforderungen (Der Hub braucht einfach genügend Anschlüsse)

11.1.3 Linien

Die **Linien**-Topologie ist eine Serie-Schaltung der einzelnen Client. Wird ein zusätzlicher Client angefügt müssen alle Clients irgendwie darüber informiert werden. Dies muss im Protokoll definiert werden. Eine Nachricht wird von einem Client zum nächsten weiter gereicht, bis Sie ihr Ziel erreicht hat.

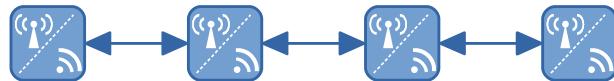


Abbildung 11.4: Linien Topologie

Vorteil

Der Verkabelungsaufwand ist gering. Es lassen sich einfach Clients aus dem Netz ausschliessen.

Nachteil

Wird ein Client aus dem Netz ausgeschlossen, sind auch die Clients hinter dem «Störenfried» getrennt. Bei einer defekten Verbindung spaltet sich das Netz in Teilnetze (Subnetze). Die Clients können zwar in Ihrem Subnetz kommunizieren, aber die anderen sind nicht mehr erreichbar.

Voraussetzung

Jeder Client braucht mind. 2 unabhängige Anschlüsse. (Ausser die «Rand->Client)

11.1.4 Ring

Die **Ring**-Topologie ist im Grunde eine geschlossene Linien Topologie. In der Regel ist jeder Client gleichberechtigt. Alle Client können von jedem anderen Client erreicht werden, indem die Daten über einen oder mehrere Client geroutet wird.

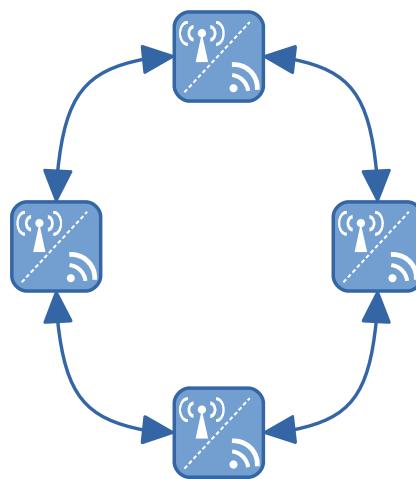


Abbildung 11.5: Ring Topologie

Vorteil

Bei einem geschlossenen Ring kann ein Client ausfallen ohne das der Rest des Netzwerks ausfällt. Die Verbindungen können eleganter verlegt werden. Der gesamte Datendurchsatz kann höher sein als bei einer vergleichbaren Bus-Verkabelung.

Nachteil

Fällt mehr als ein Client aus, sind alle Clients hinter den Störstellen nicht mehr erreichbar, das Netz wird in Subnetze unterteilt. Das Protokoll muss die Zugriffsreihenfolge steuern. Das Protokoll muss verhindern, dass die Datenpakete im Netzwerk rotieren. Jeder Client ist mit zwei anderen Stationen verkabelt, daher ein hoher Verkabelungsaufwand.

Jeder Client welche Daten weiter routen muss, hat Zugriff auf die Daten.

Voraussetzung

Jeder Client braucht mind. 2 unabhängige Anschlüsse

11.1.5 Baum

Die **Baum**-Topologie kennt eine sogenannte **Wurzel (root)**. Der Baum weist Verzweigungen auf, diese werden als **nodes** bezeichnet. Vom **root** aus können alle Client erreicht werden. Will man Daten von einem Ast auf einen anderen Ast senden, muss über die Nodes (max. zur **root-node**) durch geroutet werden. Im Grunde ist die Baum Topologie eine **Linien- mit Stern-**Topologie.

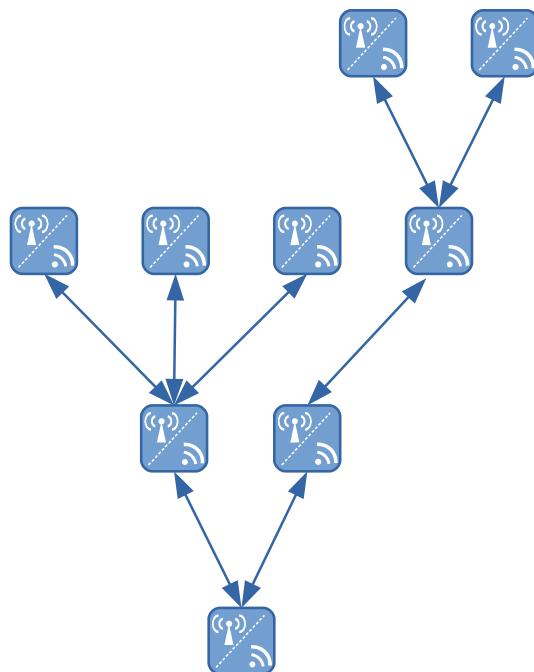


Abbildung 11.6: Baum Topologie

Vorteil

Einzelne Subnetze lassen sich leicht vom Netzwerk abkoppeln. Bei der Netzplanung lassen sich stark beanspruchte Äste zusammenfügen, so dass das übrige Netzwerk nicht überlastet wird.

Nachteil

Der **root-node** ist besonders kritisch. Fällt dieser aus, ist ein Grossteil des Netzwerks separiert.

Voraussetzung

Keine speziellen Anforderungen. Jeder Client muss mind. 2 Anschlüsse haben, ein Node mind. 3 und Endpunkt nur 1 Anschluss.

11.1.6 Masche

Die **Maschen-Topologie** (engl. **mesh**) ist nach dem Ansatz erstellt, dass jeder Client mit jedem anderen Client verbunden ist. Es wird in der Netzwerktheorie als **ideales Netz** bezeichnet, da ein Ausfall eines beliebigen Clients die anderen Netzwerkteilnehmer nicht beeinflusst. Aber in unserem Alltag sind grössere Netze nicht realisierbar, da die Anzahl Verbindungen quadratisch mit der Anzahl Clients wächst.

$$n_v = \frac{n_c \cdot (n_c - 1)}{2} \quad \begin{array}{l} n_c = \text{Anzahl Client} \\ n_v = \text{Anzahl Verbindungen} \end{array} \quad (11.1)$$

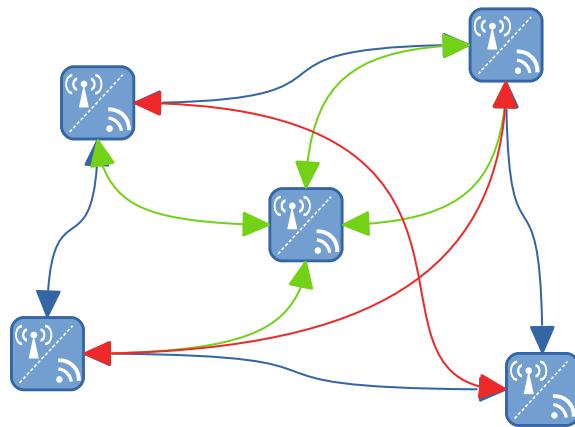


Abbildung 11.7: Mesh Topologie

Vorteil

Höchstmögliche Ausfallsicherheit

Nachteil

Bei grösseren Netzen unrealistischer Verkabelungsaufwand.

Voraussetzung

Jeder Client hat zwischen $1, \dots, \infty$ Anschlüsse

11.1.7 Hybride Formen

Wie wir schon bei der Baum Struktur gesehen haben, lassen sich Netze auch aus verschiedenen Grundformen zusammensetzen. Hier kurz erwähnt sind eine Stern-Ring Topologie, also ein Ring aus Hubs, an denen die Clients sternförmig verbunden sind. Der Ring wird dabei als Multistation Access Unit (MAU) bezeichnet und findet man in Serverfarmen. Auch kurz erwähnt sei die double-ring Topologie, bei der man ein Netzwerk aus einem Äusseren und Inneren Ring aufbaut. Dies wird häufig im Backbone-Bereich von Serverfarmen verwendet.

Beispiele

Im LAN Bereich (Maschinen- / Gebäudevernetzung) dominiert heutzutage die Baum-Topologie.

Im WAN Bereich (Haus zu Fernverteiler) dominieren die Stern-Topologie. Dies ist vor allem der Ausfallsicherheit und der historisch gewachsenen Telefonverkabelung geschuldet.

Ebenso im GAN Bereich. Hier ist die Ausfallsicherheit der entscheidende Faktor.

In den Geräten selbst, werden die einzelnen Chips meist mit Bus-Topologien verbunden, da hier die Anzahl Pins am Chip-Package, sowie der «Verkabelungsaufwand» die limitierenden Faktoren darstellen.

Was die Begriffe LAN, WAN, GAN bedeuten, sehen wir in Kapitel 11.3 : Netzwerkausdehnung.

11.2 Anisotropie / Richtungsabhängigkeit

Als nächstes gehen wir näher auf die einzelnen Verbindungen zwischen den Kommunikationsteilnehmer ein.

11.2.1 Simplex

Man nennt eine Verbindung Simplex fähig, wenn man entweder nur senden oder nur empfangen kann. Diese Verbindungsart wird auch *one-way communication* genannt.



Abbildung 11.8: Simplex

Ein typisches Beispiel ist hier die Radio- oder Fernsehübertragung. Das Fernsehsignal wird ausgesendet, egal wer zu sieht. Wird in der Automation ein einfacher Schrittmotor ohne Drehwinkelgeber angebunden ist dies eine typische Simplex Verbindung. Sie senden die Steuerpulse und gehen davon aus, das diese richtig empfangen und ausgesteuert wurden.

Vorteil

Sie müssen die Daten abschicken und können den Rest vergessen. Dies ist allerdings auch der grösste Nachteil.

Nachteil

Sie wissen nicht ob der Empfänger ihre Nachricht erhalten hat. Sie haben keine Ahnung ob der Empfänger die Nachricht auch richtig (ohne Fehler) erhalten hat.

11.2.2 Halb-Duplex

Man nennt eine Verbindung Halb-Duplex fähig, wenn man **abwechselnd entweder nur senden oder empfangen kann, aber nie gleichzeitig**.

Ein Halb-Duplex System nennt man auch eine *Full-Duplex emulation* oder *two-way alternate communication*.

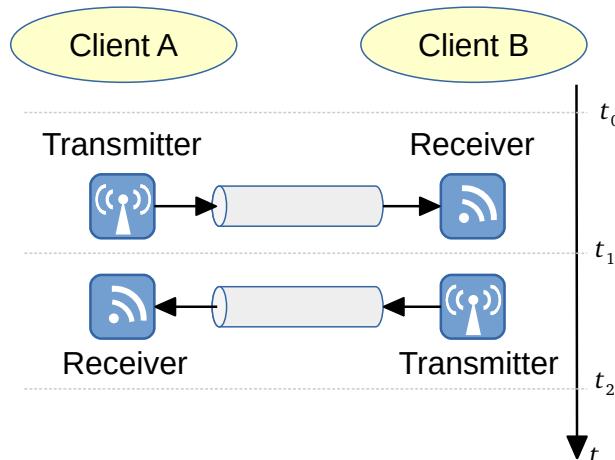


Abbildung 11.9: Halb-Duplex

Ein typisches Beispiel für eine Halbduplexverbindung ist zB. USB, Walkie-talkie oder eine (Tür-) Gegen sprechanlage. In der Automation ist das Abfragen eines Sensorwerts in der Regel eine Halbduplexverbindung. Dieses wird in der Regel im sog. Master-Slave Prinzip angesprochen. Zuerst wird der Slave (Sensor) vom Master (Controller) angefragt die Daten zu übertragen und danach schaltet der Master in den Empfangsmodus und empfängt die Daten des Sensors.

Vorteil

Die Daten können nicht nur gesendet, sondern auch Daten empfangen werden. Der (initiale-) Sender kann sich die Daten bestätigen lassen, so dass dieser überprüfen kann, ob die Daten richtig empfangen wurden.

Nachteil

Es steht bei einer Halbduplex eine verminderte Datenrate zur Verfügung, da diese nun zum Senden und Empfangen genutzt wird. Es treten Zeitverzögerungen zwischen Anfrage und Antwort an, je nach Auslastung des Mediums. Das Protokoll um mit Halbduplexverbindung zu kommunizieren ist aufwändiger. Es können **Kollisionen** auftreten.

11.2.3 (Voll-) Duplex

Man nennt eine Verbindung **(Voll-) Duplex** fähig, wenn man *gleichzeitig Senden und Empfangen* kann.



Abbildung 11.10: (Voll-) Duplex

Das typische Beispiel einer Vollduplexverbindung ist 100 Mbit/s LAN (10 Mbit/s war meist noch Halb-Duplex). Das Netzwerkkabel weist dabei dedizierte Sende- und Empfangs-Kabelpaare auf. Man mag nun behaupten, dass dies eigentlich zwei Simplex Verbindungen sind. Und da haben sie recht.

Vorteil

Sie können gleichzeitig Senden und Empfangen.

Es treten kleinere Antwortzeiten (**round-trip delay-time**) auf als bei einer Halb-Duplex Verbindung.

Nachteil

Sie müssen auf irgend eine Art und Weise einen Kanal duplizieren. Entweder durch **physikalische Aufsplittung** oder indem Sie die zur Verfügung stehende «**Bandbreite**» aufteilen. Das Protokoll ist weniger anspruchsvoll als eine Halb-Duplex Verbindung.

Beispiel UART

Wir haben ebenfalls gesehen wie wir ein System aus zwei RS-232 Geräten zusammen Verbinden können:

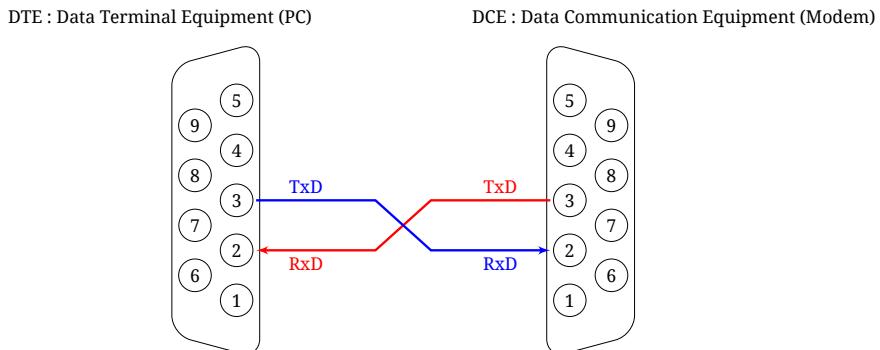


Abbildung 11.11: RS-232 : Verkabelung

In Abbildung 11.11 sehen wir die beiden Signale, jeweils für den Hin- und den Rück-Weg der Kommunikation. Da es separate unabhängige Hin- und Rück-Leiter gibt, handelt es sich um eine (Voll-) Duplex Verbindung.

11.2.4 Shared-Medium

Wir kennen nun Simplex, Halbduplex und Vollduplex Verbindungen. Wenn wir mehr als 2 Teilnehmer haben, müssen wir mit dem Protokoll dafür sorgen, dass der richtige Teilnehmer ausgewählt wird und die Daten übertragen werden. In der Regel werden den Teilnehmern Nummern vergeben, so dass sich die Teilnehmer identifizieren lassen. zB. Telefonnummer oder IP-Adresse.

Wie so eine Adresse aussieht werden wir im Kapitel 12 : Dataframe besprechen.

Shared-Simplex

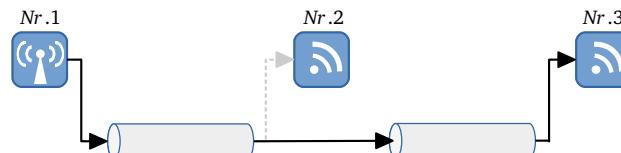


Abbildung 11.12: Shared Medium : Shared-Simplex

Beim Shared-Medium : Simplex, können Daten wieder nur in eine Richtung übertragen werden. Die nicht adressierten Teilnehmer ignorieren die Nachricht auf dem Shared-Medium. Nur der adressierte Teilnehmer verarbeitet die Daten.

Shared-Halb-Duplex

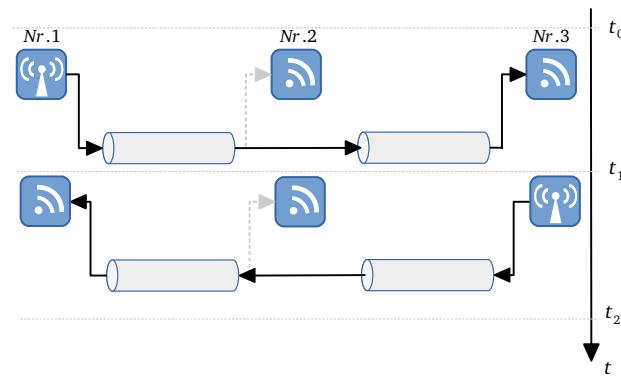


Abbildung 11.13: Shared Medium : Shared-Halb-Duplex

Beim Shared-Medium : Halbduplex Verfahren wird wie gehabt, entweder gesendet oder empfangen. Das Protokoll muss in diesem Fall noch besser handhaben, wann ein Teilnehmer sendet, da Kollisionen oder im schlimmsten Fall auch Kurzschlüsse auftreten können, falls ein Teilnehmer eine Nachricht missinterpretiert. (zB. **listen-before-talk**).

Shared-Duplex

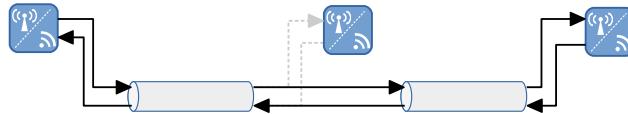


Abbildung 11.14: Shared Medium : Shared-Duplex

Beim Shared-Medium : Vollduplex wird der Kanal wieder aufgesplittet und es kann gleichzeitig gesendet oder empfangen werden. Auch hier gilt, dass das Protokoll verhindern muss, dass ein nicht angesprochener Teilnehmer gleichzeitig mit anderen Teilnehmer sendet.

Vorteil

Der Vorteil eines Shared-Medium ist der Preis (Bei Funk, WLAN, Drahtlos etc. ist man Alternativlos. Es gibt schlichtweg nur ein Medium das man nutzen kann). Es muss nur ein Medium zwischen verschiedenen Teilnehmern aufgebaut werden. (Verschieben der Komplexität von der Hardware zur Software)

Nachteil

Sind viele Teilnehmer am gleichen Shared-Medium angebunden, kann es schnell dazu führen, dass das Netzwerk überlastet wird. Schlussendlich kann nur immer 1 Teilnehmer auf das Shared-Medium schreiben.

11.3 Netzwerkausdehnung

Kommunikationssysteme lassen sich auch nach ihrer Netzwerk Ausdehnung kategorisieren:

Bezeichnung		Ausdehnung	
PAN	Personal Area Network	< 10 m	Griff-nähe
LAN	Local Area Network	< 1 km	Gebäude
MAN	Metropolitan Area Network	< 10 km	Stadt
WAN	Wide Area Network	< 1000 km	Land
GAN	Global Area Network	> 1000 km	Planet

Tabelle 11.1: Netzwerk Ausdehnung

11.4 Fallbeispiele

Sehen wir uns als Fallbeispiele I2C und RS-485 an.

11.4.1 I2C

I2C ist ein serieller, synchroner Übertragungs-Standard für Bus-Topologie und wird auch als *TwoWire-Interface* bezeichnet. Er besteht aus einer Daten und einer Takteleitung. (excl. Ground Verbindung)

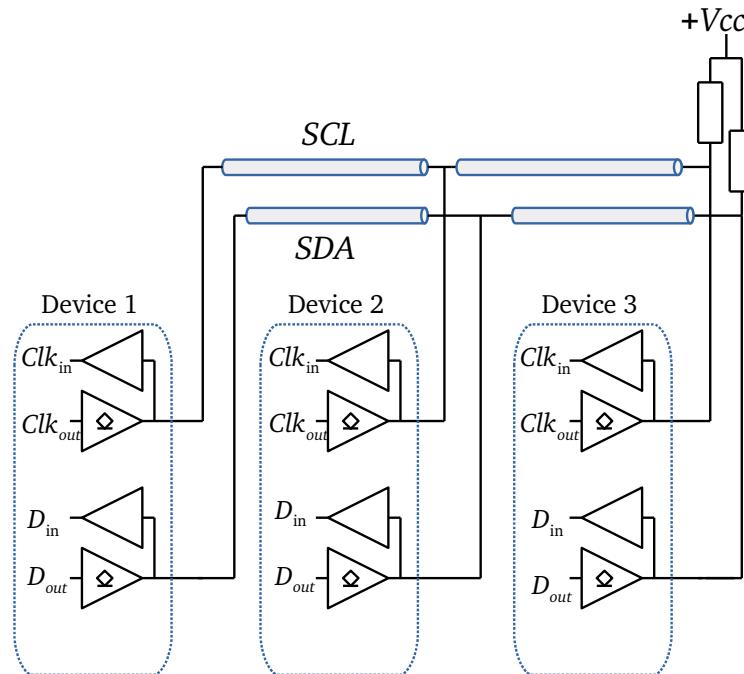


Abbildung 11.15: Beispiel I2C

In Abbildung 11.15 ist eine Beispielschaltung abgebildet. Wir sehen zum einen, dass die Daten- und Takt-Leitungen jeweils zusammen gehängt wurden. Daraus lässt sich zum einen schliessen, dass es sich um eine **Bus**-Topologie handelt und zum anderen das dieses System entweder eine Simplex oder ein Halb-Duplex Verbindung aufbauen können. (In der Tat ist I2C Halb-Duplex fähig) Das Symbol bei den Treiberbausteinen und die Pull-Up Widerstände sagt uns, dass es sich hierbei um Open-Collector oder Open-Drain Treiber handelt. Außerdem sehen wir an allen Pins, das pro Pin jeweils ein Treiber und eine Detektor angehängt ist. Es handelt sich somit um *bidirectional-Pins*. Ein weiteres Indiz dafür, dass es sich hierbei um eine Halb-Duplex System handelt.

Was wir nicht sehen : Abschlusswiderstände! I2C ist impedanzmässig nicht abgeschlossen. I2C Verbindungen können also nicht sehr lange gezogen werden. (Wenige Meter, mit *Range-Extender* bis ~ 20 m)

11.4.2 RS-485

Der RS-485 ist ein serieller, differentieller, asynchroner Übertragungs-Standard. Er bildet die Grundlage für den sog. **Profibus DPV0**.

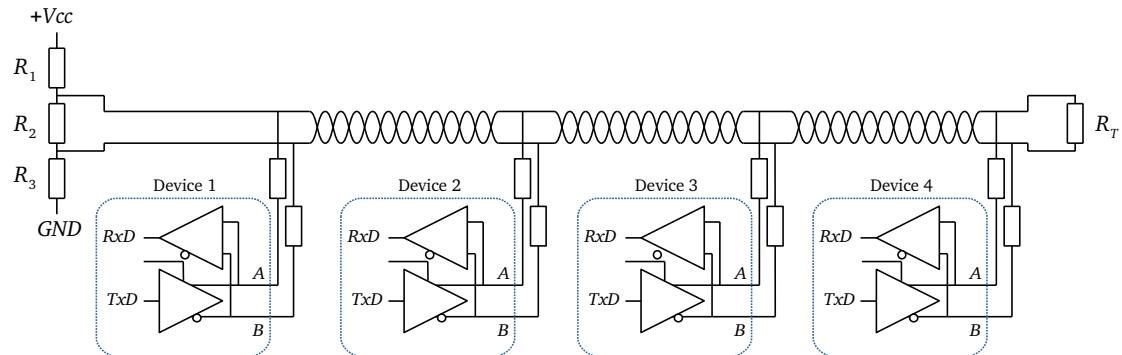


Abbildung 11.16: Beispiel RS-485

Die einzelnen RS-485 Devices gibt es in Halb-Duplex und Voll-Duplex Varianten. In Abbildung 11.16 ist die Halb-Duplex Variante abgebildet. Wie der Name schon sagt, handelt es sich hierbei ebenfalls um eine **Bus-Topologie**. Wir sehen pro Device nun zwei Pins. Aber es ist jeweils ein differentieller Treiber und eine differentieller Detektor angebracht. Es handelt sich also wieder um ein *bidirectional*-Pins. Die Treiber haben ein Enable Signal eingezeichnet. Es handelt sich daher um Tristate-Treiber. Wir sehen ausserdem, dass die Drähte «verwoben» dargestellt sind. Das heisst, dass wir ein Twisted-Pair Kabel verwenden. Ausserdem sehen wir am linken Rand ein Widerstandsnetzwerk und rechten Rand einen Abschlusswiderstand. RS-485 wird demnach impedanzmässig abgeschlossen. Wir können damit sagen, dass RS-485 gut für weitere Distanzen geeignet ist. In der Tat lassen sich Kabellängen von über 1.2 km mit RS-485 abdecken. (Allerdings mit reduzierter Datenrate)

12

Dataframe

Inhalt des Kapitel

12.1 Datenfluss	166
12.1.1 Asynchron	166
12.1.2 Isochron	167
12.1.3 Plesiochron	167
12.1.4 Synchron	168
12.2 Dataframe	168
12.2.1 Felder	169
12.3 Adressierung	170
12.4 Dataframe Beispiel Implementationen	172
12.4.1 UART Dataframe	172
12.4.2 I2C Frame	172
12.4.3 Ethernet Frame	173
12.5 Bit-Stuffing	174

Zum Inhalt

In diesem Kapitel besprechen wir, wie wir einen **Dataframe** zusammenstellen und was alles in diesen hineingehört. Außerdem untersuchen wir bekannte Schnittstellen, wie diese ihren (OSI-Layer 2) Dataframe zusammensetzen.

12.1 Datenfluss

Bevor wir jetzt aber direkt in die Materie einsteigen, müssen wir noch unser Vokabular weiter ausbauen. Wir haben im Kapitel 9.1.1 : Takt-Synchronisierung gesehen, dass wir für die Datentakt-Synchronisierung Begriffe wie **synchron** und **asynchron** verwendet haben.

Wichtig

Beim Datenfluss tauchen die Begriffe **synchron** und **asynchron** wieder auf, haben aber eine andere Bedeutung.



Übertragungssysteme lassen sich dadurch charakterisieren, wie die Daten zeitlich übermittelt werden. Wir nennen die Überkategorie hierfür etwas schwerfällig **Datenfluss**. Wie wir sehen werden, gibt es Systeme welche nicht die ganze Zeit am senden sind und andere welche kontinuierlich Daten versenden. Wir können diese in die folgenden Gruppen unterteilen:

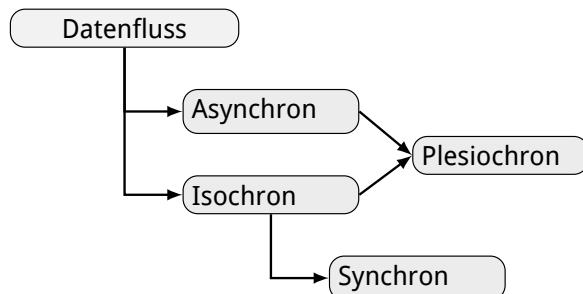


Abbildung 12.1: Übersicht Datenfluss

12.1.1 Asynchron

Werden die Datenpakete in undefinierten Zeitabschnitten übertragen spricht man von einer asynchronen Datenübertragung.

Start / Stop

In der Regel wird bei der Asynchronen Übertragung zusätzlich eine Start- und Stop-Sequenz angehängt.

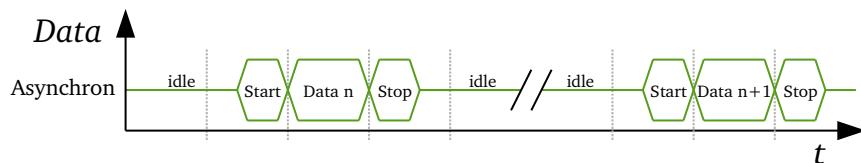


Abbildung 12.2: Asynchron

Beispiele hierzu ist das RS-232 Protokoll oder aber auch Ethernet, ISDN-S0 etc.

12.1.2 Isochron

Werden die Datenpakete in definierten Zeitabschnitten gesendet, nennt man dies eine isochrone Datenübertragung. Die Zeitabschnitte sind für eine Isochrone Übertragung immer gleich gross.

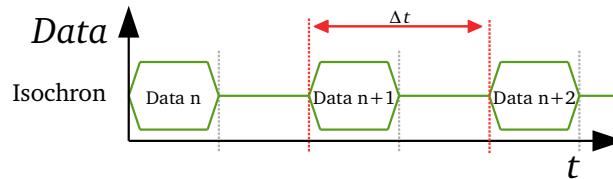


Abbildung 12.3: Isochron

Die Isochrone Datenübertragung wird vor allem in der Echtzeit-Datenverarbeitung eingesetzt.

Zeitintervall

Das Zeitintervall Δt zwischen zwei Isochronen Übertragungen nennt man auch die **Isochrone-Zeiteinheit**.

12.1.3 Plesiochron

Plesiochrone Datenübertragung ist eine quasi-Isochrone Datenübertragung. Dies ist eine asynchrone Datenübertragung, die über ein Protokoll gesteuert, sich nach «außen» isochron verhält.

Anforderung

Hierzu muss die asynchrone Datenübertragung schneller als die Isochrone-Zeiteinheit sein.

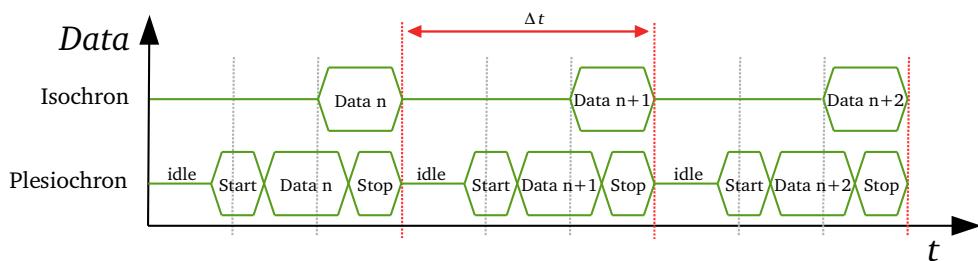


Abbildung 12.4: Plesiochron

Plesiochrone Datenübermittlung wird zB. als Protokoll Ergänzung bei *Ethernet* und *Real-Time Profinet* eingesetzt.

Aber Achtung: auch wenn man eine plesiochrone Datenübertragung per Protokoll-Ergänzung an ein asynchrones System anheften kann, so ist das System noch nicht garantiert Plesiochron! Für ein wirkliches Plesiochrones-Echtzeitsystem muss man dafür sorgen (mit der Netztopologie, Router Protokoll etc.), dass das System die Einhaltung der Isochronen Zeiteinheit **garantieren** kann.

12.1.4 Synchron

Der Begriff synchroner Datenfluss ist nicht ganz korrekt ausgedrückt. Besser wäre die Bezeichnung kontinuierlicher Datenstrom.

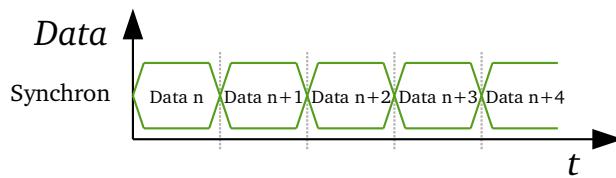


Abbildung 12.5: Synchron

Beim synchronen Datenfluss werden **immer** Daten mit gleicher Länge gesendet. Eigentlich ist es eine Untergruppe der Isochronen Datenübertragung, bei der die Synchrone Zeiteinheit Δt gleich der Sendedauer entspricht.

12.2 Dataframe

Wie wir bereits in *Abbildung 4.7* gesehen haben, erstellt der OSI-Layer 2 aus den *Datagram* des OSI-Layer 3 ein sog. **Dataframe**.

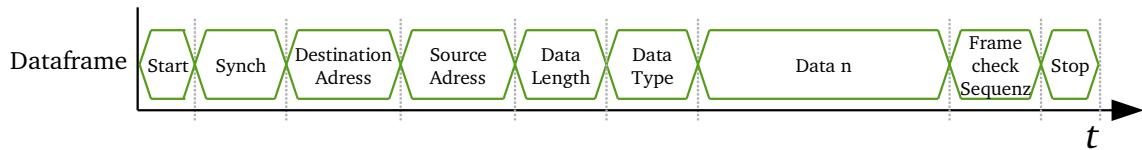


Abbildung 12.6: Allgemeiner Dataframe

In *Abbildung 12.6* ist so ziemlich das allgemeinste Dataframe abgebildet, wie es in «Vollausstattung» vom *Media Access Control* generiert wird.

Info

Merken Sie sich: nicht jeder Standard spezifiziert genau solch ein Dataframe
Je nach Standard oder OSI-Layer 1 Definition sind gewisse Felder gar nicht
nötig. Auch die Reihenfolge entspricht nicht zwingend dem abgebildeten Bei-
spiel.

Wir sehen hier einzelne sog. Felder : zB. Start, Synch, Destination Adress etc. Je nach Standard werden Felder aus dem Dataframe bereits in den höher liegenden Schichten in die Daten eingebettet.
(Manchmal sind diese sogar doppelt enthalten ☺)
Aber jedes Feld dient einem bestimmten Zweck!

Datenfeld	Zweck
Start / Stop	Asynchron- / Plesiochrone Übertragung (Datenfluss)
Synch	Asynchrone Verbindung (Datentakt)
Destination / Source Adress	Addressierung (Shared-Medium / Layer 2 Routing)
Data-Length	Grösse des Empfangsbuffers / Fehlerhäufigkeit
Data-Type	Zugriffsverfahren / Coexistence
Frame-Check-Sequenz	Fehlerhäufigkeit / Zugriffsverfahren

Tabelle 12.1: Felder im Dataframe - Zweck

12.2.1 Felder

Start / Stop Die Start und Stop Felder haben wir bereits bei RS-232 / UART angetroffen. Die Start-/Stop Symbole brauchen wir in Asynchronen- oder Plesiochronen-Systemen damit wir erkennen, wenn ein Datenframe anfängt / beendet wird.

Synch Das Synch Feld brauchen wir für die Synchronisierung der internen Baudratengeneratoren bei Asynchronen Verbindungen, welche eine Taktrücksynchroneisierung benötigen und nur durch eine zusätzliche Trainingssequenz aufsynchronisiert werden können. (zB. Ethernet / USB / Mobilfunk)

Destination- / Source-Adress In einem System mit mehreren Teilnehmern, müssen die einzelnen Teilnehmer voneinander unterscheidbar sein. Diesen wird normalerweise eine sog. MAC Adresse zugewiesen. Address-Felder brauchen wir natürlich nur, wenn wir mehrere Clients in einem Netzwerk betreiben.

Data Length Das Data-Length Feld brauchen wir, wenn wir zum einen unterschiedlich lange Datensätze übermitteln wollen und zum anderen, wenn wir aufgrund der Fehlerhäufigkeit die Datensatz-Länge variieren wollen (Kurze Datensätze weisen weniger Fehler auf, als lange Datensätze. Haben aber mehr «Overhead»).

Data-Type Das Datentyp-Feld brauchen wir um zu unterscheiden, ob die gesendeten Daten für den Empfänger OSI-Layer 2 oder 3 bestimmt sind. Geben wir hier ein Beispiel : Bei einem *Token-Passing* Zugriffsverfahren müssen wir den Token versenden können. Für diesen Fall ist im Datentyp-Feld der *Token* enthalten und im Datenwert den Wert des Tokens. Ist im Datentyp-Feld der Wert *Daten* eingetragen, werden die Daten im Datenfeld an den OSI-Layer 3 weiter gereicht.

Was genau Token-Passing ist werden wir im Kapitel *Kapitel 14 : Zugriffsverfahren* besprechen.

Frame-Check-Sequence Die Frame-Check-Sequence enthält die Checksumme für die Fehlerprüfung oder Fehlerkorrektur.

12.3 Adressierung

Die Adressierung der Teilnehmer ist unterschiedlich geregelt.

Ethernet IEEE802

Bei Ethernet (IEEE 802) sind alle Geräte mit einer sog. MAC Adresse versehen. (media access control address) Die MAC Adresse besteht aus einer 48 Bit Zahl und war ursprünglich als «unique identifier» geplant. (einmalige Adresse)

Dargestellt wird die MAC Adresse in der Regel als Hexadezimales doppel Hextet. zB. 00 : 25 : 22 : CE : 1B : F2

I2C

Bei I2C ist die Adresse in der Regel 7 Bit lang. Es gibt aber auch Bausteine welche Adressen mit 10/16 Bit spezifizieren. (→ Inkompatibel zueinander...)

CAN

Bei CAN beinhaltet das «Identifier-Field» die Adresse und ist 11 Bit lang. Es gibt dabei noch ein Extended Frame Format, bei welchem das Identifier-Field zusätzlich noch die Message-Priorität enthält. Dieses ist zusammen 29 Bit lang ist.

Mobilfunk

Bei Mobiltelefonen sind alle Teilnehmer über die IMSI (international mobile subscriber identity) eindeutig nummeriert und identifizierbar.

IMSI ist eine 64 Bit Zahl. Die ersten 3 Ziffern sind der mobile country code (MCC) danach folgen 2/3 Bit für den mobile network code (MNC). Die ganze Nummerierung ist noch ein wenig komplizierter, da auch andere Numerierungsvarianten definiert und im Umlauf sind.

Da wir alle im Drahtlosnetz abhörfähig sind, wird der IMSI selbst nur selten übertragen. Es wird dem Telefon nach dem Anmelden eine temporäre Nummer : TMSI vergeben, welche bei der Kommunikation verwendet wird. Aber seien wir ehrlich. Wird ein Gebiet länger überwacht, können die TSMI der IMSI zugeordnet werden.

Zusammenfassung

Das Adressierungsschema ist bei jedem System anders. Schlussendlich entscheidet, wie viel Teilnehmer eineindeutig zugeordnet werden sollen.

Im Mobilfunk wollen wir klar eine IMSI Adresse für genaue eine Sim Karte auf der Welt haben.

Für ein I2C Steuerungsbust an dem sowieso nicht sehr viele Geräte angeschlossen sind reichen uns die 7 Bit aus. Und bei Ethernet will man genau eine MAC Adresse in einem Subnetz haben.

Aufgabe 12.50

Frage

Wie viele IEEE802 MAC Adressen sind möglich?

Antwort

$$2^{48} = 281'474'976'710'656$$

Aufgabe 12.51

Frage

Sie sehen hier ein I2C-Write Dataframe abgebildet. Zeichnen Sie das Adressfeld ein. Wieviele Adressen können Sie damit anwählen?

Antwort

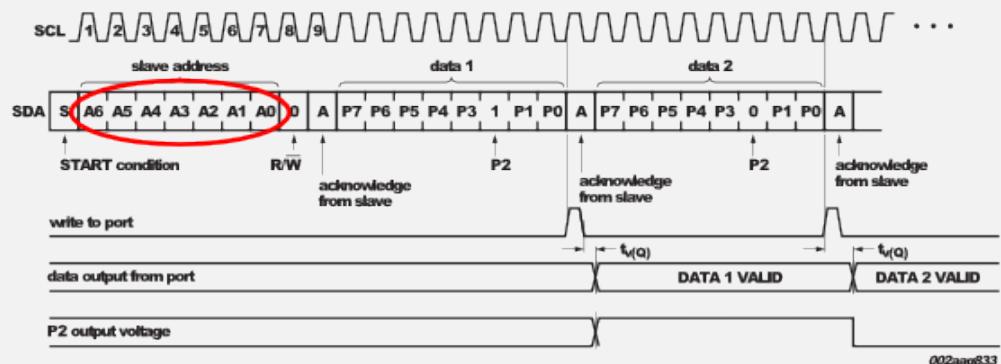


Fig 12. Write mode (output)

$$2^7 = 128$$

12.4 Dataframe Beispiel Implementationen

Besprechen wir nun ein paar Dataframe Beispiel Implementationen von diversen Standards. Ich möchte das Sie verstehen, dass das Design eines Protokolls darauf basiert, die notwendigen Bausteine auszuwählen und sinnvoll zusammen zu setzen.
Einfach gesagt : Wir spielen Lego™

12.4.1 UART Dataframe

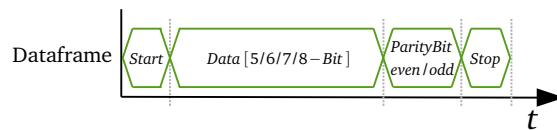


Abbildung 12.7: UART Dataframe

In Abbildung 12.7 sehen wir die Dataframe Implementation von UART. Wir sehen ein Start und Stop-Feld, ein optionales Checksummenfeld (Parity Bit) und natürlich das Datenfeld, welches zwischen 5 – 8 Bit aufweisen kann.

12.4.2 I2C Frame



Abbildung 12.8: I2C Dataframe

In Abbildung 12.8 ist ein Read Dataframe einer I2C Verbindung abgebildet. I2C weist ebenfalls ein Start- und Stop-Feld auf. Nach dem Start-Feld folgt das Adressierungs-Feld.

Danach folgt ein Datatype-Feld. Man nennt dieses Feld bei I2C das Write/Read Feld. Mit diesem Feld sagt man, ob das nächste Byte vom Master gesendet (write) oder vom Slave gesendet wird (read).

Und nach diesem Feld, folgt ein ACK-Feld. (Acknowledge) Diese Feld gehören eigentlich zum OSI-Layer 3. Mit dem Acknowledge-Feld muss der Slave, dem Master signalisieren, dass er die Nachricht erhalten und verarbeitet hat. Wir werden das genauer im Kapitel 16 : Flusssteuerung betrachten.

Wir sehen, I2C bricht hier mit dem OSI-Modell.

12.4.3 Ethernet Frame

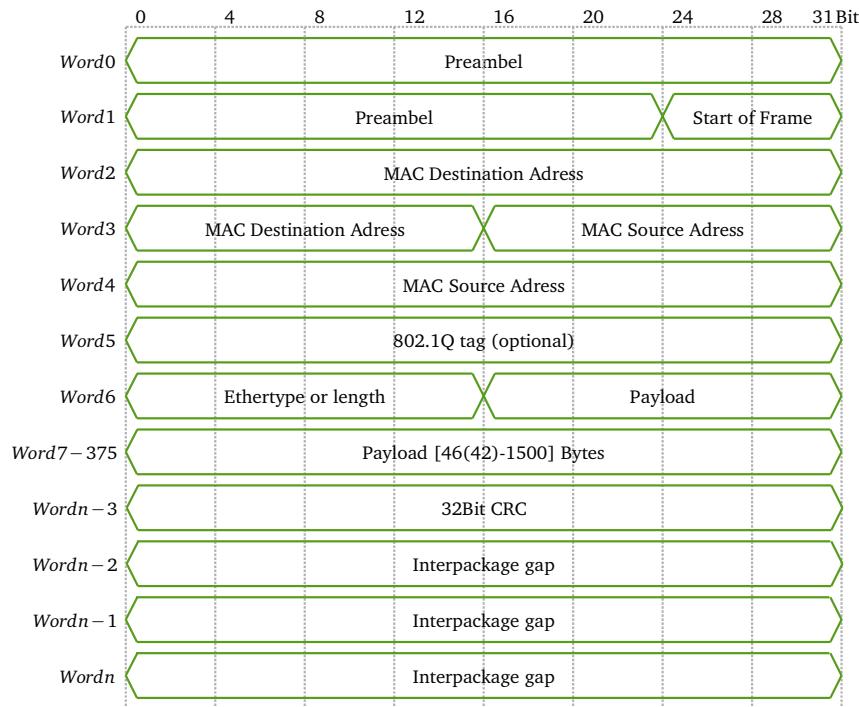


Abbildung 12.9: Ethernet Dataframe

In Abbildung 12.9 ist ein Ethernet MAC-Frame abgebildet. Er bildet den allgemeinen Dataframe sehr gut ab.

Wir haben ein Start- und Stop-Feld. (Start-of-Frame / Interpackage Gap) Die Preamble ist das Synch-Feld. Außerdem haben wir die Destination- und Source-Adress-Felder definiert.

Wir haben danach einen etwas exotischen 802.1Q tag. Dieser wird für sog. Virtual-LAN auf Layer 2 benötigt. Dies decken wir aber in diesem Kurs leider nicht ab.

Wir haben ein EtherType-Feld (Datatype-Feld). Danach folgen die Daten (Payload) und zum Schluss mit dem 32 Bit CRC das Checksummen-Feld.

Zusammenfassung

Wie wir bei diesen Beispielen gesehen haben, sind die Grundbausteine für ein Dataframe immer die selben und decken klassischerweise die OSI-Layer 1/2 Bedürfnisse ab.

Wir haben ausserdem gesehen, dass I2C oder auch Ethernet in Ihrem Dataframe noch andere Datenfelder definiert (ACK / 802.1Q tag).

12.5 Bit-Stuffing

Ich möchte an dieser Stelle noch kurz das sog. **Bit-Stuffing** besprechen.

Was ist eigentlich das Problem?

Wir haben das Problem, das unsere Daten genau gleich aussehen können wie ein Start-/Stop-Symbol. Wir erkennen aus dem Bitstrom nicht, ob wir uns innerhalb eines Dataframes befinden oder nicht.

Ethernet verwendet hierzu sog. Bit-Stuffing, um das *Start-of-Frame* zu maskieren.
(Siehe Abbildung 12.9)

Was ist damit gemeint?

Generell können unsere Daten jede beliebige Bitfolge enthalten. Aber wenn die Daten gleich aussehen wie ein **Start-of-Frame** kann die Start-Erkennungslogik durcheinander kommen und ein bereits angefangenen Frame mitendrin verwerfen.

Im Grunde ist dies ein Softwareproblem, welches man mit ein wenig mehr Hardware-Aufwand lösen kann.

Wie funktioniert Bit-Stuffing bei Ethernet

Das *Start-of-Frame* Feld wird bei Ethernet mit einer Bitfolge 0111110 definiert.

Das heisst, wir versuchen diese Bitfolge in den Daten zu maskieren, so dass wir unsere maskierten Daten nicht wie ein *Start-of-Frame* aussieht. Aber wir machen das so elegant, dass wir beim Empfänger die maskierten Daten erkennen und wieder zurückrechnen können.

Die *Start-of-Frame* Bitfolge ist wie erwähnt 0111110 demnach 6 · 1 und jeweils eine 0 am Anfang und am Ende.

Wir definieren uns nun eine Regel, wie wir unsere **Daten** verändern. In diesem Falle definieren wir die Regel so, dass nach 5 · 1 wir eine 0 einschieben.

So entsteht in den Daten niemals die gesuchte *Start-of-Frame* Bitfolge und der Empfänger weiss, dass er nach 5 1 das nachfolgende 0 herauslöschen muss, damit er den Original Bitstrom erhält.

Daten :	[01011111 01011111 10111011111 11011111 111]
Bitstuffed :	[01011111 0 01011111 0 10111011111 0 11011111 0 111]

Abbildung 12.10: Bit-Stuffing Beispiel : Nach 5 · 1 wird eine 0 eingeschoben

In Abbildung 12.10 sind die Sendebitströme als Grafik abgebildet. Wir sehen zu oberst den originalen (Daten-) Bitstrom. Darunter sehen wir den «Bit-Stuffed» Bitstrom, wie er über das Medium versendet wird.

Aufgabe 12.52

Frage

Wir möchten ein Bit Stuffing für die folgende Sequenz definieren : Start-/Stop-Symbol :01010101

Wie könnte eine Bit Stuffing Regel dafür aussehen? (es gibt mehrere Lösungen)

Stopfen sie die folgende Bitsequenz:

Bitsequenz :111010101111010101011001010

Antwort

Beispielsweise können wir nach 010101 eine **1** einschieben:

Original Bitsequenz : 111010101 111010101 011001010

(12.2)

Bit-stuffed Sequenz : 111010101**1**111010101**1**011001010

Aufgabe 12.53

Frage

Nehmen wir nochmals die gleiche Sequenz und das gleiche Start-/Stop Symbol wie in

Aufgabe 12.52

Start-/Stop-Symbol :01010101

Bitsequenz :111010101111010101011001010

Wir definieren die Regel : Nach 010101 eine **0** einschieben.

Funktioniert diese Methode?

Antwort

Original Bitsequenz : 111010101 111010101 011001010

Bit-stuffed Sequenz : 111010101**0**111010101**0**011001010

(12.4)

111010101**0**1110101010011001010

Nein! Diese Bit-Stuffing Regel funktioniert nicht! Wir haben (rot markiert) eine Bit-stuffed

Sequenz generiert, welche das Start- / resp. Stop Symbol enthält.

Damit haben wir die eigentliche Funktion des Bitstuffings nicht erfüllt!

13

Fehlerkorrektur

Inhalt des Kapitel

13.1	Grundlagen	178
13.1.1	Block-Codes	178
13.1.2	Convolution-Codes	178
13.2	Hamming Code	179
13.2.1	Hamming-Distanz	180
13.2.2	Linearer Code	181
13.2.3	Fehlererkennung	181
13.2.4	Fehlerkorrektur	181
13.2.5	Diverse Hamming-Codes mit $d_{min} = 3$	181
13.2.6	Zusammenfassung	181
13.2.7	Hamming Code mit Matrizen	183
13.3	BCH-Code	187
13.3.1	Encoder	187
13.3.2	Decoder	187
13.3.3	Beispiel	188
13.4	Reed-Solomon Code	189
13.5	Trellis / Viterbi	190
13.5.1	Encoder Struktur	190
	Coderate	191
	State-Machine	191
	Free Distance / Correcting Capability	192
	(Viterbi-) Decoder	193
13.6	Turbo-Code	194

Zum Inhalt

Wir besprechen in diesem Kapitel wie wir Infodaten mit sog. Block-Codes und Faltungs-Codes schützen können, so dass wir nicht nur Fehler erkennen können, sondern auch Fehler korrigieren können. Dies nennt man auch eine sog. *Forward Error Correction (FEC)*.

Dieses Kapitel baut stark auf *Kapitel 10 : Fehlererkennung* auf.

13.1 Grundlagen

Wie in *Kapitel 10.1 : Kanalmodell* beschrieben, lässt sich mit der *forward error correction (FEC)* der Sendebitstrom so codieren, dass man den verrauscht und damit fehlerbehafteten Empfangs-Bitstrom wieder fehlerfrei decodieren kann.

Aber das hat seinen Preis! Wir müssen viel mehr Rechnen. FEC-Codes sind daher für (billige-) Sensoren oder (schwache-) Systeme nicht geeignet.

Wie so oft gibt es viele Varianten, welche gewisse Vor- und Nachteile aufweisen. Wir werden die bekanntesten kurz besprechen. Man kann diese grob in **Block-Codes** und **Faltungs-Codes** (engl. *Convolution-Codes*) einteilen.

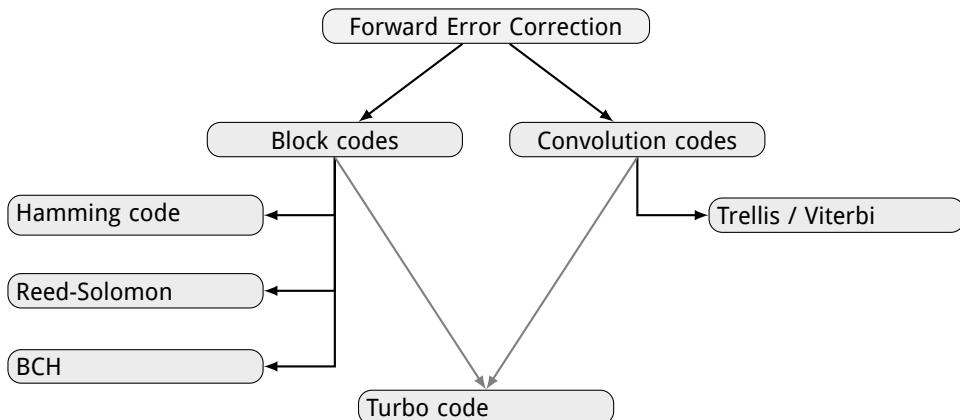


Abbildung 13.1: Forward Error Correction - Übersicht

In *Abbildung 13.5* sind die FEC's aufgeführt, welche wir besprechen werden. Den Turbocode werden wir nicht im Detail anschauen. Zum einen ist er sehr komplex und zum anderen fehlt uns die Zeit. Sie müssen schlicht wissen, dass der Turbocode zur Zeit einer der besten Fehlerkorrektur-Algorithmen der Welt darstellt.

13.1.1 Block-Codes

Block-Codes werden immer in festen Datenblöcken zusammen gefasst und gesamt codiert. Die algebraischen Eigenschaften der Block-Codes sind gut untersucht. Diese basieren auf sog. Linearen-Codes und werden daher auch als **linear block codes** bezeichnet.

Nachteil der Block-Codes ist, dass für die Codierung / Decodierung der ganze Datensatz vorliegen muss. Wird eine Nachricht empfangen, muss also gewartet werden, bis die gesamte Nachricht empfangen wurde. Erst danach kann mit der Decodierung begonnen werden. Daher sind die Codier- / Decodier-Latzen relativ hoch.

Latenz

Als Latenz [W36] bezeichnet man in der Technik eine Verzögerungszeit zwischen 2 Ereignissen. In diesem Fall ist das Ereignis der Zeitpunkt des Empfangs eines neuen Nachrichten-Block und der Zeitpunkt bis wir die Daten restlos Decodiert haben. Dadurch, dass wir zunächst abwarten müssen, bis der ganze Datenblock vorliegt, vergeht eine (je nach Blockgrösse) lange Latenz-Zeit.

Wir werden sehen, das dies bei einem Convolution-Code nicht der Fall ist.

13.1.2 Convolution-Codes

Die Convolution-Codes (deutsch *Faltungs-Codes*) sind die derzeit effizienteste Klasse der Forward Error Correction Codes. Diese basieren auf (komplizierten) Anordnungen von XOR und Schieberegistern. Damit wird das Sendesignal mit bereits gesendeten Daten überlagert. In der Mathematik, nennt man dies eine Faltung, daher auch der Name.

Wir erinnern uns an die Eigenschaften des BSK (Binär Symmetrischen Kanal, Siehe : *Kapitel 10.1 : Kanalmodell*). Dieser ist gedächtnislos. Die Convolution-Codes nutzen diese Eigenschaft aus, indem man die Bit-Information über die Zeit verschmiert.

Die Convolution-Codes haben bei Ihrer Vorstellung um ca. 1991 die derzeitigen Übertragungsraten um ein vielfaches übertrafen und sind nahe an der Shannon-Kanal-Kapazitäts-Grenze. Insbesondere der Turbo-Code wurde anno dazumals als «near shannon limit error-correcting-code» bezeichnet. Nachteilig ist die hohe Komplexität.

13.2 Hamming Code

Der Hamming-Code [W26] gehört zu den Block-Codes. Je länger der Datenbitstrom, desto effizienter wird der Hamming-Code. Zur Erklärung nehmen wir aber eine kurze, einfache aber ineffizienten Code Länge.

Beispiel

Nehmen wir einen Hamming (3,1). Dieser encodiert unser Datenwort I zum Codewort C wie folgt:

$$\begin{aligned} I = 0 &\rightarrow C = 000 \\ I = 1 &\rightarrow C = 111 \end{aligned} \tag{13.1}$$

Dies entspricht eigentlich einem Datenbit mit zwei even-Parität-Bit. Wir können uns das Ganze nun als Würfel vorstellen. Jedes Codebit entspricht einer Achse (x, z, y):

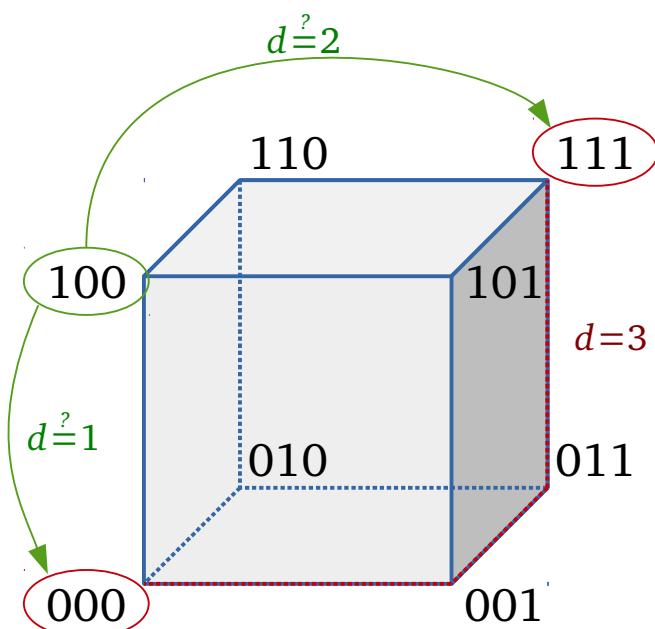


Abbildung 13.2: Hamming Code (3,1)

Sehen wir uns *Abbildung 13.2* genauer an. Rot umkreist sind unsere möglichen gesendeten Codewörter. Jedes Codewort ist über eine Kante mit einem anderen Codewort verbunden. Wenn wir jede Kante als Distanz $d = 1$ auf fassen, sind unsere gesendeten Codewörter über die Distanz $d = 3$ verbunden (rot gestrichelte Linie). Grün eingezeichnet ist ein empfangenes Codewort und deren Distanzen zu den zulässigen Codewörtern.

Fehlerkorrektur Dilemma

Wir erkennen nun auch gut das Dilemma des Hamming Codes (3,1).

Empfangen wir zB. den Code $C = 100$, so wissen wir, dass ein Fehler aufgetreten ist. Aber ist es ein *single-bit-error* oder ein *double-bit-error*?

Wir wissen es nicht!

Wir können nun immer annehmen, dass bei so einem Codewort nur 1 Fehler aufgetreten ist und das Codewort auf das nächste Symbol korrigieren. Dies ist ein legitimer Ansatz, aber ein Zweibit-Fehler würde falsch korrigiert werden.

13.2.1 Hamming-Distanz

Der «Abstand» von zwei Codewörtern wird Hamming-Distanz genannt. Die Hamming-Distanz ist auch bei anderen Block-Codes von Interesse und ist daher ein wichtiges Kriterium.

Die Hamming-Distanz ist folgendermaßen definiert:

$$d_{ij} = \sum_{\text{Bits}} C_i \text{ xor } C_j \quad (13.2)$$

Die mathematische Formulierung ist hier ein wenig ungewöhnlich zu verstehen.

Daher hier ein Beispiel: Wir haben zwei (beliebige) Codewörter $C_1 = 0x95$ und $C_2 = 0xB3$

$C_1 = 0x95 :$	1	0	0	1	0	1	0	1
	$= 1$							
$C_2 = 0xB3 :$	1	0	1	1	0	0	1	1
	xor: 							
	0	0	1	0	0	1	1	0
	$\rightarrow d_{12} = \sum_{\text{Bits}} = 3$							

Abbildung 13.3: Hamming Distanz

Die beiden Codewörter werden zunächst XOR verknüpft und danach die Anzahl log. 1 gezählt. Das ist die Hamming-Distanz. In diesem Beispiel ist die Hamming-Distanz $d_{12} = 3$

minimale Hamming-Distanz

Wichtiger als die Hamming-Distanz beliebiger Codewörter ist die **minimale Hamming-Distanz** über alle Codewörter.

$$d_{min} = \min_{i,j \in C: i \neq j} (d_{ij}) \quad (13.3)$$

Auch hier ist die mathematische Beschreibung schwierig zu lesen: Bei der *minimalen Hamming-Distanz* vergleichen wir alle Codewörter miteinander $\min(d_{ij})$. Damit wir jetzt mathematisch korrekt sind, müssen wir den Wertebereich von $i, j \in C$ angeben. Aber es gibt die Ausnahme, dass i und j nicht den selben Index aufweisen dürfen, sonst würden wir zweimal das gleiche Codewort miteinander XOR verknüpfen. Daher $i, j \in C : i \neq j$

13.2.2 Linearer Code

Der Hamming-Code gehört zu den Linearen-Codes, hierzu müssen folgende Eigenschaften erfüllt sein:

- Das Null Codewort muss enthalten sein (Alle Bits 0)
- zwei Codewörter mit XOR Verknüpft geben wieder ein Codewort $C_i = C_x \text{ xor } C_y$

13.2.3 Fehlererkennung

Ein linearer Block-Code kann:

$$k_r = d_{min} - 1 \quad (13.4)$$

Fehler erkennen.

13.2.4 Fehlerkorrektur

Ein linearer Block-Code kann:

$$k_c = \frac{d_{min} - 1}{2} \quad (13.5)$$

Fehler korrigieren.

13.2.5 Diverse Hamming-Codes mit $d_{min} = 3$

Den Hamming Code können wir noch verallgemeinern.

Erweitern wir die Länge der Datenbits, müssen die Parity-Bits einem gewissen Muster nachfolgen, damit wir immer einen *single-bit-error* korrigieren können und damit eine min. Hamming-Distanz von $d_{min} = 3$ erreichen.

Bezeichnung	Datenbits	Parity-Bits	Total Bits	Coderate
Hamming(3,1)	1	2	3	$R_C = R_B \cdot \frac{1}{3} \approx 0.333 \cdot R_B$
Hamming(7,4)	4	3	7	$R_C = R_B \cdot \frac{4}{7} \approx 0.571 \cdot R_B$
Hamming(31,26)	26	5	31	$R_C = R_B \cdot \frac{26}{31} \approx 0.839 \cdot R_B$
Hamming(m,n)	$n = 2^p - p - 1$	p	$m = 2^p - 1$	$R_C = R_B \cdot \frac{2^p - p - 1}{2^p - 1}$
Hamming(m,n)	$n = m - p$	p	m	$R_C = R_B \cdot \frac{n}{m}$

Tabelle 13.1: Hamming-Codes für $d_{min} = 3$

Durch Anhängen zusätzlicher Parity-Bits können noch mehr *bit-error* detektiert und korrigiert werden. In der letzten Zeile ist noch die allgemeine Berechnungsformel für den Hamming-Code gegeben. Aber Achtung : Hier wissen Sie nicht wie gross die min. Hamming-Distanz ist.

Hier müssen Sie alle Codewörter testen um die minimale Hamming Distanz zu ermitteln!

13.2.6 Zusammenfassung

Vorteil

Der Hamming Code kann Fehler detektieren und korrigieren.

Nachteil

Der Hamming Code kann höchstens so viele Burst-Fehler korrigieren, wie allgemeine Fehler. Die Implementation basiert auf einem «Dictionary» Ansatz. Der Encoder wie Decoder ist nicht sehr schnell.

Aufgabe 13.54

Frage

Bestimmen Sie die minimale Hamming Distanz der folgenden Codewörter.

Ist dies ein Linearer Code?

Antwort

Wir müssen nun von allen Kombinationen von Codewörtern xor Verknüpfen und die resultierenden log. 1 zählen. Das ist die Hamming Distanz. Unser gesuchtes Ergebnis ist dann die minimale Hamming Distanz.

$$\begin{aligned}
 d_{ij} &= \sum_{\text{Bits}} C_i \text{ xor } C_j \\
 d_{01} &= \sum_{\text{Bits}} 0011 \text{ xor } 1100 = \sum_{\text{Bits}} 1111 = 4 \\
 d_{02} &= \sum_{\text{Bits}} 0011 \text{ xor } 1001 = \sum_{\text{Bits}} 1010 = 2 \\
 d_{03} &= \sum_{\text{Bits}} 0011 \text{ xor } 0000 = \sum_{\text{Bits}} 0011 = 2 \\
 d_{12} &= \sum_{\text{Bits}} 1100 \text{ xor } 1001 = \sum_{\text{Bits}} 0101 = 2 \\
 d_{13} &= \sum_{\text{Bits}} 1100 \text{ xor } 0000 = \sum_{\text{Bits}} 1100 = 2 \\
 d_{23} &= \sum_{\text{Bits}} 1001 \text{ xor } 0000 = \sum_{\text{Bits}} 1001 = 2 \\
 d_{min} &= \min(d_{ij}) = \min(d_{01}, d_{02}, d_{03}, d_{12}, d_{13}, d_{23}) = 2
 \end{aligned}$$

Falls die gegebenen Codewörter alle vorhandenen Codewörter sind : Nein es ist kein Linearer Code, da zwei xor verknüpfte Codewörter noch andere Bitsequenzen generieren.
Immerhin, das Null Codewort ist gegeben.

Aufgabe 13.55

Frage

Berechnen Sie die Coderate des folgenden Hamming-Codes:

Antwort

$$R_C = R_B \cdot \frac{1013}{1023} = 0.99022 \cdot R_B$$

Aufgabe 13.56

Frage

Wie viele Fehler kann man mit einer minimalen Hamming Distanz von $d_{min} = 12$ detektieren und wie viele korrigieren?

Antwort

detektieren = 11

korrigieren = 5

Achtung, immer abrunden, da man nicht ein halbes Bit korrigieren kann!

13.2.7 Hamming Code mit Matrizen

Wir konstruieren nun als Beispiel einen Hamming-Code(7,4) in Matrizen-Darstellung. Damit möchte ich zeigen, dass wenn unsere (Kommunikations-) Maschine, Matrizen rechnen kann, dann ist ein Hamming Code sehr einfach umzusetzen. Aus der Definition können wir nun schliessen, das wir 4 Databit und 3 Paritybit zu einem 7 Bit langen Codewort zusammen setzen möchten.

Wir überlegen uns zunächst wie wir die Parity-Bits auf die Databits aufteilen wollen, damit wir eine möglichst gute Fehlerkorrektur realisieren können:

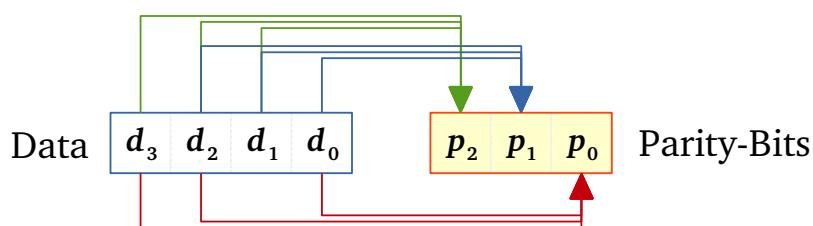


Abbildung 13.4: Hamming Code (7,4)

In Abbildung 13.4 ist eine sinnvolle Aufteilung dargestellt.

Mathematisch, lässt sich das folgendermassen darstellen:

$$\begin{aligned} p_2 &= d_3 + d_2 + d_1 \\ p_1 &= d_2 + d_1 + d_0 \\ p_0 &= d_3 + d_2 + d_0 \end{aligned} \quad (13.6)$$

Wir sehen in Formel (13.6) das wir die Parity-Bits über alle Datenbits verteilen. Aber jedes Parity-Bit deckt unterschiedliche Datenbits ab!

schlechte Paritybit Wahl

Ich zeige nun eine schlechte Wahl: (benutzen Sie diese Variante niemals!)

$$\begin{aligned} p_2 &= d_3 + d_2 + d_1 \\ p_1 &= d_0 \\ p_0 &= d_0 \end{aligned} \quad (13.7)$$

In Formel (13.7) sehen wir nun die schlechte Variante. Die Parity-Bits p_1, p_0 decken das gleiche Datenbit ab und nur p_2 die restlichen Daten. Daraus lässt sich keine gute Fehlerkorrektur realisieren! (ohne Beweis)

Aber : Schlussendlich werden Sie nie selbst ein Hamming-Code Schema erstellen müssen. Hierzu gibt es bereits genügend getestete Varianten, welche in Tabellenbüchern hinterlegt sind.

Paritybit-Matrix P

Zurück zu unserem Beispiel:

Wir schreiben die Gleichungen aus *Formel* (13.6) nun als Matrize:

$$P^T = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

$$P = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \quad (13.8)$$

Generator-Matrix G

Damit generieren wir uns die Generator-Matrix:

I_k = Einheitsmatrix

$$G := [P|I_k]$$

$$G := \left[\begin{array}{ccc|cccc} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{array} \right] \quad (13.9)$$

Encoder

Mit der Generator-Matrix, können wir nun unseren Encoder bauen.

Jedes Codewort wird wie folgt berechnet:

u := Infowort / Daten, x := (Sende-) Codewort

$$x := u \cdot G \quad (13.10)$$

Codewort

Damit sieht unser Codewort folgendermassen aus:

$$x = u \cdot G = : \boxed{p_2 \ p_1 \ p_0 \ d_3 \ d_2 \ d_1 \ d_0}$$

Abbildung 13.5: Hamming(7,4) Codewort

Parity-Check Matrix : H

Damit wir die Daten wieder decodieren können, entwerfen wir die sog. Parity-Check Matrix:

$$H := [I_{n-k}|P^T]$$

$$H := \left[\begin{array}{ccc|cccc} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{array} \right] \quad (13.11)$$

Fehler-Syndrom Vektor : s

Aus dem Empfangs-Codewort und der Parity-Check Matrix können wir ausrechnen, ob wir einen Fehler in den empfangenen Daten haben. Dieser Fehlervektor wird **Syndrom** genannt.

y := (Empfangs-) Codewort, e := Error-Vektor

$$s := y \cdot H^T = (x + e) \cdot H^T \quad (13.12)$$

Haben wir keine Fehler empfangen, ist das Syndrom $s = 0$. Andernfalls sind Fehler im Empfangs-Codewort enthalten.

Korrektur-Tabelle

Damit wir nun allfällige Fehler korrigieren können, müssen wir zunächst noch ein wenig vorarbeit leisten. Wir betrachten zunächst die Berechnung des Syndrom-Vektors. Wir sehen in *Formel* (13.12) das unser Empfangs-Codewort y eigentlich nichts anderes ist, als unser Sende-Codewort x plus einen Fehler-Vektor e . Umgekehrt, wenn wir e kennen, können wir unsere korrigiertes Codewort berechnen!

$x' = y + e$. Ausserdem wissen wir, dass das Syndrom $s = 0 \stackrel{!}{=} x \cdot H^T$ ist, wenn unser Empfangswort keinen Fehler aufweist.

$$\begin{aligned} s &= y \cdot H^T = (x + e) \cdot H^T \\ s &= x \cdot H^T + e \cdot H^T \\ 0 \stackrel{!}{=} x \cdot H^T \\ s &= 0 + e \cdot H^T = e \cdot H^T \end{aligned} \tag{13.13}$$

Wir können nun vorausberechnen, was ein 1 Bit Fehler für ein Syndrom-Vektor erzeugt.

Wir bauen uns aus allen möglichen Fehlervektoren eine Tabelle, wo wir herauslesen können, welche Fehler zu welchem Syndrom führen.

$e_x : 1$ Bit Fehler-Matrix

$$S = e_x \cdot H^T \Rightarrow \left[\begin{array}{ccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right] \cdot \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{array} \right] = \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{array} \right] \tag{13.14}$$

Wir sehen hier nun folgendes : Welcher Fehlervektor führt zu welchem Syndrom. Oder umgekehrt: Wir können aus dem Syndrom herauslesen wie der Fehlervektor e aussieht.

Fehlerkorrektur

Wie gehen wir bei der Fehlerkorrektur vor:

1. Wir berechnen aus unserem Empfangswort das Syndrom: $s = y \cdot H^T$
2. $s \stackrel{?}{=} 0$, falls ja kein Fehler empfangen = Daten okay
3. Falls $s \neq 0$ suchen wir in der S Tabelle den gleichen Eintrag i wie das Syndrom.
4. Wir nehmen auf der Zeile i den dazugehörigen Fehlervektor e_i heraus.
5. Wir korrigieren unser Empfangswort $x' = y - e$

Anmerkung

Auch hier verwenden wir die binäre Mathematik. Ebenfalls bei einer Matrizenrechnung. Das heisst, dass alle gerade Werte zB. (0,2,4,6,8) auf ein log. **0** führen, während alle ungerade Zahlen auf ein log. **1** führen.

Beispiel:

$$\begin{aligned} [1 & 1 & 1 & 0] \cdot \left[\begin{array}{ccc} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{array} \right] \\ &= [1 \cdot 1 + 1 \cdot 1 + 1 \cdot 1 + 0 \cdot 0, 1 \cdot 0 + 1 \cdot 1 + 1 \cdot 1 + 0 \cdot 1, 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 0 + 0 \cdot 1] \\ &= [3, 2, 2] = [1, 0, 0] \end{aligned}$$

Aufgabe 13.57**Frage**

Rechen wir die Codewörter für unseren Hamming-Code(7,4) für die folgenden Datenwörter:
Benutzen Sie die Werte aus *Kapitel 13.2.7*

Antwort

$$x = u_0 \cdot G = [1 \ 1 \ 0 \ 0] \cdot \left[\begin{array}{ccc|cccc} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{array} \right] = [0 \ 1 \ 0 \mid 1 \ 1 \ 0 \ 0]$$

$$x = u_1 \cdot G = [1 \ 1 \ 0 \mid 1 \ 0 \ 0 \ 1]$$

$$x = u_2 \cdot G = [1 \ 0 \ 1 \mid 1 \ 0 \ 0 \ 0]$$

$$x = u_3 \cdot G = [0 \ 0 \ 0 \mid 0 \ 0 \ 0 \ 0]$$

Die Linie zwischen dem 3ten und 4ten Bit wurde von mir eingefügt und hat keine mathematische Bedeutung. Sehen Sie sich das mal genauer an. Der rechte Teil ist immer unser Infowort. Dies folgt aus der Einheitsmatrix in der Generatormatrix. Solch einen Code nennt man übrigens Systematisch.

Aufgabe 13.58**Frage**

Wir haben folgende Codewörter erhalten. Welches davon weist einen Fehler auf?
Die Codewörter wurden von unserem Hamming-Code(7,4) generiert. Benutzen Sie die Werte aus *Kapitel 13.2.7*

Antwort

$$s = y_0 \cdot H^T = [1 \ 1 \ 1 \mid 0 \ 1 \ 0 \ 0] \cdot \left[\begin{array}{ccccccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{array} \right] = [0 \ 0 \ 0] \Rightarrow \text{Kein Fehler}$$

$$s = y_1 \cdot H^T = [0 \ 0 \ 0] \Rightarrow \text{Kein Fehler}$$

$$s = y_2 \cdot H^T = [1 \ 0 \ 1] \Rightarrow \text{Fehler!}$$

Aufgabe 13.59

Frage

Bestimmen Sie mithilfe der Error-Tab S den Fehlervektor e_i und das richtige korrigierte Codewort x' , aus dem fehlerbehafteten Codewort aus der letzten Aufgabe.

Benutzen Sie die Tabelle aus Kapitel 13.2.7

Antwort

$$s = [1 \ 0 \ 1] \Rightarrow \text{Aus der Tabelle :} e_i = [0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0]$$

$$x' = y - e = [0 \ 1 \ 1 \mid 1 \ 0 \ 0 \ 1] - [0 \ 0 \ 0 \mid 1 \ 0 \ 0 \ 0] = [0 \ 1 \ 1 \mid 0 \ 0 \ 0 \ 1]$$

13.3 BCH-Code

BCH-Codes [W6] wurden nach deren Entwickler, Raj **Bose**, D.K. Ray-Chaudhuri und Alexis **Hocquenghem** benannt. Die BCH-Codes wurden daraufhin entwickelt, dass diese über die Anzahl korrigierbare Bit entworfen werden.

13.3.1 Encoder

Der BCH Encoder ist als *zyklischer Block-Code* ausgelegt und wird über ein Generatorpolynom beschrieben. Wenn sie das nun an das CRC Verfahren erinnert liegen sie richtig. BCH Encoder werden auch mit rückgekoppelte Schieberegister realisiert.

Die Wahl des Generatorpolynoms hängt vom Designziel ab und kann davon abgeleitet werden.

Man kann die folgenden Parameter variieren:

$BCH(m, n, t)$

- m : Codewort Länge
- n : Datenwort Länge
- t : error-correcting capability
- $k = m - n$: Anzahl der Redundanz-Bits

Es können wie beim Hamming-Code t Fehler erkannt, $t = \frac{d_{min}-1}{2}$ korrigiert und beim BCH noch $r \leq k$ burst-error erkannt werden.

Der Grad des Generatorpolynoms lässt sich über $k = \deg[g(x)]$ definieren.

Die Generatorpolynome sind bis zu einer Länge von ca. 1023 Bit bereits tabelliert. [O1]

Die Codierung wird wie bei einem CRC implementiert.

13.3.2 Decoder

Zur Decodierung wird häufig der Berlekamp-Massey Algorithmus verwendet.

Die Beschreibung würde hier den Rahmen sprengen.

Einfach kurz die Eckpunkte : Zuerst wird aus den empfangenen Codesymbolen das sog. Syndrom berechnet (Fehlervektor). Ist dieser $s \neq 0$ ist ein Fehler aufgetreten. Mithilfe des Generatorpolynoms und des Syndroms wird der error-locator-vector $\lambda(x)$ berechnet, der angibt an welcher Position sich der Fehler befindet. Danach kann man das korrigierte Codewort wieder dem Datenwort zuordnen.

Vorteil

Klare Vorgabe der Performance (Fehlerkorrektur, Länge) möglich. Effiziente Implementation. Erkennt burst-error.

Nachteil

Typisches Block-Code Problem: Latenz.

m	n	t	Generator Polynom
7	4	1	0xB
	1	3	0x4F
15	11	1	0x13
	7	2	0x1D1
	5	3	0x537
	1	7	0x7FFF
	26	1	0x25
31	21	2	0x769
	16	3	0x8FAF
	11	5	0x1626D5
	6	7	0x32DEA27
	1	15	0x7FFFFFFF

Tabelle 13.2: BCH-Codes : [O1]

Anmerkung

Beim Generatorpolynom des BCH Codes sind im Gegensatz zum CRC Polynom alle Koeffizienten angegeben!

13.3.3 Beispiel

Machen wir kurz ein Beispiel zum BCH-Code. Wir benutzen einen $\text{BCH}(7, 4, 1)$. Damit ist das Generatorpolynom $G = [0xB]$. Um unser Codewort zu bestimmen, benutzen wir die Multiplikationsmethode. Es gibt noch die Divisionsmethode, welche auf einen sog. Systematischer Code führt. Aber dies nur zur Info. (Bei einem Systematischen Code ist das Infowort enthalten)

$$\begin{aligned} \text{Data} &= [1100] \\ G(x) &= x^3 + x^1 + 1 = [1011] \end{aligned}$$

Data	Generator Polynom	Codeword
1 1 0 0	• 1 0 1 1	1 1 1 0 1 0 0
	1 1 0 0	
	1 1 0 0	
	1 1 0 0	
	1 1 0 0	
	1 1 1 0 1 0 0	= Codeword

Abbildung 13.6: BCH Code : Beispiel Multiplikationsmethode

Wir decodieren nun das Codewort beim Empfänger. Als erstes müssen wir das Syndrom bestimmen. Die Rote **log. 0** bezeichnet ein Falsch empfangenes Bit.

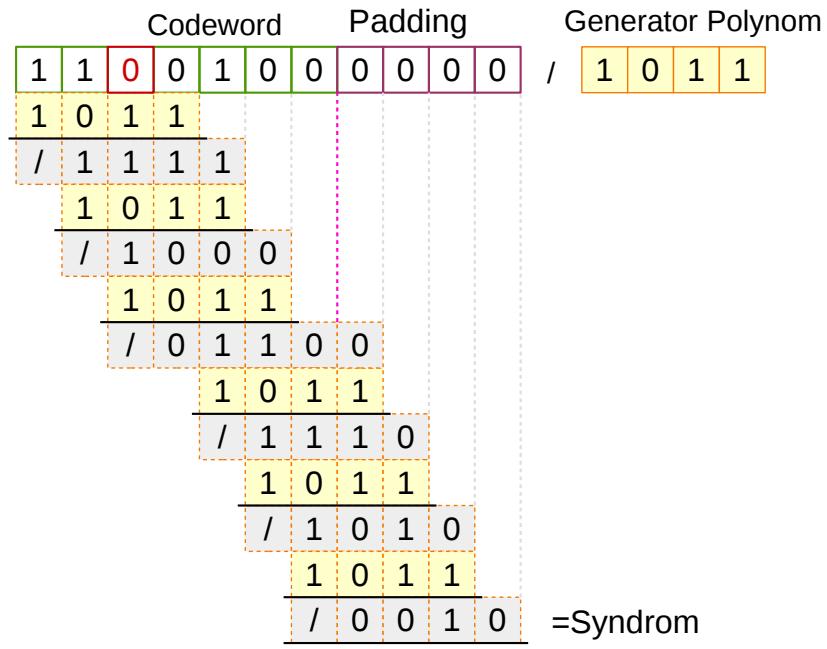


Abbildung 13.7: BCH Code : Syndrom berechnen

Das Syndrom ist nicht Null → somit ein Fehler im Codewort.

Wir beenden das Beispiel hier. Den Fehlervektor würde man nun mit dem Berlekamp-Massey Algorithmus auflösen.

13.4 Reed-Solomon Code

Der **Reed-Solomon Code (RS)** [W60] ist eine Untergruppe der BCH-Codes. Die RS-Code gehören ausserdem zu den sog. *maximum-distanz-separable code*. Der RS-Code ist nur für mehrwertige Symbole entworfen. (Binäre Codewörter genügen der MDS Voraussetzung nicht)

Vorteil

Wie BCH, aber effizienter bei mehrwertiger Modulation (QAM / Tertiär etc), vorteilhafte Distanzeigenschaften.

Nachteil

Gleich wie bei BCH

13.5 Trellis / Viterbi

Die Faltungscodes (Convolutional-Codes) wurden erstmalig 1965 von Peter Elias vorgestellt. Kurz darauf (1967) konnte Andrew Viterbi einen effizienten Decodieralgorithmus für die Klasse der Convolutional-Codes vorstellen. Auch die heutigen Convolution-Codes werden mit dem **Viterbi Algorithmus** [W72] decodiert.

Die **Trellis Codierung** [W68] wurde von Gottfried Ungerboeck in den 1970er Jahren entwickelt, aber erst 1982 veröffentlicht. Und ist der erste welche sehr nahe an die *Shannon-Kanal-Kapazitäts-Grenze* (C_s) heran kam. (Zur Erinnerung : Siehe Formel (5.1))

Anfang der 1980 Jahren war die grösste Übertragungsrate im *POTS (Plain old telefon service)* bei 14 kbit/s mit einer QAM-4 Modulation. Dies sind ca. 40 % der Shannon-Kanal-Kapazität bei den spezifizierten Bandbreite und Signalstärken. Mit der Trellis Codierung konnte G. Ungerboeck 33.6 kbit/s realisieren. Das sind ca. 96 % der Shannon-Kapazität!

Prüfung

An der Prüfung müssen Sie die Funktionsweise des Trellis nicht erklären können, aber die Eigenschaften nennen können.

13.5.1 Encoder Struktur

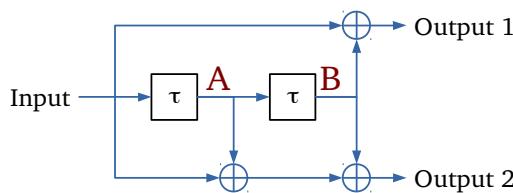


Abbildung 13.8: non-recursive Conv.Code $R_C = \frac{1}{2}$

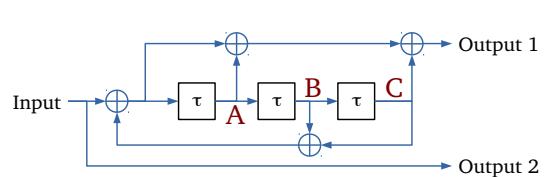


Abbildung 13.9: recursive Conv.Code $R_C = \frac{1}{2}$

In Abbildung 13.8 ist ein sog. *non-recursive Trellis* mit einer Coderate von $R_C = \frac{1}{2}$ abgebildet. Das heisst, das auf ein Datenbit, zwei Codebits folgen. Rot (A,B) sind die sog. states (engl. für Zustand) eingezeichnet. Das Viereck mit dem kleinen Tau τ soll ein Schieberegister darstellen. Die Kreuzsymbole sind EXOR Verknüpfungen.

Wie bereits erwähnt hat ein Trellis sog. Zustände. Mit jedem Bit welchen wir in den Trellis verarbeiten, verändert sich der «Zustand» des Trellis. Das Codewort wird mit den beiden Ausgängen zusammengesetzt.

Das ganze System bildet eine sog. Zustandsmaschine (engl. State-Machine). Sie werden den Begriff sicherlich noch in einer Software-Vorlesung hören.

Wie funktioniert der Trellis

Das interessante am Trellis ist, dass wir die Codebits von den vorhergehenden Datenbits abhängig machen. Man bezeichnet dies auch als «Verschmieren der Information über die Zeit». Man könnte sagen, wir sorgen mit der State-Machine dafür, dass unser Encoder ein «Gedächtnis» hat.

Bei einem nicht rekursiven Trellis, wird die «Erinnerung» über die Anzahl Schieberegister definiert und ist klar Deterministisch.

Der Trellis aus Abbildung 13.9 ist ein sog. rekursiver Trellis. Man erkennt dies daran, dass mind. einer der Ausgänge auf den Eingang zurückgeführt wird. Ein Rekursiver-Trellis kann bei gleicher Anzahl Schieberegister bessere Ergebnisse liefern, als ein non-recursive Typ, ist aber auch schwieriger zu analysieren. Ein Rekursiver-Trellis wird auch als *recursive-systematic convolution encoder* (RSC-Encoder) bezeichnet.

Coderate

$$\begin{aligned}
 k &:= \text{Anzahl Informationsbit} \\
 n &:= \text{Anzahl Codebit} \\
 R_C &= \frac{k}{n}
 \end{aligned} \tag{13.15}$$

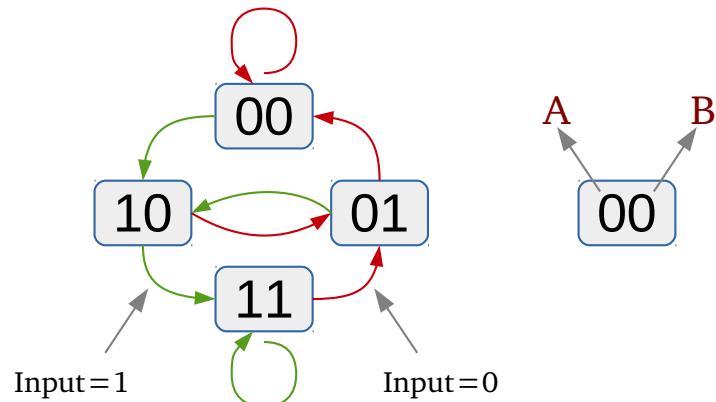
State-Machine

Abbildung 13.10: Zustandsdiagramm für $R_C = \frac{1}{2}$ non-recursive Trellis

In Abbildung 13.10 ist das Zustandsdiagramm des Trellis aus Abbildung 13.8 zu sehen. Die State-Machine besteht aus den beiden Ausgängen Output 1 / 2. Durch die Verschmierung sind nun nur noch gewisse Zustandsübergänge zulässig.

Terminierung

Da wir in der Regel immer Blockweise Daten übermitteln, können wir den Endzustand der State-Machine Nutzen um die Fehlerkorrektur noch zu verbessern. So wird am Schluss der Übertragung extra eine Sequenz übertragen, so dass der Endzustand immer den gleichen (auf beiden Seiten bekannten) Endzustand eintritt.

Allerdings verlängert die Terminierung auch die Codesequenz, womit die Netto-Datenrate abnimmt.

Trellis

Sehen wir uns einmal an, wie das ganze aussieht, wenn wir ein Datensignal auf den non-recursive $R_C = \frac{1}{2}$ Trellis geben.

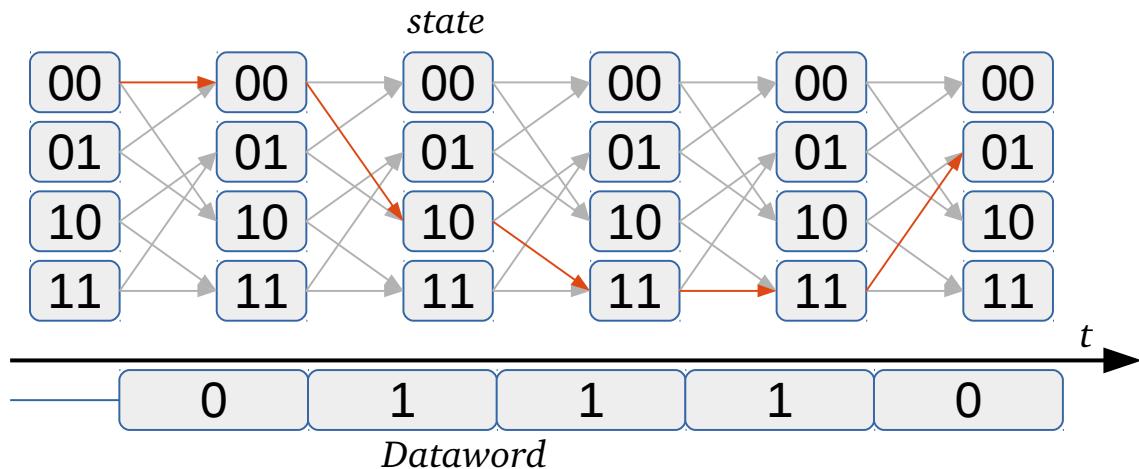


Abbildung 13.11: Trellis Diagram

In Abbildung 13.11 sind die Zustände der State-Machine und damit die Ausgänge Output 1 / 2 im Verlauf der Zeit eingezeichnet. Unten das gesendete Signal.

Free Distance / Correcting Capability

Das Interessante an den Convolution-Codes ist auch ihr Verhalten zur Fehlerkorrektur.

Zunächst, es gibt ebenso eine Hamming-Distanz bei den Convolution Code. Diese wird **free distance** d_{free} genannt. Diese ist ein wenig kompliziert zu berechnen, daher lassen wir das weg.

Mit der *free distance* können wir die Anzahl Bits bestimmen die Korrigiert werden können:

$$t = \frac{d_{free} - 1}{2} \quad (13.16)$$

Diese Formel ähnelt sehr Formel (13.5). Aber bedenken Sie : $d_{free} \neq d_{min}$

Wichtig!

Anders als bei den Block-Codes oder einem BCH kann sich ein Trellis / Viterbi «erholen»

Damit ist folgendes gemeint : Treten max. t Fehler auf, kann der Trellis / Viterbi diese korrigieren und werden danach d_{free} Bits fehlerfrei empfangen, so hat sich der Trellis erholt und kann erneut max. t Fehler korrigieren!

(Viterbi-) Decoder

Der Decoder wird mit dem Viterbi Algorithmus realisiert. Am besten wir machen ein kleines Beispiel:

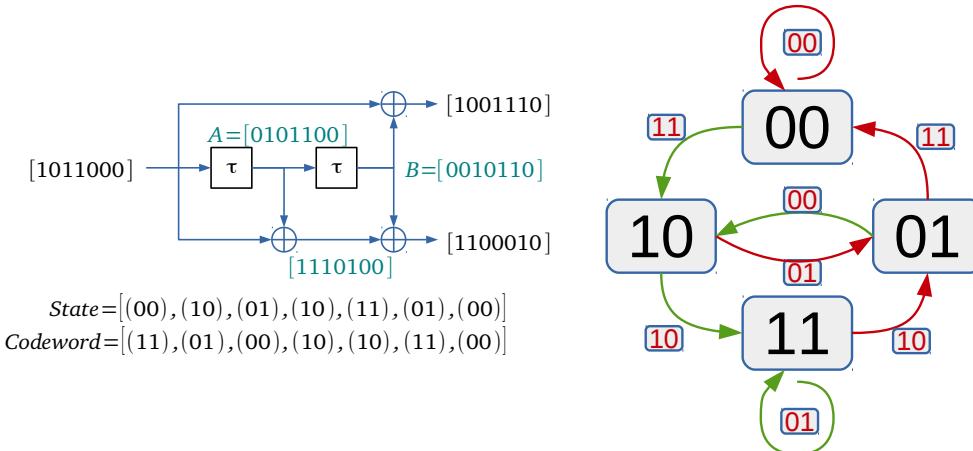


Abbildung 13.12: Trellis Encoder (Beispiel)

Wir brauchen zunächst unser Sendesignal. In Abbildung 13.12 sehen wir den Encoder und die dazugehörige Statemachine. Wir generieren unser Sendesignal. Die effektiv gesendeten Codewörter sind die Ausgänge des Encoders. Wir haben zusätzlich noch die States aufgeschrieben, aber dies nur, damit wir das Beispiel einfacher gegenprüfen können.

$$\text{Codeword} = [(11), (10), (00), (10), (11), (11), (00)]$$

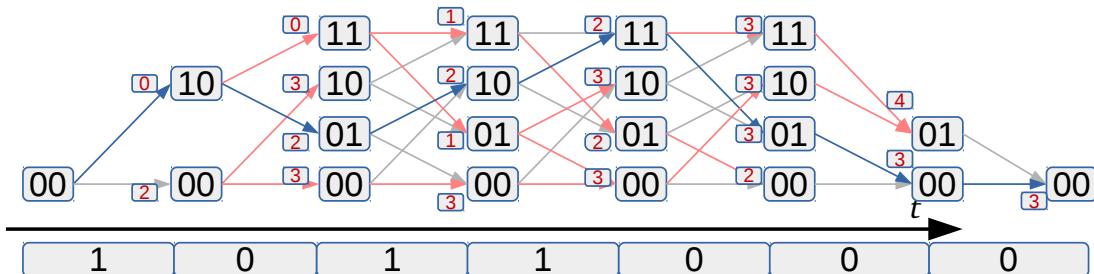


Abbildung 13.13: Trellis Diagramm

Zur Decodierung nehmen wir das Trellis Diagramm und weisen jeder Verbindung eine Metrik zu.

Metrik

Erinnern wir uns an Abbildung 13.11: Wir wissen, dass nicht jeder State-Übergang zulässig ist. Unsere Metrik ist zunächst $m = 0$. Für jeden unzulässigen State-Übergang inkrementieren wir die Metrik (inkrementieren : addieren eine 1). Es gewinnt die minimale Metrik.

Decodieren

Wir müssen nun zuerst die Anzahl d_{free} Bits empfangen. Danach bilden wir die Metrik und entscheiden uns für das empfangene Codewort.

Die Abschluss-Metrik gibt die Anzahl Fehler an. Man kann so, sogar bestimmen, wie verlässlich die Übertragung war. Dieses Wissen kann für die sog. *soft-decision* eingesetzt werden.

Vorteil

Sehr hohe Spektrale Effizienz, nahe an der Shannon-Kanal-Kapazität. Kleine Latenz, da schon vor einer abgeschlossenen Übertragung die Metriken bestimmt werden können. «Erholen des Algorithmus» nach längerer Fehlerlosen Übertragung.

Nachteil

Der Trellis / Viterbi ist komplex aufgebaut. Kann «catastrophic» werden. (nur noch Fehler)

13.6 Turbo-Code

Die Turbo-Codes sind eine Weiterentwicklung der Trellis / Viterbi Algorithmen. Es werden mehrere kurze Convolution Coders miteinander verkettet. Diese werden sog. interleaved (abwechselnd), daher immer abwechselnd der eine, dann der andere Encoder verwendet. Dadurch sinkt der Viterbi Decodieraufwand. Zusätzlich werden sie mit einem «äusseren» Reed-Solomon Code versehen.

Die Turbo Codes sind derzeit die besten Kanalcodierungen!

Vorteil

Noch näher an der Shannon-Kanal-Kapazität. Reduzierter Decodier-Aufwand im Vergleich zur Trellis / Viterbi Codierung.

Nachteil

Aufwändig.

14

Zugriffsverfahren

Inhalt des Kapitel

14.1 Master-Slave Prinzip	197
14.2 Polling	197
14.3 Token-Passing	198
14.4 Resource reservation	199
14.5 CSMA/CD	199
14.5.1 ALOHA-Algorithmus	202
14.6 CSMA/CA	203

Zum Inhalt

In diesem Kapitel werden wir die verschiedenen Zugriffsverfahren kennen lernen. Das Zugriffsverfahren steuert, wer, wann auf eine **Shared-Medium** zugreifen darf.

Nachdem nun alle Teilnehmer physikalisch miteinander verbunden sind und geregelt ist wie die Daten auf das Medium aufgetragen werden, muss noch deren Kommunikation untereinander geregelt werden.

Zugriffsverfahren

Das Zugriffsverfahren regelt,
wer und **wann** auf das Medium zugreifen darf

Haben wir nur eine einfache Verbindung, also nur 2 Teilnehmer, gestaltet sich das ganze recht einfach. Interessanter wird es bei mehr als zwei Teilnehmer. Wird ein angeschlossenes Übertragungsmedium von mehr als 2 Teilnehmer benutzt (oder mitbenutzt) wird dieses als *Shared-Medium* bezeichnet.

Bei einer Shared-Medium Topologie könnten mehrere Clients gleichzeitig versuchen auf das Medium zu-zugreifen. Verfälscht empfangene Daten sind die Folge. Um dies zu verhindern wurden verschiedene Zugriffsverfahren entworfen, so dass immer nur ein Teilnehmer auf das Medium zugreift.

Grundsätzlich gibt es drei Klassen von Zugriffsverfahren:

- Auswahl-Verfahren
- Reservierungs-Verfahren
- Zufalls-Verfahren

Oder man klassifiziert nach der Vorhersagbarkeit des Zugriffs:

- deterministisch
- nicht deterministisch

Für eine *Real-Time* Anwendung ist ein deterministisches Verhalten wünschenswert. Leider sind die nicht deterministischen Verfahren weiter verbreitet.

Auswahl-Verfahren

Zentral oder dezentral wird nach dem Ende einer Übertragung der nächste sendeberechtigte Teilnehmer bestimmt. Der Zeitpunkt des nächsten eigenen Zugangs zum Shared-Medium kann vom Benutzer nicht genau vorausberechnet werden. Beim Auswahlverfahren können keine Konflikte auftreten.

Reservierungs-Verfahren

Im Gegensatz zum Auswahlverfahren wird hier für jeden Teilnehmer eine gewisse Übertragungskapazität reserviert. Der Zeitpunkt des nächsten eigenen Zugangs zum Shared-Medium kann vom Benutzer genau vorausberechnet werden.

Zufalls-Verfahren

Die Teilnehmer können zu jeder Zeit auf das Shared-Medium zugreifen. Tun dies zwei oder mehr Teilnehmer zur selben Zeit, entstehen Kollisionen. Diese Kollisionen werden von den Teilnehmern erkannt. Somit müssen die kollidierten Nachrichten nochmals gesendet werden. Damit nicht wieder alle zur selben Zeit die Nachrichten erneut senden, bedarf es einer zusätzlichen Logik, welches das Wiederholen «zufällig» gestaltet.

Deterministisch

Der Zeitpunkt von nächsten Zugriff zum Shared-Medium ist bekannt und kann genau vorausberechnet werden.

nicht Deterministisch

Der Zeitpunkt von nächsten Zugriff zum Shared-Medium ist unbekannt.

14.1 Master-Slave Prinzip

Das Master-Slave Prinzip ist kein Zugriffsverfahren. Aber Polling und Token-Passing basieren darauf. Das Master-Slave Prinzip beschreibt, dass ein Teilnehmer der designierte Master ist. Der Master initiiert immer den Datentransfer und sendet eine Anfrage an die Slaves. Diese Antworten nur auf die Anfrage des Masters.

14.2 Polling

Polling ist eine Implementation des Master-Slave Prinzips. Ein Teilnehmer ist der Master und fragt seine Slaves nacheinander ab, ob sie neue Daten zum versenden haben.

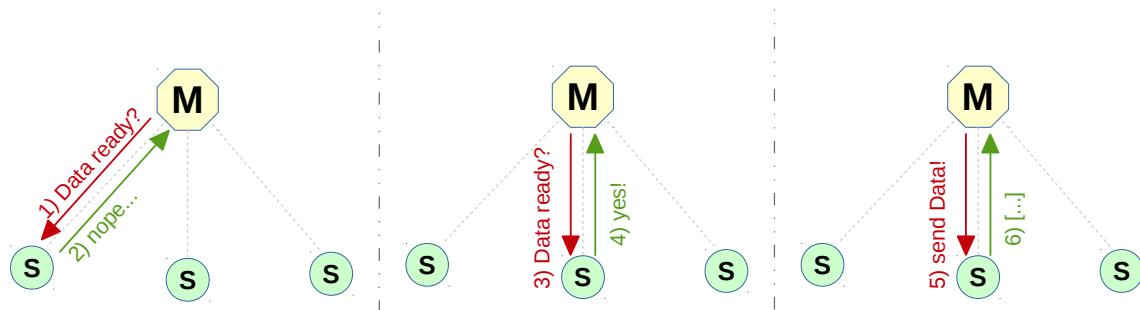


Abbildung 14.1: Beispiel Polling

Polling kann deterministisch oder nicht deterministisch sein, je nach Implementation. Polling wird zum Beispiel bei USB eingesetzt und besitzt auch einen Isochronous Mode. (deterministisch, aber Plesiochron)

Beim Polling ist der Master definiert und ändert nie. Die beste Performance erreicht Polling bei einer Stern-Topologie.

Vorteil

Einfaches Verfahren.

Nachteil

Der Zeitbedarf für das Abfragen jedes einzelnen Slaves, ob dieser denn neue Daten zu verschicken hat, ist schnell höher als die eigentliche Datenübermittlung.

14.3 Token-Passing

Für **Token-Passing** gibt es zwei Standards. Token-Ring sowie Token-Bus. Diese sind jeweils für Ring- / Bus-Topologie definiert. Token-Passing kann aber auch auf andere Topologien angewendet werden. Token-Passing beschreibt wie in einem gleichberechtigten Netz mithilfe von einem **Token** (engl. für «Jetton», Münze) eine *diskriminierungsfreie* Kommunikation realisiert werden kann. Zu Beginn sind alle Teilnehmer gleichberechtigt. Ist kein Token im Umlauf, wird mit einer Zufallszahl bestimmt wer einen Token generieren darf. Nur der Besitzer des Tokens darf Nachrichten versenden. (Der Tokenbesitzer ist der Master) Ist der Tokenbesitzer fertig, oder hat dieser ein Timeout erreicht, wird der Token an den nächsten Teilnehmer weiter gereicht. Der Token-Austausch wird nach dem *round-robin* Prinzip ausgeführt (gleichberechtigte Verteilung).

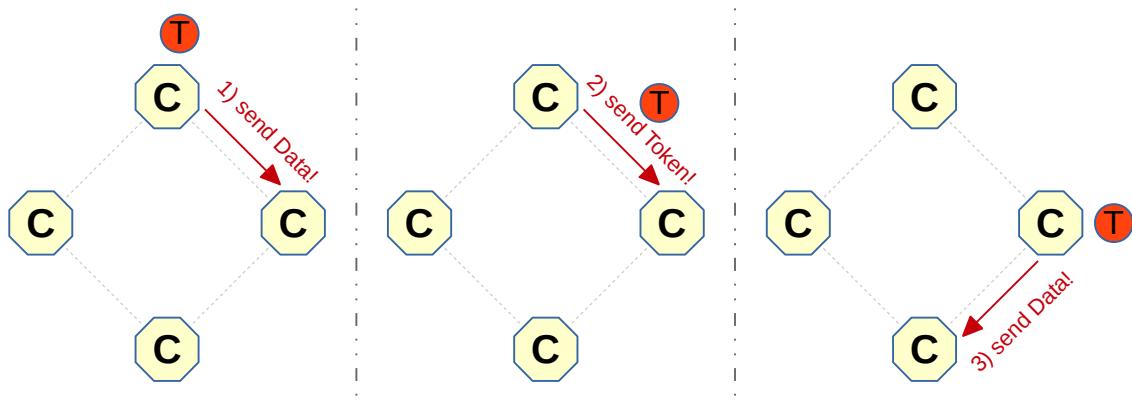


Abbildung 14.2: Beispiel Token-Passing

Token-Passing ist nicht deterministisch. Kann aber durch ein angepasstes Protokoll «recht» deterministisch gemacht werden.

Token-Passing wird beim Profibus Standard eingesetzt.

Vorteil

Hoher Datendurchsatz, keine idle Zeiten.

Nachteil

Lange Latenzen, bei kleinem Datendurchsatz. Es können zwei oder mehr Token im Umlauf sein, dies muss erkannt werden und die überschüssigen Token müssen eliminiert werden.

14.4 Resource reservation

Bei diesem Verfahren wenden wir entweder **FDMA** (Frequency Division Multiple Access) oder **TDMA** (Time Division Multiple Access) zur normalen Übertragung an. Damit haben wir ein deterministisches Verfahren.

Dies wird zB. bei Radio / Fernsehen eingesetzt.

Vorteil

Einzig Variante für «harte» Echtzeit Anforderung.

Nachteil

Ressource ist fest zugeteilt und kann nicht mehr «shared» benutzt werden.

14.5 CSMA/CD

CSMA/CD (Carrier sense multiple access / collision detection) gehört zu den Zufalls-Verfahren und ist nicht deterministisch.

Bei CSMA/CD sind alle Teilnehmer gleichberechtigt. Die Kommunikation ist nach dem *winner-takes-it-all* Prinzip realisiert. Es gehört daher zur Kategorie der *diskriminierenden* Kommunikation.

Wir haben die folgende Ausgangslage:

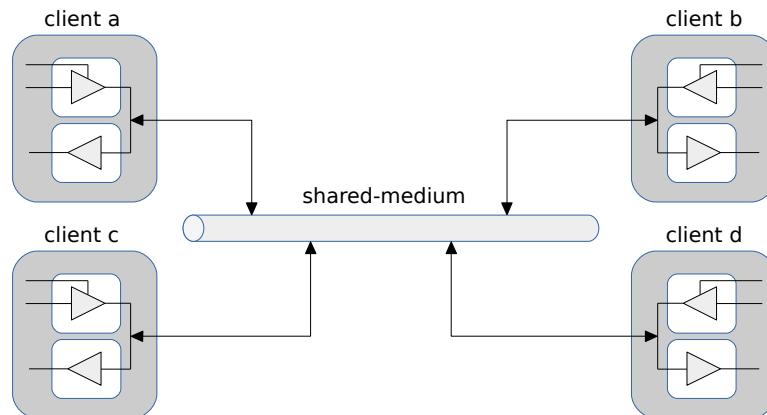


Abbildung 14.3: Ausgangslage CSMA/CD

In Abbildung 14.3 ist ein Netzwerk mit 4 Clients abgebildet. Alle können auf das gleiche Medium zugreifen, daher ist es ein Shared-Medium. Client A möchte nun auf das Medium schreiben. Es überprüft ob das Medium frei, indem es «hört» ob gerade auf dem Medium gesendet wird:

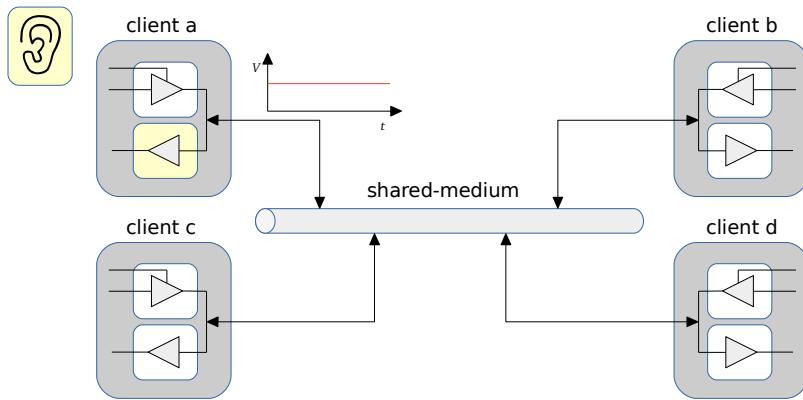


Abbildung 14.4: CSMA/CD Medium verkehr «mithören»
Ear Icon : Icon made by Freepik from www.flaticon.com

Ist das Medium frei, kann Client A senden.

Falls jemand am senden ist, wartet Client A bis die Übertragung abgeschlossen wurde und wartet bis das Medium frei ist.

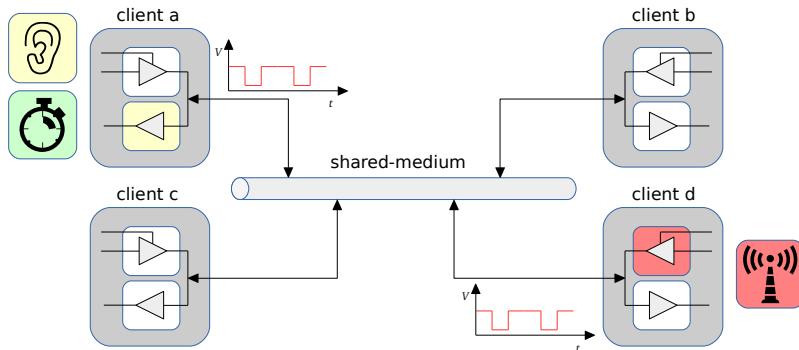


Abbildung 14.5: CSMA/CD shared-medium busy
Timer / Transmitter Icon : Icon made by Freepik from www.flaticon.com

Collision

Aber es kann passieren, dass Client A und D zur gleichen Zeit senden möchten. Dann werden beide Clients zuerst «hören» ob das Medium frei ist. Beide erkennen zur gleichen Zeit, dass das Medium frei ist und entscheiden sich dazu zu senden. Dabei werden beide gleichzeitig anfangen zu senden.

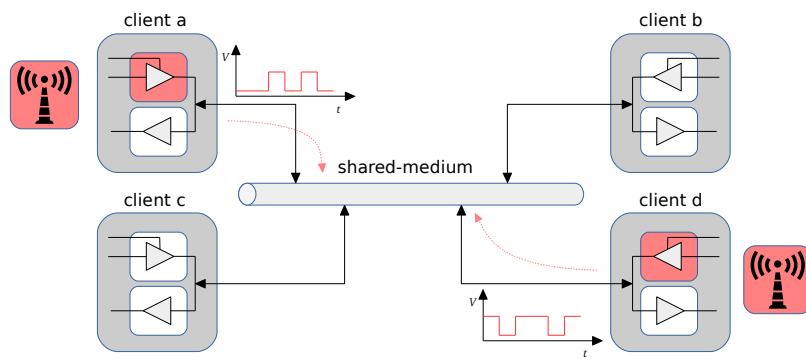


Abbildung 14.6: CSMA/CD gleichzeitiges Senden
Transmitter Icon : Icon made by Freepik from www.flaticon.com

Da die Signale eine gewisse Signallaufzeit auf dem Medium überwinden müssen, werden sich die Signale überlagern und an jedem Client eine eigene «Mischung» von beiden Sendesignalen erhalten.

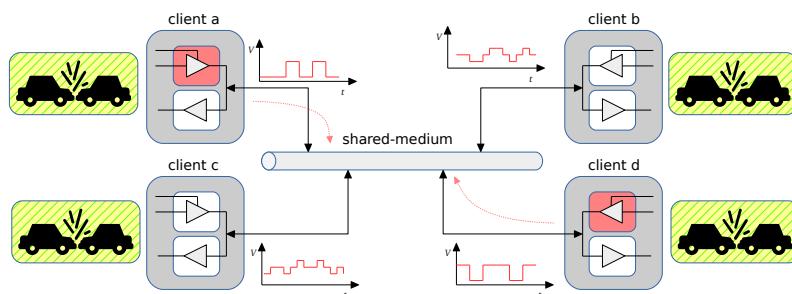


Abbildung 14.7: CSMA/CD Collision
Crash Icon : Icon made by Freepik from www.flaticon.com

Die Kollision von diesen Signalen wird entweder über eine Detektorschaltung oder über die Frame-Check-Sequence im Dataframe erkannt. (Siehe Kapitel 10 : Fehlererkennung / Kapitel 13 : Fehlerkorrektur)

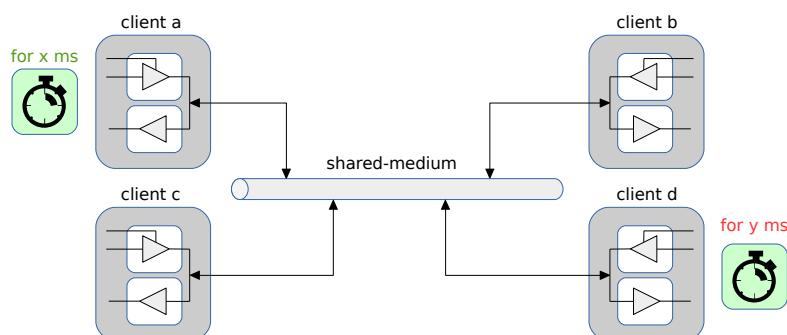


Abbildung 14.8: CSMA/CD recovery

Nach einer erkannten Kollision wird die Übertragung sofort abgebrochen und alle Teilnehmer wählen eine Zufalls-Zeit (backoff-time) und warten diese ab. Danach beginnt alles wieder von vorne. Da jeder eine eigene backoff-time «würfelt» ist sichergestellt, dass nicht nochmals eine Kollision entsteht. Das ganze sieht als Flussdiagramm folgendermassen aus:

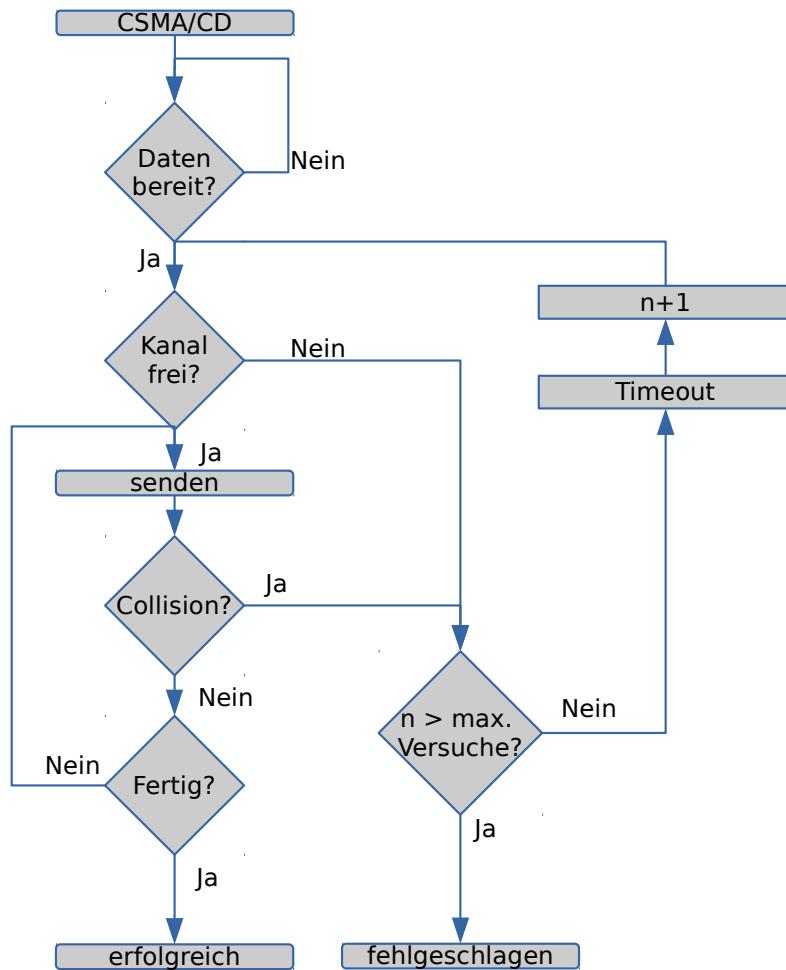


Abbildung 14.9: CSMA/CD Ablaufschema

14.5.1 ALOHA-Algorithmus

Damit nach einer Kollision, nicht wieder die gleiche Konstellation unter den Teilnehmern herrscht, wird nach dem ALOHA Prinzip gehandelt. Das heisst, das nach der Kollision, alle Teilnehmer auf dem Shared-Medium eine «backoff-time» (Wartezeit) abwarten. Der backoff ist dabei von einer Zufallszahl abhängig, so dass statistisch gesehen immer ein Teilnehmer schneller als alle andern ist. Entsteht schon wieder eine Kollision, wird der *backoff* mit einem zusätzlichen Faktor versehen, so dass alle Wartezeiten noch länger werden.

CSMA/CD mit ALOHA wird unter anderem bei WLAN, älteren Ethernet Standards, RFID, Bluetooth, CAN eingesetzt.

Vorteil

Bei kleiner Last, kurze Latenzen, da nur wenn Daten bereit sind, eine Übermittlung vorgenommen wird.

Nachteil

Wollen viele Teilnehmer auf einmal senden, entstehen grosse Latenzen, da viele Kollisionen. Die Netto-Datenrate bricht drastisch ein.

14.6 CSMA/CA

Collision sense multiple access / collision avoidance ist dem *CSMA/CD* sehr ähnlich.

CSMA/CD Es wird zunächst abgehört, ob der Kanal frei ist, danach das Datenpacket gesendet. Sendet zum gleichen Zeitpunkte ebenfalls ein Teilnehmer, entsteht eine Kollision. Beide Datenpackete sind verloren.

CMSA/CA

1. Es wird zunächst ein kurzes Handshake Packet gesendet
2. Alle Teilnehmer erkennen das Handshake Packet und warten ab
3. Der Sender sendet das (lange) Datenpacket. Er wird nicht von anderen Teilnehmern unterbrochen

Der Vorteil gegenüber dem CSMA/CD Verfahren ist, dass nur immer Handshake Packete kollidieren können. Die Handshake Packete sind extra kurz gehalten. Eine Kollision, blockiert den Kanal nur kurz. Während im CSMA/CD Verfahren auch lange Packete kollidieren, welche erst beim CRC-Check als «beschädigt» identifiziert werden.

Vorteil Damit wird der grösste Kritikpunkt von CSMA/CD angegangen, nämlich den Einbruch der Datenrate bei hoher Teilnehmerzahl (und Workload).

Derzeit wird CSMA/CA nur bei wenigen WLAN Access Points unterstützt, aber dürfte in den nächsten Jahren weitere Verbreitung finden.

15

Multiplexing

Inhalt des Kapitel

15.1 Frequency Division Multiplexing (FDM)	206
15.1.1 Frequency Hopping	207
15.2 Time Division Multiplexing (TDM)	207
15.3 Code Division Multiplexing (CDM)	208
15.3.1 Asynchrones CDM	210
15.4 Space Division Multiplexing (SDM)	212
15.5 Wavelength Division Multiplexing (WDM)	214
15.6 Multiplexing	215

Zum Inhalt

In diesem Kapitel lernen wir die verschiedenen Multiplexing-Varianten kennen. Mit Multiplexing ist das Verfahren gemeint, wie man mehrere Datenströme auf einen gemeinsamen *Kanal* zu übertragen.

Wie wir bisher gesehen haben, ist die Kanalkapazität unseres Übertragungsmedium beschränkt. Es ist ein «rares-Gut» und in den meisten Fällen sind wir daran interessiert, dieses möglichst effizient zu Nutzen. Häufig sind wir auch daran interessiert verschiedene Datenströme über einen gemeinsamen Kanal (engl. Channel) zu übertragen. Als Beispiel sei hier Radio, Fernsehen, Telefon oder TCP/IP genannt. Multiplexing bezeichnet nun den Vorgang, wie einzelne Datenströme (streams) auf einem gemeinsamen Medium verteilt werden. Demultiplexing ist das Inverse und beschreibt wie die einzelnen Datenströme wieder separiert werden.

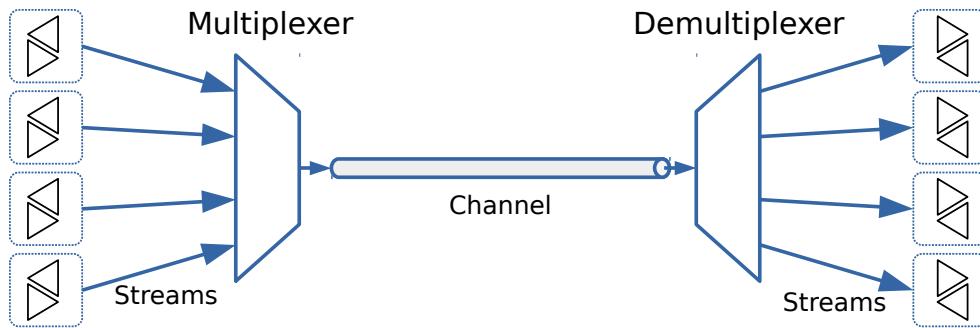


Abbildung 15.1: Multiplexer / Demultiplexer

15.1 Frequency Division Multiplexing (FDM)

Bleiben wir gleich beim Radio.

Wie Sie vielleicht wissen, können Sie selbst auswählen welchen Radiosender Sie gerade hören. Dies daher, da alle (regional-) verfügbaren Radiosender gleichzeitig über die Luft übertragen werden. Die Radiosender sind in der Schweiz über verschiedene Trägerfrequenzen verteilt.

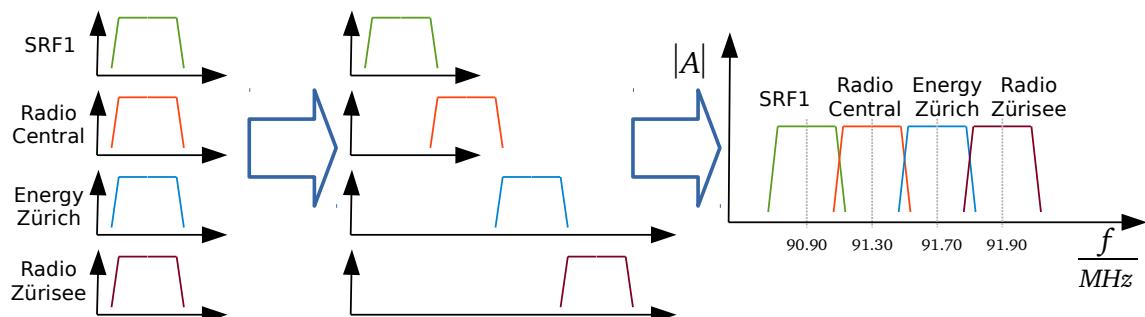


Abbildung 15.2: FDM : Radiofrequenzen Zürich

Diese Methode wird **Frequency Division Multiplexing (FDM)** genannt. Die Bandbreite des Kanals wird dabei auf die einzelnen Frequenzbänder verteilt. Mit FDM ist es möglich aus einer einfachen Halbduplex Verbindung, eine Vollduplex Verbindung mit geteilter Bandbreite zu realisieren. FDM eignet sich, wenn Sie mehrere Datenströme haben, welche einzeln nicht all zu hohe Datenraten aufweisen, aber einen stetigen (ev. Plesio-/ Isochronen-) Datenfluss haben. FDM wird bei vielen Standards eingesetzt. Als nicht abgeschlossene Aufzählung sei hier WLAN, GSM, Bluetooth, DOCSIS (Digitalfernsehen), ADSL (Internet) genannt.

Es gibt auch Standards (wie z.B. WLAN / Bluetooth) bei der sind die einzelnen Bänder definiert, aber nicht deren Belegung. Die einzelnen Devices müssen selbst feststellen, welcher Frequenzbereich frei ist und realisieren eine Verbindung darüber.

15.1.1 Frequency Hopping

Darüber hinaus werden bei Bluetooth V1 die Bänder in einer vorgegeben Pseudo-Zufalls Reihenfolge «angesprungen». Dieses Verfahren wird *Frequency Hopping* genannt. Bei Bluetooth wird zusätzlich noch eine sog. *Spread Spectrum Code Division Multiplexing* Variante angewendet. Damit lässt sich das Signal zusätzlich Spektral verschmieren und im Rauschen anderer Signale verstecken.

15.2 Time Division Multiplexing (TDM)

Anstatt die verfügbare Bandbreite aufzuteilen, können wir die Benutzungsdauer des Übertragungsmediums aufteilen.

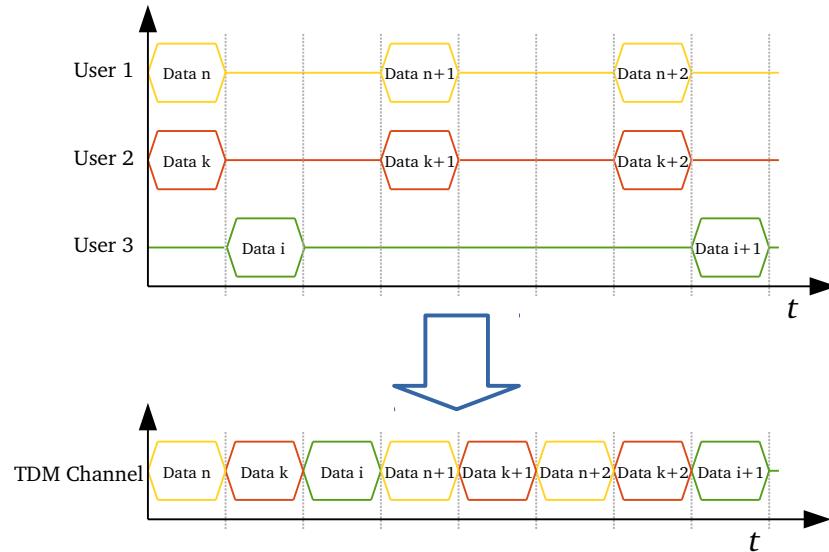


Abbildung 15.3: Time Division Multiplexing

In Abbildung 15.3 sind 3 Datenströme abgebildet, welche Plesiochron verlaufen (zwei davon Synchron). Diese übertragen nicht die ganze Zeit Daten. Man kann dies ausnutzen und in den jeweiligen Pausen ein anderen Datenstrom zwischen schieben.

Der TDM Kanal muss dabei nicht zwingend gleich «schnell» sein. Nehmen wir an, der TDM Kanal hat eine doppelt so hohe Datenrate wie die User-Kanäle in *Abbildung 15.4*.

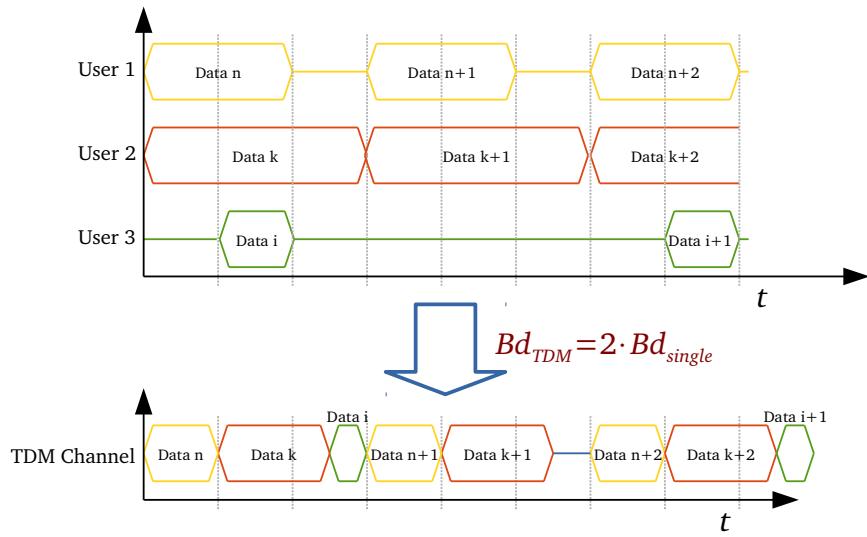


Abbildung 15.4: TDM : Asymmetrische Datenraten

Und siehe da, somit passen alle 3 User-Kanäle in den einen TDM Kanal.

Betrachten wir TDM als Baustein, dann ist das ein sog. Seriell / Parallel Wandler. TDM wird häufig bei seriellen Bussen eingesetzt. Zum Beispiel : CAN, PCIe, Ethernet, aber auch zur Separation von GSM Nutzern etc. Damit wir eine Halbduplex Verbindung realisieren können, wird ebenfalls TDM eingesetzt.

15.3 Code Division Multiplexing (CDM)

Code Division Multiplexing (CDM) ist eine recht «ausgefeilte» Multiplexing Art, die es erlaubt Signale zu überlagern.

Zunächst braucht man mehrere Überlagerungscodes, welche eine gute Orthogonalität zueinander aufweisen.

Vielelleicht erinnern Sie sich an die Vektorgeometrie. Zwei Vektoren sind orthogonal zueinander, wenn sie rechtwinklig zueinander stehen. Prüfen kann man die Orthogonalität indem man das Skalarprodukt bildet:

$$\begin{aligned} \langle x, y \rangle &= \sum x_i \cdot y_i \\ \langle x, y \rangle &\stackrel{?}{=} 0 \rightarrow \text{orthogonal} \end{aligned} \tag{15.1}$$

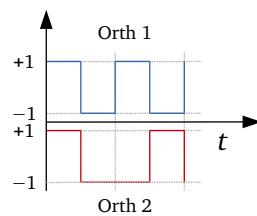


Abbildung 15.5: CDM Orthogonal Code

In *Abbildung 15.5* ist ein einfacher Orthogonaler Code abgebildet. Aber überprüfen wir doch rasch, ob der Code auch tatsächlich orthogonal ist:

$$\langle x, y \rangle \stackrel{!}{=} \sum x_i \cdot y_i = (1 \cdot 1) + (-1 \cdot -1) + (1 \cdot -1) + (-1 \cdot 1) = 1 + 1 - 1 - 1 = 0 \tag{15.2}$$

Machen wir nun ein Beispiel mit zwei Signalen:

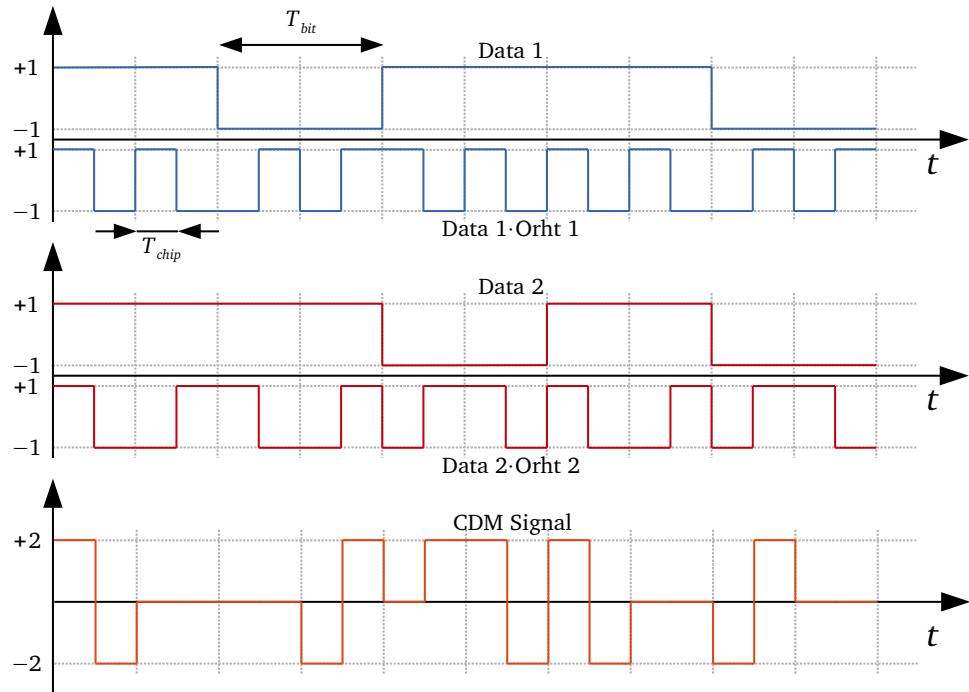


Abbildung 15.6: CDM : Modulation

Das CDM Signal in *Abbildung 15.6* sieht nun überhaupt nicht mehr wie das ursprüngliche Signal aus. Außerdem steigt der Bandbreitenbedarf, da die Frequenz des Datensignals f_{bit} auf die Chipfrequenz f_{chip} erhöht wurde.

Gehen wir nun zur Demodulation. Wir führen nun ein sog. Korrelation durch:

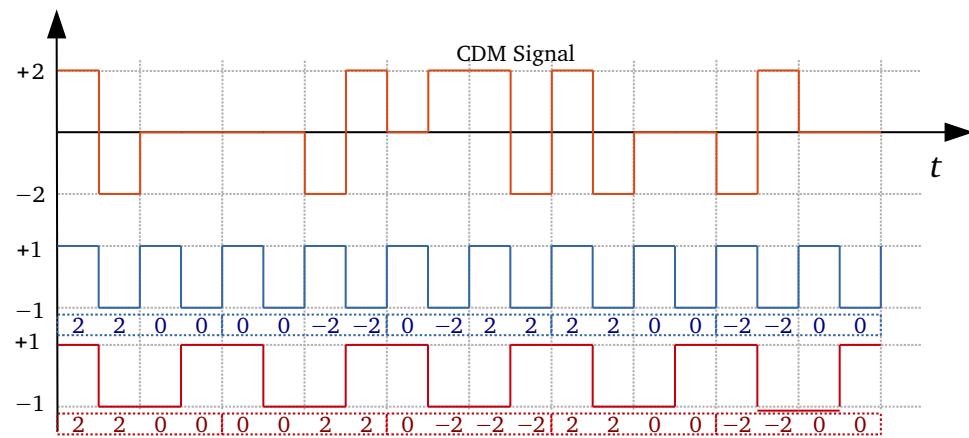


Abbildung 15.7: CDM : Demodulation

In *Abbildung 15.7* ist der Korrelationsvorgang abgebildet. An sich multiplizieren wir die CDM Sequenz mit unserem Orthogonalem Signal. Ist die Summe in einem Zeitintervall positiv, wurde eine +1 gesendet, ist sie negativ war es eine -1.

$$c_{korr} = \sum_{i=1}^{T_{bit}} x_i \cdot r_i \quad (15.3)$$

Und wie Ihnen vielleicht aufgefallen ist, ist die Berechnung der Korrelation eigentlich das *Skalarprodukt* des Empfangssignals mit der orthogonalen Code-Sequenz.

Spektral gesehen verschmieren wir das Sendesignal auf ein breiteres Spektrum.

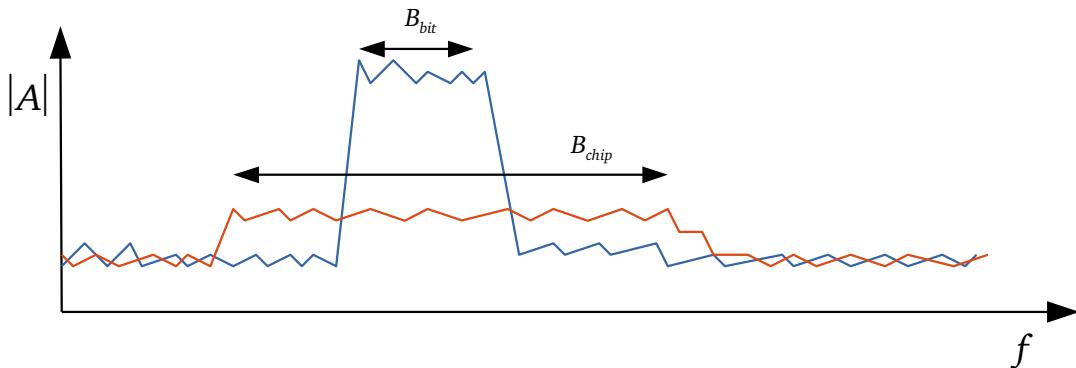


Abbildung 15.8: CDM : Spektrale Verschmierung

Die Spitzenleistung nimmt dabei um den Processing Gain G_p ab.

$$G_p = 10 \cdot \log \left(\frac{R_{Chip}}{R_{Data}} \right) \quad [G_p] = dB \quad (15.4)$$

Wählt man die Code-Sequenz lange genug, kann ein Signal auch ins Rauschen hinab gedrückt werden und auch wieder heraus korreliert werden.

Das sieht alles recht toll aus. Ich habe Ihnen aber noch etwas verschwiegen.

Genau so funktioniert es nur in den wenigsten Fällen. Orthogonale Signale müssen synchron zueinander laufen, ansonsten verlieren sie die Eigenschaft der Orthogonalität zueinander.

15.3.1 Asynchrones CDM

Wir suchen nun Code-Sequenzen die Orthogonal zueinander stehen, auch wenn diese zeitlich gegeneinander verschiebt. Und tatsächlich, es gibt solche Sequenzen, aber sie sind einerseits rar und nicht über die ganze Zeit perfekt orthogonal.

Solche Sequenzen nennt man Gold-Sequenz und können mit rückgekoppelten Schieberegistern erzeugt werden. Wenn man diese genauer untersucht, merkt man, dass es sich um Pseudo-Random Generatoren handelt. Aber wie gesagt, nicht jede Sequenz ist gut genug.

Machen wir ein «einfaches» Beispiel:

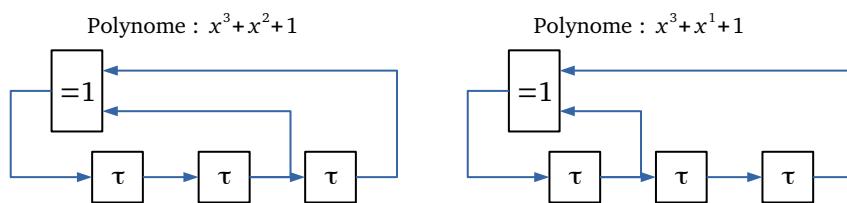


Abbildung 15.9: 3 Bit Pseudo-Random Generator

In Abbildung 15.9 sind zwei rückgekoppelte Schieberegister (*Linear-Feedback Shift Register* : LFSR) mit 3 Bit Wortbreite abgebildet. Diese bilden eine Pseudo-Random Sequenz der Länge $2^{(n-1)} = 7$. Für die CDM nehmen wir das hinterste Bit.

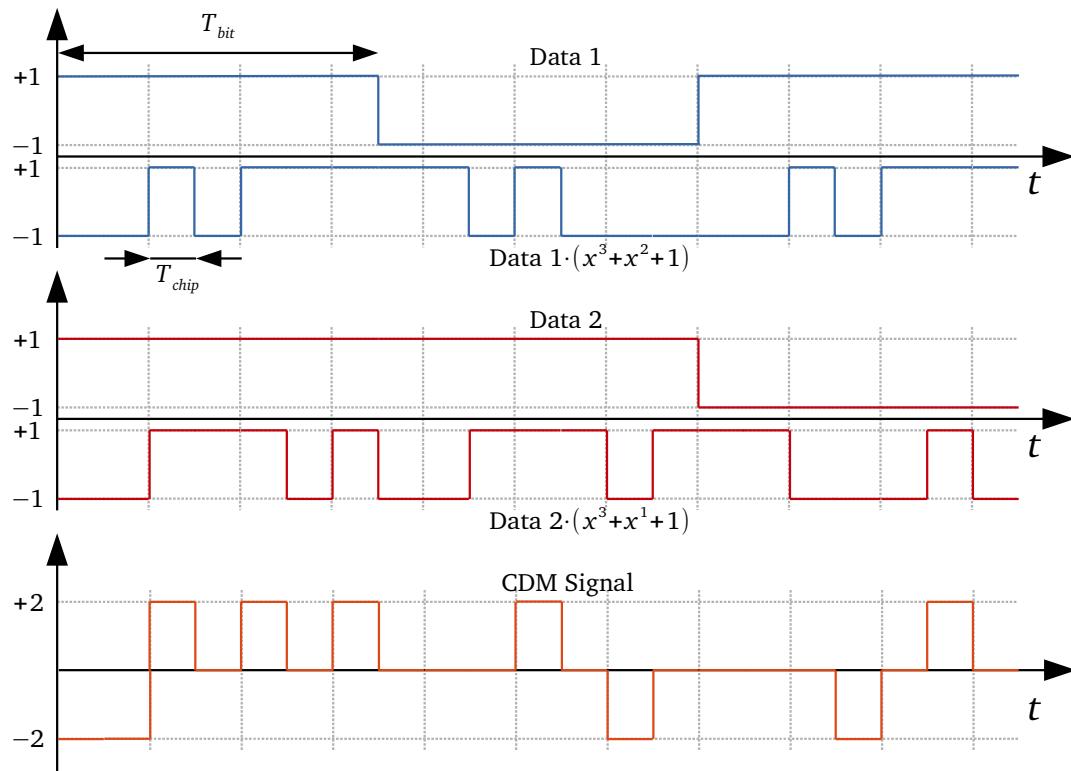


Abbildung 15.10: AC-CDM : Modulation

Damit bilden wir wieder eine Spreiz-Sequenz, wie im vorherigen Beispiel.

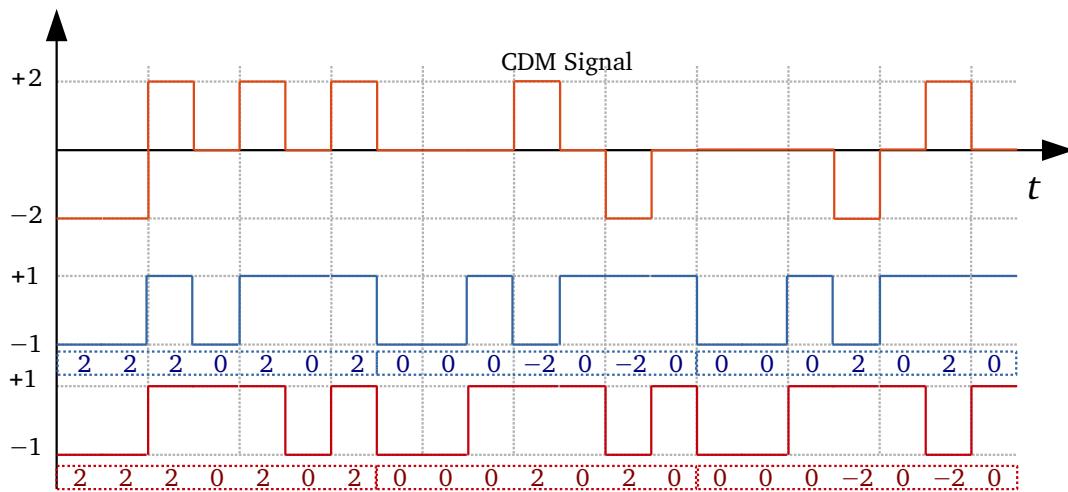


Abbildung 15.11: AC-CDM : Demodulation

Und zur Demodulation, korrelieren wir über die beiden *LFSR-Bit-Sequenzen*.

Und siehe da, es funktioniert!

CDM wird von verschiedenen Standards verwendet. Zum Beispiel Bluetooth, CDMA2000, ZigBee, GPS. Zu GPS ist noch interessant zu wissen, das zu Beginn GPS nicht öffentlich zugänglich war und das nur das amerikanische Militär Zugang zu GPS hatte, da es die benutzten Goldcodes kannte. Das Signal wird mit einem $f_{chip} = 1.023$ MHz Chiptakt getaktet. Damit liess sich das GPS Signal im Rauschen verstecken. Dieses System ist heute öffentlich und das Militär hat pro Satellit eine weitere CDM Sequenz mit 10 mal

höherer Chiprate und damit 10 mal höherer Genauigkeit.

Ein weiterer interessanter Punkt:

Im Weltall haben Sie grundsätzlich folgende Probleme : Es ist kalt, aber Sie bringen die Hitze nicht weg. Sie haben (fast) immer Sonnenlicht und damit Energie, aber können nicht viel (Spitzen-) Leistung generieren. Mit CDM wird das Frequenzspektrum verschmiert. Sie brauchen nicht so eine Spitzenleistung an der Senderendstufe. Das schont die Batterien und der Sender kann auf Effizienz optimiert werden, so dass dieser im Vergleich weniger Hitze generiert. Die Energiedichte bleibt aber die gleiche!

15.4 Space Division Multiplexing (SDM)

Bei Space Division Multiplexing (SDM) handelt es sich um Räumliche Trennung des Übertragungsmediums

Leitungsgebunden

Leitungsgebunden heisst SDM eigentlich, dass man mehrere Kabel verwendet. Für eine Vollduplex Verbindung braucht man also zwei Kabel, eines zum senden, eines zum empfangen.

Drahtlos

Interessanter wird SDM bei einer Drahtlos Kommunikation. Denn dort haben wir explizit nur 1 Übertragungsmedium. Aber wir machen uns der Eigenschaft zu Nutze, dass die Signale eines Senders mit der Distanz abnimmt. Ein Sender kann nicht das gesamte Übertragungsmedium auf einmal abdecken. Ist man von einem Sender weit genug entfernt, kann ein weiterer Sender diesen Bereich wieder abdecken.

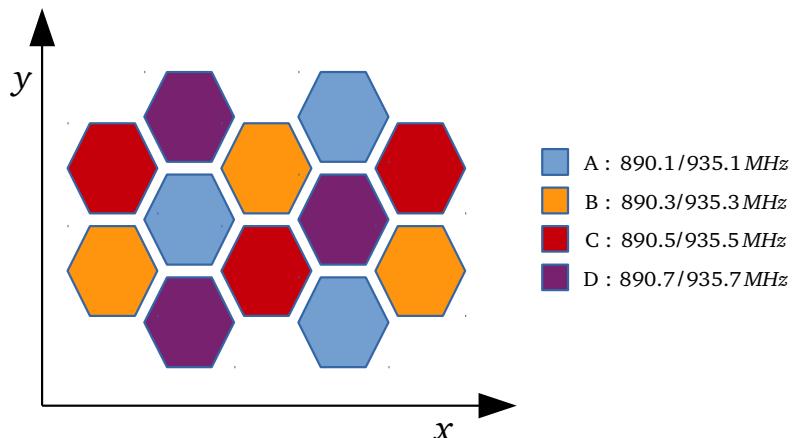


Abbildung 15.12: SDM : GSM Zellenverteilung

Beim Mobilfunk, aber auch beim WLAN haben wir eine begrenzte Anzahl FDM Kanäle. Damit sich diese nicht immer gegenseitig stören, werden beim Mobilfunk die einzelnen FDM Kanäle auf die einzelnen Basisstationen verteilt. Benachbarte Basisstationen verwenden nie die gleichen Kanäle wie benachbarte Zellen. Dies ist in Abbildung 15.12 abgebildet. Beim Mobilfunk hat man sich darauf geeinigt eine Art Hexagon Abdeckung zu realisieren.

Sehen wir uns noch kurz die WLAN 2.4GHz Frequenzbandverteilung an.

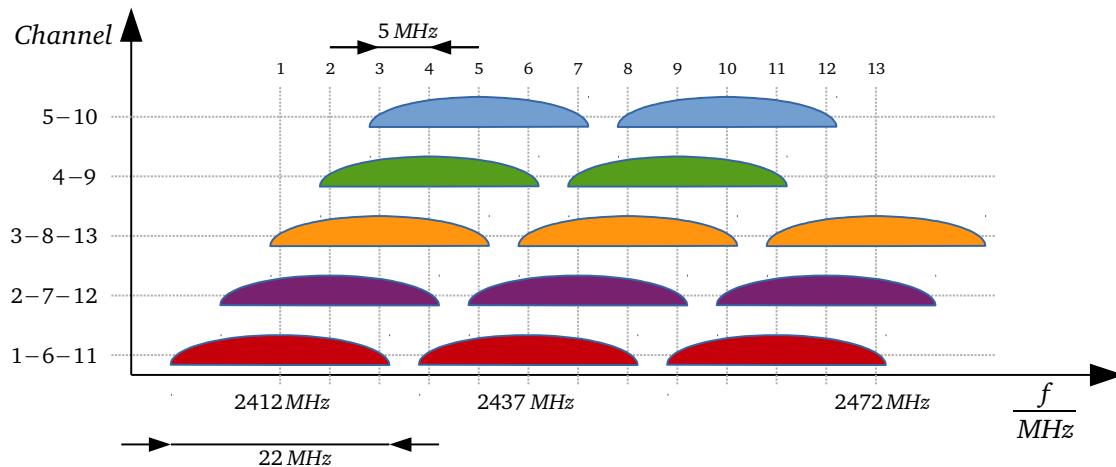


Abbildung 15.13: SDM : WLAN Frequenz Kanal Verteilung 2.4GHz

In Abbildung 15.13 ist alles Wissenswerte enthalten. Bei 2.4 GHz WLAN haben wir 13 Kanäle mit einem Frequenzabstand von $\Delta f = 5 \text{ MHz}$. Die Bandbreite pro Kanal beträgt allerdings $B_{CH} = 22 \text{ MHz}$. Die einzelnen Frequenzbänder sind überlappend. Zulässige Konfiguration pro WLAN Funkzelle sind allerdings nur die Gruppen [1 – 6 – 11] / [2 – 7 – 12] / [3 – 8 – 13] / [4 – 9] / [5 – 10].

Aufgabe 15.60

Frage

Haben Sie ihr WLAN zu Hause richtig eingerichtet?

Antwort

Sind die Kanäle zu nah beieinander aktiv, stören sie sich gegenseitig und reduzieren die Netto-Datenrate von beiden Kanälen.

15.5 Wavelength Division Multiplexing (WDM)

Wavelength Division Multiplexing (WDM) lässt sich nur auf die optischen Übertragungsmedien anwenden. Stellen sie sich WDM wie FDM mit Farben vor. Sie haben bei der optischen Übertragung als Sender Laserdioden (oder LED's), welche je nach Bauart unterschiedliche Wellenlängen aussenden. Bei WDM verwendet man mehrere Laserdioden mit unterschiedlichen Wellenlängen und leitet diese in einen Lichtleiter. Die einzelnen Wellenlängen sind unabhängig voneinander und stören einander kaum.

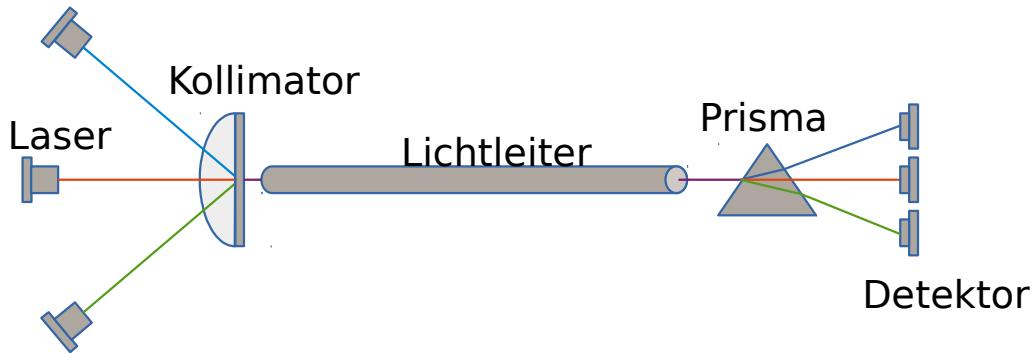


Abbildung 15.14: WDM : Aufbau (stark vereinfacht)

In Abbildung 15.14 ist ein optischer Übertragungsaufbau mit mehreren Laserdioden beschrieben. (stark vereinfacht) Zunächst braucht man Laserdioden mit unterschiedlichen Wellenlängen, einen Kollimator welcher alle 3 Lichtstrahlen auf das Medium leiten kann. Auf der Austrittsseite braucht es mind. ein Prisma oder ein Beugungsgitter um das Licht wieder in die einzelnen Wellenlängen aufzuteilen und 3 Detektoren für die jeweilige Wellenlänge. All diese Komponenten sind teuer und nicht trivial aufzubauen. WDM lohnt sich daher nur Systeme mit extremen Datendurchsatz. (zB. Interkontinental Datenverkehr)

Polarization Devision Multiplexing (PDM)

Wie sie vielleicht wissen ist Licht polarisiert. Das heisst, Licht hat eine Drehrichtung und somit auch eine Phase. Es gibt Filter welche nur polarisiertes Licht mit einer bestimmten Phase durchlassen. zB. passive 3D Kino Brille. Und wie im 3D Kino können wir für einzelne Detektoren das Licht nach der Polarisation aufteilen.

15.6 Multiplexing

Wir haben nun die wichtigsten *Multiplexing*-Varianten erarbeitet. Die heutigen Kommunikations-Standards verwenden häufig gleich alle Multiplex-Verfahren in der einen oder anderen Weise.

Werden die Multiplex Verfahren eingesetzt um mehrere physische Kommunikations-Teilnehmer auf dem gleichen Medium-Zugriff zu ermöglichen, so spricht man von sog. Multiple-Access. TDM wird dann als → *time division multiple access (TDMA)* eingesetzt. Analog hierzu CDM → *code division multiple access (CDMA)*, FDM → *frequency division multiple access (FDMA)*.

Zum Abschluss noch eine Grafik wie TDM / FDM / CDM miteinander verwandt sind.

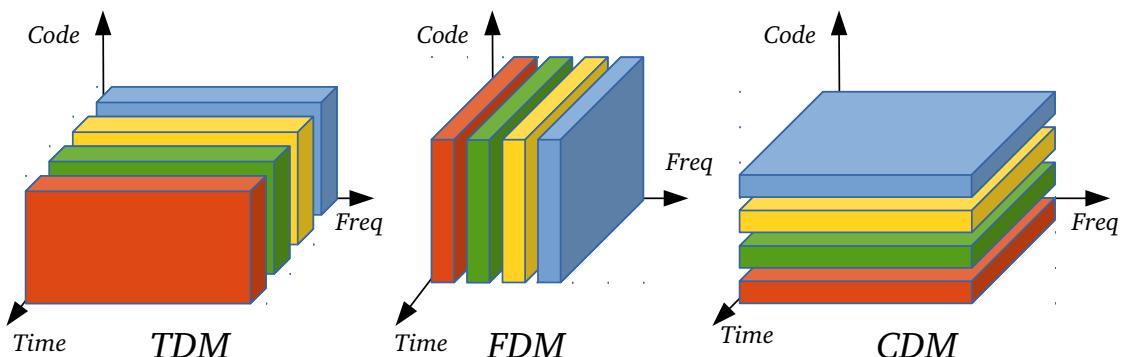


Abbildung 15.15: Multiplex Verfahren TDM / FDM / CDM

Und so sieht das Ganze kombiniert aus.

Dies wird beispielsweise im Mobilfunk auch tatsächlich gemacht. Hier werden alle unsere Multiplexverfahren, parallel genutzt um mit einer Basisstation möglichst viele Clients bedienen zu können.

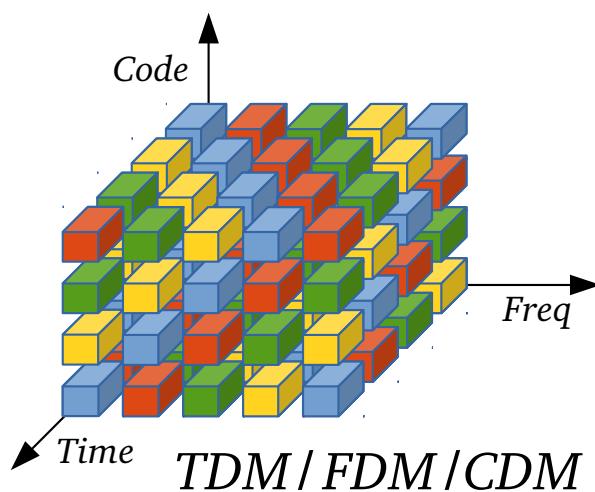


Abbildung 15.16: Multiplexing : TDM / FDM / CDM Combined

16

Flusssteuerung

Inhalt des Kapitel

16.1 ACK / NACK	218
16.2 RTS / CTS	220
16.3 Feedback Error Control (ARQ)	221
16.4 Verbindungsprotokolle	222

Zum Inhalt

Wir wissen nun wie Daten gesendet werden und wer sendet. Aber ist der Empfänger auch bereit? Und hat er alles richtig erhalten? Hier setzt die Flusssteuerung an, auch *flow control* genannt. Im OSI-Layer 2 Protokoll wird beschrieben, wie die Flusssteuerung umgesetzt wird. In den meisten Fällen wird ein sog. *ACK/NACK (Acknowledge / not Acknowledge) Handshaking* eingesetzt. Die Flusssteuerung wird häufig mit der **forward error control (FEC)** kombiniert.

16.1 ACK / NACK

Bei Acknowledge / Not Acknowledge sind zwei Symbole definiert. Sie ahnen es : **ACK / NACK**.

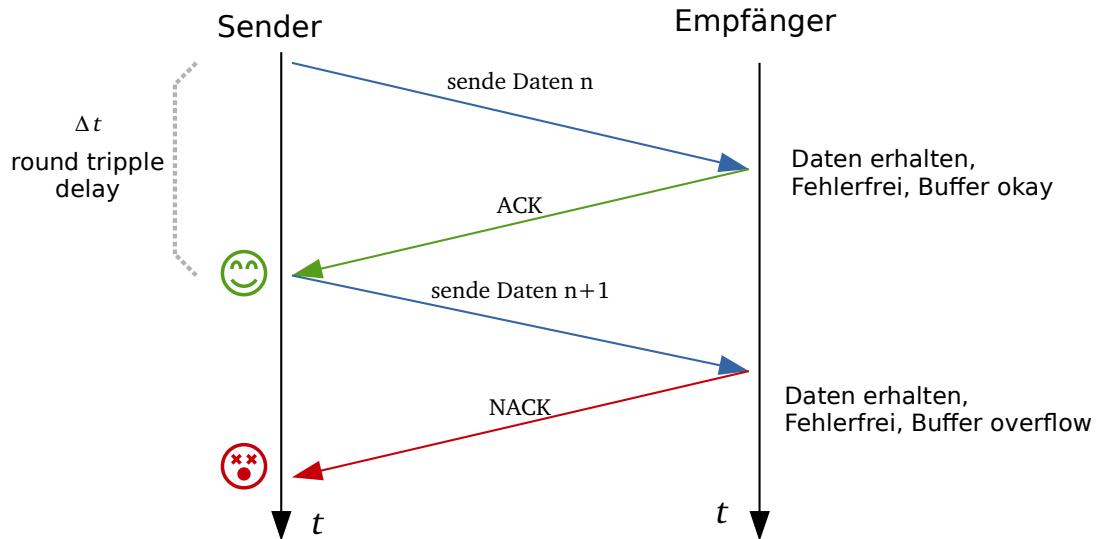


Abbildung 16.1: ACK / NACK einfaches Beispiel

In der einfachsten Variante sendet der Sender seine Daten. Hat der Empfänger alles richtig erhalten und kann die Daten annehmen, wird ein ACK zurück gesendet. Sind irgendwelche Fehler aufgetreten, z.B. Daten korrupt oder ist der Eingangsbuffer des Empfängers überfüllt, wird das NACK Symbol zurückgeschickt.

Was nach einem NACK geschieht ist Gegenstand des Protokolls.

I2C

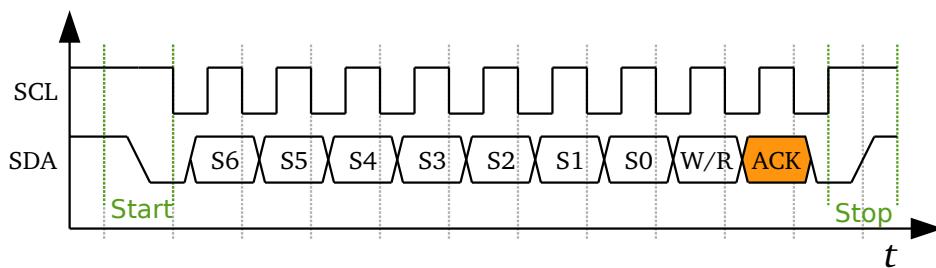


Abbildung 16.2: I2C Protokoll

Wie in Abbildung 16.2 abgebildet, ist bei der I2C Schnittstelle extra ein ACK Bit-Feld vorgesehen.

USB

Im USB Standard ist extra ein Handshake Packet vorgesehen. Also ein Datenpaket das nur für die Flusssteuerung vorgesehen ist.

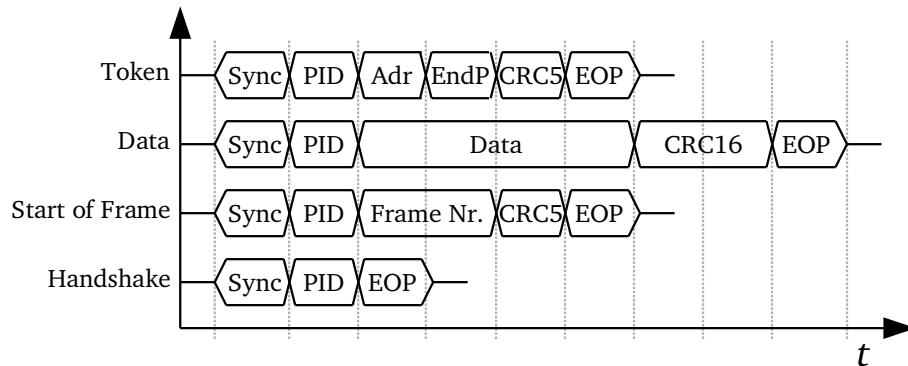


Abbildung 16.3: USB Packet Types

Es gibt folgende Handshake PID's:

PID Value	Packet Identifier
0x2	ACK
0xC	NAK
0xE	STALL
0x6	NYET (No Response Yet)

Tabelle 16.1: USB Handshake Packet PID's

Sie sehen : Auch USB verwendet eine ACK/NACK Flusssteuerung.

16.2 RTS / CTS

Als weiter Variante gibt es die ready-to-send / clear-to-send Flusssteuerung.

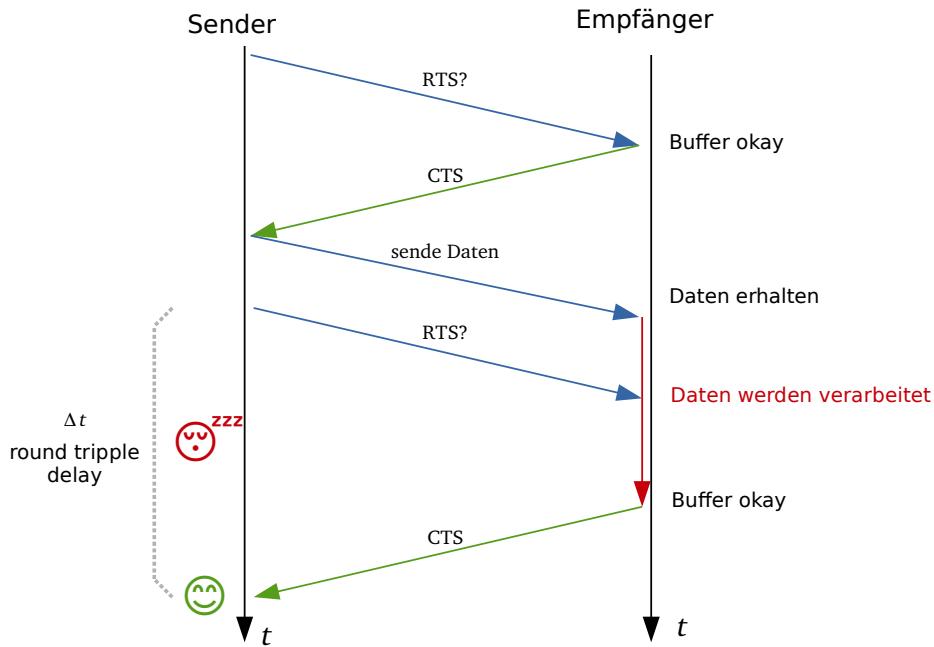


Abbildung 16.4: CTS / RTS einfaches Beispiel

Bei dieser Variante fragt der Sender vor der Übertragung mit einem *ready-to-send* ab, ob der Empfänger bereit ist. Dieser gibt mit dem *clear-to-send* Signal die Freigabe zur Übermittlung.

Mit dieser Variante nimmt der sog. *round-trip delay* im Vergleich zu ACK/NACK zu. Dieser gibt die Zeit an, welche benötigt wird um eine Übertragung durchzuführen, bis wieder die nächste Freigabe gegeben wird. Dies dadurch, da erst wieder neue Daten gesendet werden können, wenn der Empfänger bereit ist. Dauert die Übertragung vergleichsweise lange, wird hier viel Zeit vergeudet.

RS232

Zu einer Zeit, als RS232 nur Halbduplex Verbindungen unterstützte, wurde zur Flusskontrolle RTS / CTS eingesetzt. Ein ACK / NACK muss auf Software Ebene realisiert werden.

RS485

Hat keine Flusskontrolle definiert. Diese muss per Software realisiert werden.

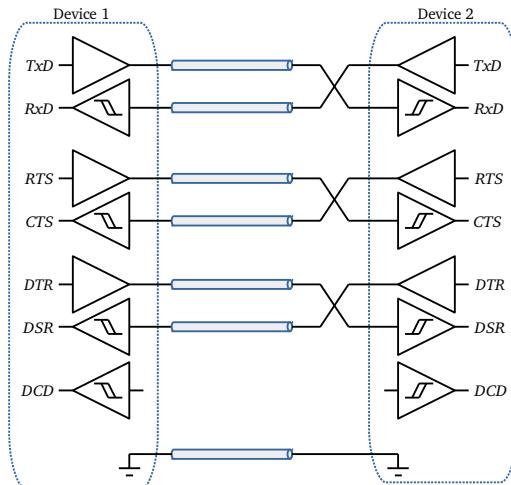


Abbildung 16.5: RS232 : RTS / CTS

16.3 Feedback Error Control (ARQ)

Feedback error control beschreibt Techniken wie mit Fehlern umgegangen wird. Meistens wird hierzu das *automatic repeat request ARQ* Protokoll eingesetzt welches auf *ACK / NACK* aufbaut. ARQ beschreibt drei Verhalten bei Fehlern:

- Stop-and-wait ARQ
- Go-back-N ARQ
- Selective-repeat ARQ

Stop-and-wait ARQ

Bei *Stop-and-wait ARQ* wird auf ein *ACK / NACK* gewartet. Wird ein *ACK* erhalten ist alles OK. Wird ein *NACK* erhalten, werden die Daten noch einmal gesendet. So lange bis ein *ACK* oder die maximale Anzahl Übertragungsversuche aufgebraucht sind.

Der grösste Nachteil an diesem Verfahren ist, dass wenn viele Pakete gesendet werden. Immer alle Pakete aufgehalten werden, wenn eines nicht übertragen wurde.

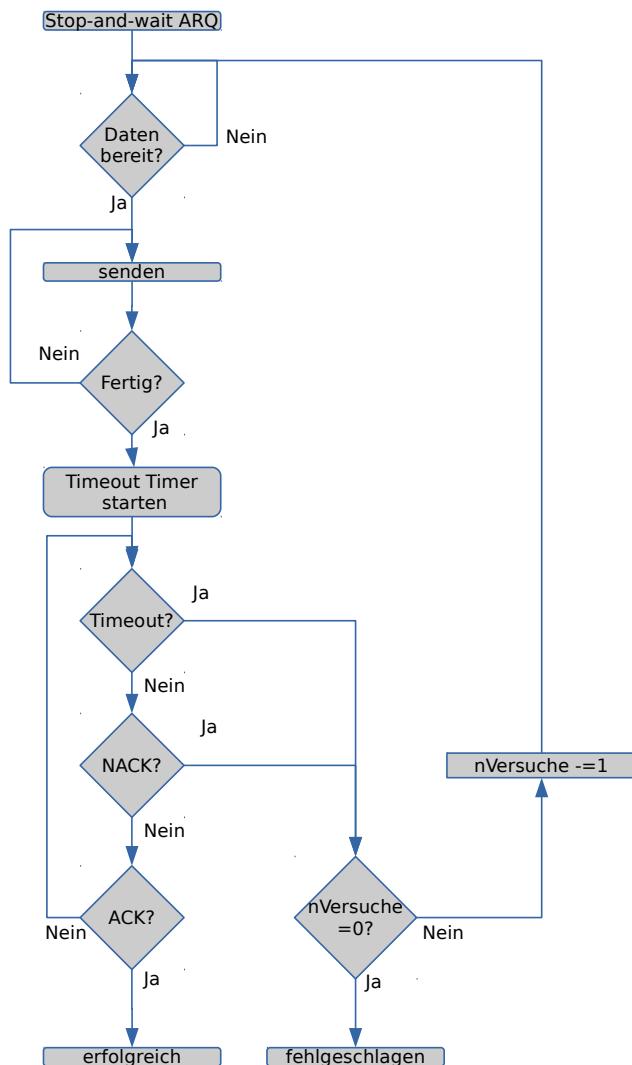


Abbildung 16.6: Stop-and-wait ARQ : Ablaufschema

Go-back-N ARQ

Bei *go-back-n ARQ* hat man *stop-and-wait ARQ* so modifiziert, dass immer max. N Datenpakete gesendet werden und gemeinsam mit einem *ACK/NACK* bestätigt werden. *Go-back-N ARQ* mit $N = 1$ entspricht *stop-and-wait ARQ*.

Die Anzahl N Datenpakete kann adaptiv auf den Kanal angepasst werden.

Go-back-N ARQ skaliert gut bei wenig Fehlern und ermöglicht einen grossen Datendurchsatz mit kleinen Latenzen. Werden aber viele Fehler erkannt, wird die Anzahl Datenpakete pro *ARQ* reduziert, womit auch die Datenrate einbricht.

Selective-repeat ARQ

Bei *selective-repeat ARQ* werden die Datenpakete nummeriert und die *ACK / NACK* werden bei der Bestätigung ebenfalls mit der Datenpaket Nummer versehen. Damit kann jedes Paket einzeln nochmals angefordert werden.

Selective-repeat ARQ ist quasi das non-plus-ultra der *ARQ* Verfahren. Aber die meisten OSI-Layer 1 Protokolle unterstützen nur ein einfaches *ACK / NACK* Signal.

Man kann dies mit Software trotzdem realisieren. Die Nummerierten *ACK / NACK* müssen in einem Datenpaket übermittelt werden. Allerdings leidet so die Netto-Datenrate und die Latenz der Übertragung.

16.4 Verbindungsprotokolle

Einige Beispiele bekannter Verbindungsprotokolle, welche ebenfalls eine Flusskontrolle definieren.

Bit-orientiert

- HDLC (Highlevel Data Link Control)

Zeichen-orientiert

- Kermit
- BSC (Binary Synchronous Communication)
- ARPANET

Kermit (Simplex)

Das Kermit-Protokoll wird für Point to Point Verbindungen verwendet. Dabei muss der eine Teilnehmer als Sender und der Andere als Empfänger konfiguriert werden. Bsp.: Taschenrechner

BSC (Halbduplex)

Das BSC-Protokoll (Binary Synchronous Communication) wird für Multicast Verbindungen verwendet. Im Netz existiert ein Master und n Slaves. Der Master pollt die Slaves, dh. er fragt die Slaves der Reihe nach ab, ob sie eine Meldung haben.

ARPANET (Vollduplex)

Das ARPANET ist der Vorläufer des Internets.

HDLC (Vollduplex)

Das HDLC -Protokoll (Highlevel Data Link Control) wird für Point to Point, Multicast und Broadcast Verbindungen verwendet. Das HDLC Protokoll, welches aus dem SDLC (Synchronous Data Link Control) Protokoll entstand, ist heute sehr verbreitet (Modem, LAN, ISDN, X.25, ...).

Abschluss OSI Layer 1 /2

Seit *Kapitel 5 : Kanal / Leiter* haben wir uns intensiv durch den OSI-Layer 1/2 gearbeitet. Wir haben die Grundlagen vom Kabel bis zur Fehlerkorrektur und Flusskontrolle erarbeitet.

In den nächsten Kapiteln folgt nun OSI Layer 3/4, diese sind abstrakter, da wir nun nur noch mit «rein digitalen» Datenpacketen arbeiten. Diverse Themen wie Checksummen oder Flusskontrollen werden wieder auftauchen, daher mein Rat : lesen Sie zuerst die vorhergehenden Kapitel bevor Sie mit dem Skript weiterfahren.

17

OSI Layer 3 : Network Layer

Inhalt des Kapitel

17.1 Routing Schemata	226
17.1.1 Unicast	226
17.1.2 Multicast	226
17.1.3 Broadcast	227
17.1.4 Anycast	227
17.1.5 Geocast	227
17.2 Vermittlung von Daten	228
17.2.1 Verbindungsorientiert	228
17.2.2 Verbindungslos	228
17.2.3 Leitungsvermittlung (Verbindungsorientiert)	229
17.2.4 Zellenvermittlung (Verbindungsorientiert)	229
17.2.5 Paketvermittlung (Verbindungslos)	230
17.2.6 Paketvermittlung (Verbindungsorientiert)	231
17.3 Verbindung von Netzen	233
17.3.1 Repeater (Hub)	233
17.3.2 Bridge (Switch)	233
17.3.3 Router	235
17.3.4 Gateway	236
17.4 Internet Protocol (IP)	236
17.4.1 Datagram	236
17.4.2 IP Adresse	238
IPv4 IP Adresse	238
17.4.3 Wieso MAC- und IP-Adressen?	241

Zum Inhalt

OSI Layer 3 wird auch Network Layer oder Vermittlungsschicht genannt und beschreibt wie die Daten den Weg durch ein Netz mit mehreren Knoten findet. Je nach Routing-Schemata muss der Network Layer Datenpakete duplizieren und an verschiedene Adressen senden (Point-Multipoint Verbindung).

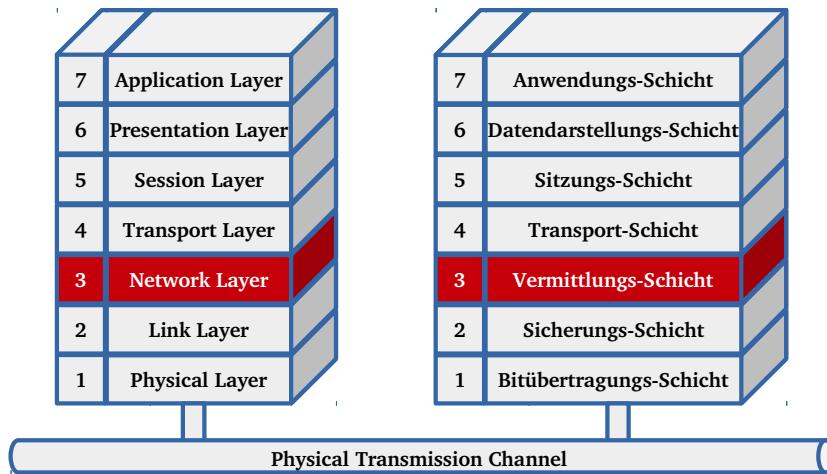


Abbildung 17.1: OSI Layer 3 (Network Layer)

17.1 Routing Schemata

17.1.1 Unicast

Routen wir Datenpakete von einer Quelladresse zu (genau) einer Zieladresse, welche durch eine einzelne eindeutig identifizierbare Adresse beschrieben wird, so nennen wir dies eine *unicast*-Übertragung.

Beispiel : Telefon zu Telefon Verbindung. Client zu Server etc.

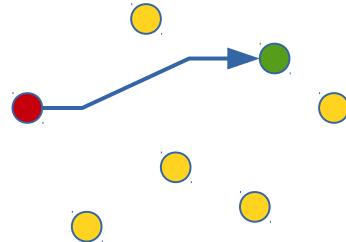


Abbildung 17.2: unicast

17.1.2 Multicast

Routen wir ein Datenpaket von einer Quelladresse zu einer Gruppe von Zieladressen, so nennen wir das eine *multicast*-Übertragung.

Diese kann noch in *application-layer-* und *networt-layer-multicast* unterteilt werden. Bei Multicast wird von der Quelladresse tatsächlich nur ein Datenpaket abgesetzt. Die Aufteilung in mehrere einzelne Datenpakete übernimmt das «Netz».

Beispiel: Internet Fernsehen, Internet Radio, «Push» Technologien

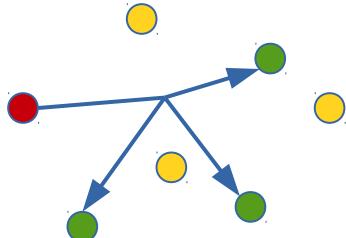


Abbildung 17.3: unicast

17.1.3 Broadcast

Routen wir ein Datenpaket von einer Quelladresse zu allen Netzteilnehmern in der «Broadcast-Domain» nennen wir dies eine *broadcast*-Übertragung.

Im Gegensatz zu multicast, wird das gesamte Netz (in der *broadcast domain*) mit dem Datenpaket beliefert.

Broadcast wird von IPv4, aber nicht mehr von IPv6 unterstützt. Stattdessen wird auf *multicast*-Übertragungen gesetzt, um alle (nicht betroffenen) Teilnehmer vor einer übermässigen Auslastung zu schützen.

Beispiel : Finden eines DHCP Servers.

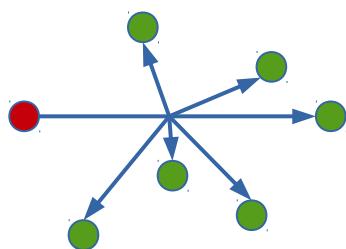


Abbildung 17.4: broadcast

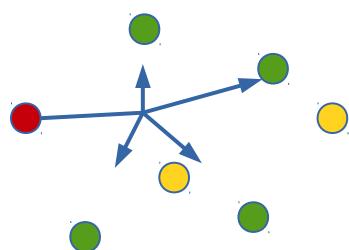


Abbildung 17.5: anycast

17.1.4 Anycast

Wird ein Datenpaket an die topologische nächste Zieladresse einer bestimmten Gruppe geroutet, spricht man von einer *anycast*-Übertragung.

Beispiel: DNS Anfrage an Root-Server, Suchmaschinen-Anfrage

17.1.5 Geocast

Wird eine Datenpaket von einer Quelladresse an eine Gruppe gesendet wird, welche sich durch ihren geografischen Ort definiert, spricht man von einer *geocast*-Übermittlung.

Geocast ist im wesentlichen eine spezielle *multicast*-Übertragung und wird vor allem in Routing Protokollen für sog. *mobile-ad-hoc* Netzwerke gebraucht.

Beispiel : SMS Unwetter Dienst, Katastrophenstelle, Regionale Tarif SMS etc.

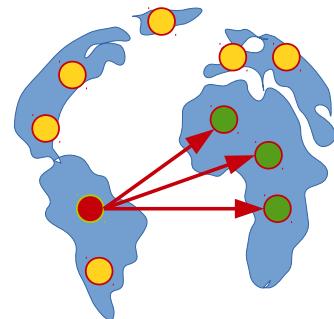


Abbildung 17.6: geocast

17.2 Vermittlung von Daten

Wir betrachten nun wie Daten in einem Netz vermittelt werden. Jeder Teilnehmer welcher am Routing beteiligt ist wird auch Knoten genannt.

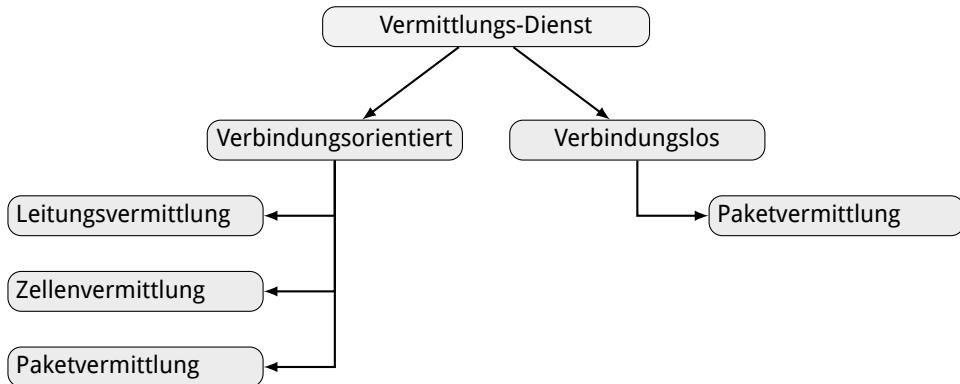


Abbildung 17.7: Vermittlung von Daten

Wie wir in Abbildung 17.7 sehen, können wir Daten entweder Verbindungsorientiert oder Verbindungslos übermitteln. Die Paketvermittlung ist bei beiden Varianten angegeben.

Wichtig

Die Paketvermittlung selbst ist ein Verbindungsloser Dienst, kann aber im OSI-Layer 3/4 Protokoll zu einem Verbindungsorientierten Dienst aufgewertet werden.

17.2.1 Verbindungsorientiert

Bei einer verbindungsorientierten Vermittlung wird zuerst eine Verbindung zwischen den Knoten aufgebaut, um anschliessend über diese Verbindung zu kommunizieren. Nach dem Ende der Kommunikation wird diese Verbindung wieder abgebaut. Somit wird während der Kommunikation immer über ein und **dieselbe, statische** Verbindung kommuniziert.

17.2.2 Verbindungslos

Bei einer verbindungslosen Vermittlung ist es nicht nötig zu Beginn der Kommunikation eine Verbindung aufzubauen, da jede vermittelte Information die gesamte Adresse des Empfängers beinhaltet. So mit wird während der Kommunikation nicht über eine statische Verbindung kommuniziert, sondern über **irgendeine** Verbindung im Netz. Dadurch ist es möglich, dass jede übermittelte Information einen anderen Weg im Netz nimmt.

17.2.3 Leitungsvermittlung (Verbindungsorientiert)

Bei der Leitungsvermittlung wird eine physikalische und oder logische Verbindung zwischen den Knoten hergestellt. Dazu wird die benötigte Bandbreite (eigentlich Kanal-Datenrate) für diese Verbindung reserviert.

Die Leitungsvermittlung wird auch *Durchschaltevermittlung* oder engl. *circuit switching* oder *line switching* genannt.

Wird das gesamte Medium reserviert spricht man von einer physikalischen Verbindung. Wird nur «Bandbreite» reserviert, spricht man von einer *logischen* oder *virtuellen* Verbindung.

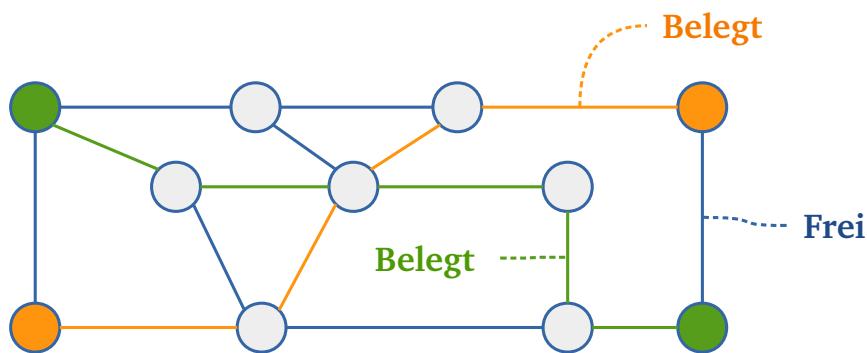


Abbildung 17.8: Beispiel Leitungsvermittlung (Verbindungsorientiert)

Vorteil

während der Verbindungsduer ist immer die gesamte «Bandbreite» mit *optimaler, gleich bleibender Qualität* vorhanden.

Nachteile

durch das Reservieren von Netzwerkressourcen werden diese nicht optimal ausgenutzt.

Beispiel

POTS (Plain old telephone service), ISDN

17.2.4 Zellenvermittlung (Verbindungsorientiert)

Bei der Zellenvermittlung werden Datenpakete mit einer fixen Grösse übermittelt. Diese werden Zellen genannt. Die Zellenvermittlung ist vorwiegend für Netze mit kleinen Fehlerraten geeignet (Glasfaser). Dadurch kann der Aufwand in der Sicherungsschicht (OSI-Layer 2) enorm reduziert werden.

Die Zellenvermittlung erfolgt über logische Verbindungen (virtuelle Kanäle).

Die Zellenvermittlung erinnert stark an die Paketvermittlung (Aufteilung in Datenpaket à gleicher Grösse) wird in der Praxis aber nahezu ausschliesslich als Verbindungsorientierter Dienst implementiert und hat sich eine eigene Bezeichnung verdient.

17.2.5 Paketvermittlung (Verbindungslos)

Bei der Paketvermittlung wird jedes Paket unabhängig übermittelt. Die Paketvermittlung wird engl. auch *packet switching* genannt. Jedes Paket enthält die *gesamte* Zieladresse und wird somit von Knoten zu Knoten gesendet, wobei jeder Knoten für sich den Weg zum nächsten Knoten auswählt und dies für jedes Datenpaket. Somit wird jedes Paket *gleich behandelt* und ist *unabhängig* von allen anderen Paketen. Diese können in den einzelnen Knoten jeweils zwischengespeichert werden. Dies wird auch *store and forward* Prinzip genannt. Für die Weiterleitung der Pakete, *das Routing*, ist ein komplexes Verfahren nötig.

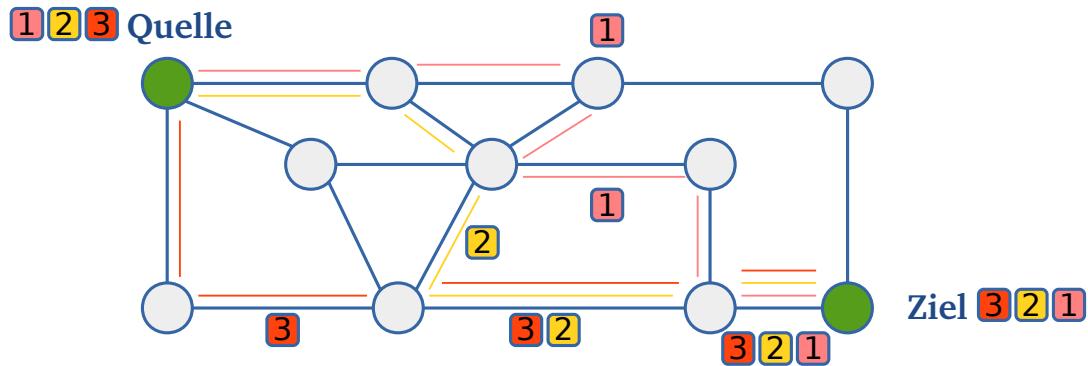


Abbildung 17.9: Beispiel Paketvermittlung

Vorteil

Effiziente Auslastung der Leitung, da multiplexed. Ressourcen werden fair verteilt. Durch die Verwendung mehrerer Routen kann Ausfallsicherheit garantiert werden.

Nachteil

Keine festgelegte Route, kein Schutz vor Überlast. Datenrate ist nicht garantiert, Schwankungen möglich.

Wichtig

Reihenfolge der Pakete ist nicht garantiert, da unterschiedliche Routen möglich.

Beispiel

Ethernet : UDP , X.25 (WAN), ATM (Asynchronous Transfer Mode)

17.2.6 Paketvermittlung (Verbindungsorientiert)

Wir können die Paketvermittlung zu einem *Verbindungsorientierten Dienst* aufwerten, indem wir jedem Paket eine Paketnummer vergeben. (Siehe Abbildung 17.9) Somit können wir die Pakete am Zielort wieder (in der richtigen Reihenfolge) zusammensetzen. Außerdem wird ein *Verbindungsaufbau*, *Verbindungszustand* und *Verbindungsabbau* definiert. Je nach Protokoll werden die Knoten zwischen den Start und Zielknoten beibehalten, so dass die Pakete immer die selben Knoten passieren. Dies muss aber nicht der Fall sein.

Die Nummerierung der Datenpakete führt ebenfalls dazu, dass wir sicherstellen können, dass jedes Paket am Zielknoten eingetroffen ist.

Vorteil

Wir haben mit einem Verbindungsorientierterem Dienst über ein Paketvermittelndes Netz eine Verbindung, welche die Ausleistung der Leitung maximiert, die Ressourcen fair verteilt und sicherstellt, dass ein Paket über ein dynamische Netzwerk sicher übermittelt werden kann.

Nachteil

Wir müssen jedem Paket eine Nummer zu ordnen und somit nochmals mehr Headerinformationen an ein Datenpaket anheften. Die Latenz ist im Vergleich zu einem Verbindungslosen Dienst höher, da wir die Pakete am Zielknoten erst wieder sortieren müssen, bevor wir die Daten an die höhere Schichten weiter leiten können. Unterschiedliche Routen sind möglich. Datenrate ist nicht garantiert, Schwankungen möglich.

Beispiel

Ethernet : TCP

Aufgabe 17.61

Frage

Beschreiben Sie die Eigenschaften eines Verbindungslosen Dienstes über ein Paketvermittelndes Netz: Als Hilfestellung sind die wichtigsten Fragen für sie vorbereitet (Bei einer Prüfung wird dies nicht der Fall sein)

Antwort

- Muss eine Verbindung aufgebaut werden?
 - Nein
- Ist den Datenpaketen etwas angeheftet? Nummer? Adresse?
 - Die Zieladresse
- Wie ist die Reihenfolge der eintreffenden Pakete?
 - Undefiniert, «out-of-order»
- Müssen die Knoten etwas tun oder bereitstellen?
 - Die Knoten müssen die Routing Information auf dem neusten Stand halten, optimieren des Weges etc.

Aufgabe 17.62

Frage

Beschreiben Sie die Eigenschaften eines Verbindungsorientierten Dienstes über ein Paketvermittelndes Netz: Als Hilfestellung sind die wichtigsten Fragen für sie vorbereitet (Bei einer Prüfung wird dies nicht der Fall sein)

Antwort

- Muss eine Verbindung aufgebaut werden?
 - Ja
- Ist den Datenpaketen etwas angeheftet? Nummer? Adresse?
 - Die Verbindungsnummer, die Zieladresse
- Wie ist die Reihenfolge der eintreffenden Pakete?
 - Definiert, «same-order»
- Müssen die Knoten etwas tun oder bereitstellen?
 - Sie müssen sich die Routing Information auf dem neusten Stand halten und die dazugehörige Sendungsnummer merken

Aufgabe 17.63

Frage

Beschreiben Sie die Eigenschaften eines Verbindungsorientierten Dienstes über ein Leitungsvermittelndes Netz: Als Hilfestellung sind die wichtigsten Fragen für sie vorbereitet (Bei einer Prüfung wird dies nicht der Fall sein)

Antwort

- Muss eine Verbindung aufgebaut werden?
 - Ja
- Ist den Datenpaketen etwas angeheftet? Nummer? Adresse?
 - Nicht zwingend. Falls möglich kann durch multiplexen die Leitung mehrfach genutzt werden.
- Wie ist die Reihenfolge der eintreffenden Pakete?
 - Definiert, «same-order»
- Müssen die Knoten etwas tun oder bereitstellen?
 - Sie müssen die Leitung reservieren

17.3 Verbindung von Netzen

Wir betrachten nun die (praktische) Verbindung von Netzen. Als Beispiel dient uns Ethernet. Die Verbindung von Ethernet Netzen wird auch *Internetworking* genannt.

17.3.1 Repeater (Hub)



Wir betrachten zunächst den Repeater. Multiport Repeater werden manchmal auch als Hub bezeichnet. (Allerdings werden auch Multiport-Bridges als Hub bezeichnet...)

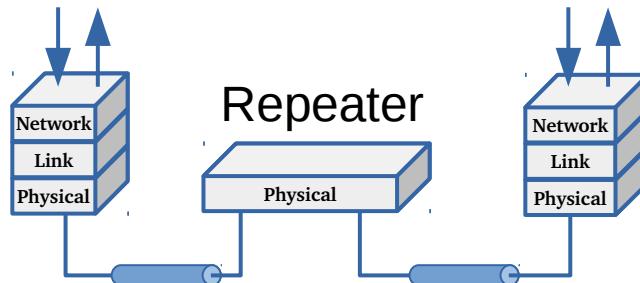


Abbildung 17.10: Repeater

Ein Repeater arbeitet auf dem *Physical Layer* (OSI-Layer 1). Er ist im Grunde eine Signalverstärker und erweitert so die Ausdehnung eines Netzes. Außerdem werden die Signale «restauriert». Erkennt der Repeater eine Kollision, werden auf allen Ports ein **Jam-Signal** ausgegeben. Wird ein Port zu lange besetzt, wird es unterdrückt. Dies wird als *jabber suppression* bezeichnet. (jabber engl. für «plappern») Repeater sind bei Ethernet am «aussterben». Sie werden immer mehr von den Bridges ersetzt.

17.3.2 Bridge (Switch)

Die Bridge arbeitet auf dem *Link Layer* (OSI-Layer 2) und trennt einzelne Ethernet Netze in separate *collision domains*. Innerhalb einer *collision domain* müssen alle angeschlossenen Knoten eine Kollision detektieren können. Sog. *Multiport-Bridge* werden auch als *Switching-Hub* oder *Switch* bezeichnet.

Eine Bridge analysiert die MAC-Adresse eines Ethernet-Frames. (Im Gegensatz zu einem Repeater, der analysiert nichts vom Datenpacket)

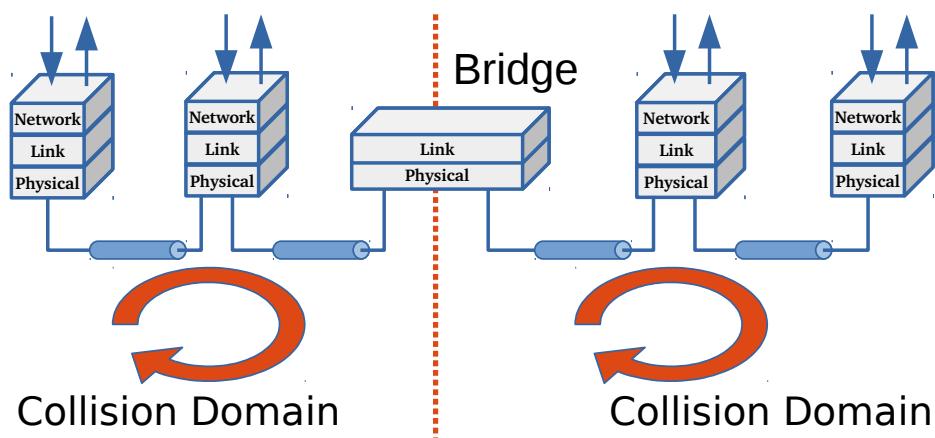
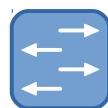


Abbildung 17.11: Bridge / Switch

Obwohl die Bridge die Netze in separate *collision domains* teilt, sind immer noch alle Knoten Teil der gleichen *broadcast domain*. Die Bridge untersucht die einzelnen Paket-Zieladressen und sendet diese nur an die beteiligten Knoten. Damit entsteht die eigentliche Separierung der *collision domain*. Aber eine falsch platzierte Bridge, kann die Lastverteilung auch negativ beeinflussen:

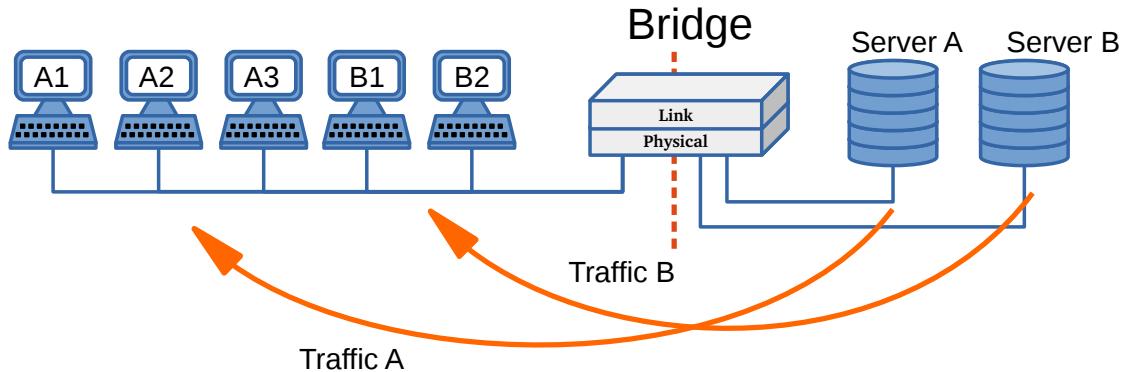


Abbildung 17.12: Bridge / Switch : bad example

In Abbildung 17.12 ist ein schlechtes Beispiel für den Einsatz eines Switches angegeben. Beide Server führen ihren gesamten Traffic über eine Bridge. Dies führt dazu, dass der Switch überlastet wird. (Besser wäre der Switch zwischen Server A mit seinen Clients und Server B mit seinen Clients. Siehe Figur Abbildung 17.13)

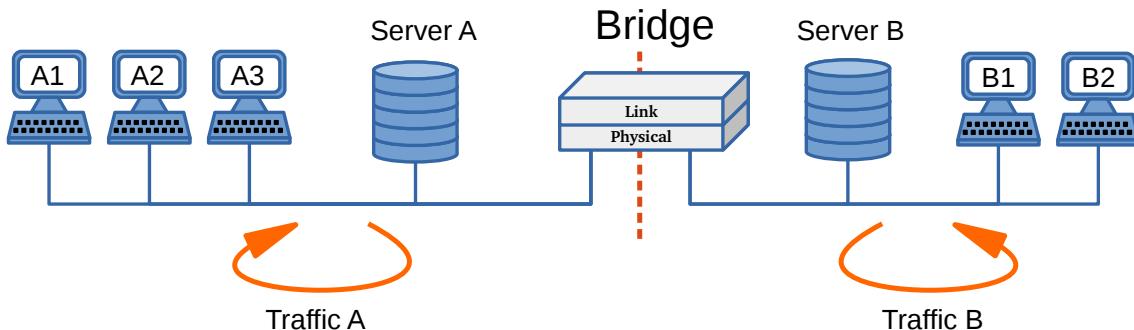


Abbildung 17.13: Bridge / Switch : good example

Die Switches müssen lernfähig sein. Daher sie lernen die Adressen der beteiligten Knoten. Beim Start-Up weiss ein Switch nichts über das Netz. Wird eine Nachricht übermittelt, wird diese zunächst an alle Angeschlossenen Knoten gesendet. Wenn das Paket angenommen wird, kennt der Switch die Adresse dieses Knoten und updatet seine interne Routingtabelle.

Moderne Switches kennen die *multicast*-Übertragung. (Diese muss meist von Hand konfiguriert werden)

Remote-Bridge

Werden zwei Netze über eine längere Distanz miteinander verbunden, werden paarweise sog. *remote-bridges* eingesetzt. Diese besitzen nicht nur einen LAN Anschluss, sondern meist auch einen speziellen Anschluss für die gegenüberliegende *remote-bridge*. Diese werden z.B. Mit Glasfaser versorgt.

17.3.3 Router



Router werden dafür eingesetzt, verschiedene Netze oder Subnetze, Domains oder Areas miteinander zu verbinden. Dazu gehört auch die Verbindung mit Netzen unterschiedlicher Protokollstacks. (Der Router muss dabei die Protokollstacks aller angeschlossenen Netze beherrschen). Außerdem limitiert / separiert ein Router die *broadcast domain*. Ein Router untersucht das Datenpacket und benutzt (bei Ethernet) die IP-Adresse um das Packet effizient zu routen.

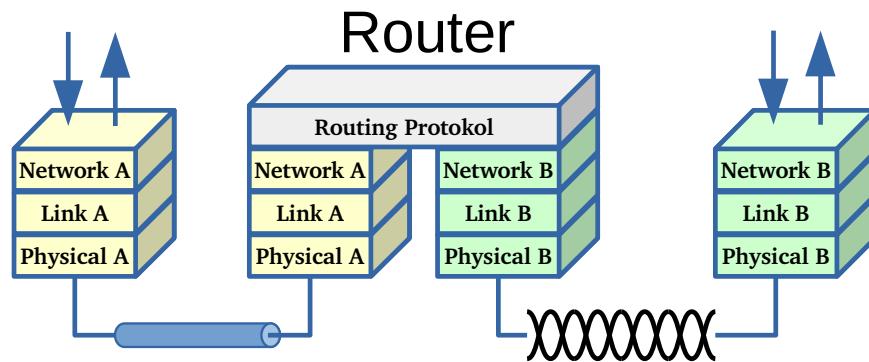


Abbildung 17.14: Router

Vorteile eines Router gegenüber Bridges

- Router benutzen immer den optimalen weg zwischen zwei Punkten. Vor allem in verzweigten Netzen wie zB. im WAN führen sie effizienter zum Ziel
- Router können grosse Netze in logische Netzbereiche unterteilen und verkehrsmässig gut voneinander trennen
- Das Trennen verschiedener Netzbereiche erhöht die Sicherheit
- Das Aufteilen grosser Netze verbessert die Verwaltung
- Router teilen eine Broadcast-Domain

Nachteile

- Router sind teurer als Bridges
- Router müssen konfiguriert werden
- Router sind abhängig vom Network-Layer, dies kann zu Problemen führen
- Es gibt Protokolle welche nicht über ein Network-Layer verfügen, diese sind in der Regel nicht mit einem Router benutzbar. Man nennt diese auch *nonroutable protocols*

Routing Protocol

Damit ein Router überhaupt routen kann, muss er die Netztopologie kennen. Diese wird über verschiedene Protokolle unter den Routern ausgetauscht.

Es gibt dabei normierte Routing-Protokolle für OSI-Netzprotokolle:

- Intermediate System- Intermediate System (IS-IS)
- Intermediate System – End System (IS-ES)

Und ebenfalls für IP-Netzprotokolle:

- Routing Information Protokoll (RIP)
- Exterior Gateway Protocol (EGP)
- Open Shortest Path First (OSPF)
- Border Gateway Protocol (BGP)

17.3.4 Gateway

In TCP/IP Netzen wird der Gateway oft als Synonym für den Router bezeichnet. Aber in der engeren Definition deckt der Gateway alle 7 OSI-Layer ab. Gateways werden in der Regel in Grossrechner Anlagen eingesetzt.

17.4 Internet Protocol (IP)

Das wohl bekannteste *Network-Layer* Protokoll ist das sog. *Internet Protocol*. Es bildet zusammen mit *TCP/UDP* die Basis für «Das Internet». Es ist nicht vollständig OSI-Layer konform.

Wir werden nun ein paar Eckpunkte von IP ansehen, damit wir den Grundaufbau kennen.

Derzeit sind zwei Versionen des IP-Standards aktuell : *IPv4* und das neuere *IPv6*

Ich hoffe Sie erinnern sich noch an *Kapitel 12.4.3 : Ethernet Frame*. Das nachfolgende Datagramm, ist dass was als Payload in einem Ethernet-Frame verschickt wird.

17.4.1 Datagram

Die Datenpakete im IP werden als *Internet Protocol Datagram* bezeichnet. Es besteht aus zwei Teilen. Dem *Header* und dem *Datenteil*. Im Header sind alle Protokoll relevanten Informationen und im Daten teil sind logischerweise die Daten.

IPv4 Datagram Header

Der IPv4 Header ist mind. 20 Byte lang. Kann aber mit zusätzlichen 40 Byte optionale Infos enthalten. Aber nicht länger als 60 Byte.

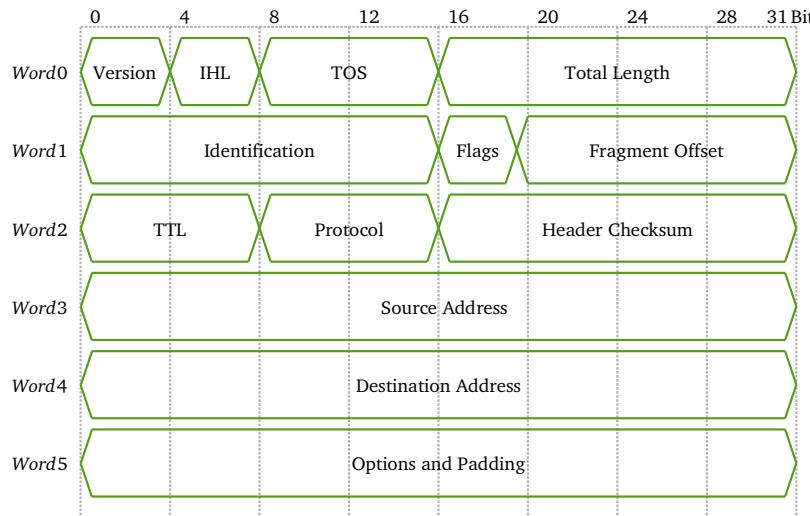


Abbildung 17.15: IPv4 Datagram Header

Aus *Abbildung 17.15 / Tabelle 17.1* ist der Aufbau des IPv4 Datagrams beschrieben. Neben den Standart Angaben ist interessant wie IPv4 mit der sog. Fragmentierung umgeht. Ein IPv4 Datagram kann im Netz geteilt werden und in mehrere kleinere Datagramme unterteilt werden. Hierzu dienen das *Flags*, *Fragment Offset* und *Identification* Feld. Wird ein Datagramm geteilt, bleibt das *Identification* Feld unberührt. Das erste Fragment erhält die Nummer 0. Das letzte Fragment enthält bei Flags das *MF=0* Bit. Der Offset im *Fragment Offset* ist der Datenoffset in Bytes des ursprünglichen Datenpaketes. Man sieht... recht kompliziert.

Interessant ist am IPv4 Header, das bei jedem Hop die Checksumme neu berechnet werden muss, da das *Time to Live* Feld bei jedem Hop verändert wird. Da man heutzutage aus Performance gründen die Checksumme einfach inkrementiert und nicht kontrolliert, hat man es in IPv6 gänzlich weg gelassen.

Header Field	Grösse	Beschreibung	Inhalt
Version	4 Bit	Protokoll Version	IPv4 / IPv6
IP Header Length (IHL)	4 Bit	Länge des IP Headers (in 32Bit Schritten)	[0-15]
Type of Service (TOS)	8 Bit	Priorisierung des IP Pakets	Bits 0-5: DSCP (Differentiated Services Code Point) Bits 6-7: ECN (Explicit Congestion Notification – IP-Flusskontrolle)
Total Length	16 Bit	Datagram Länge (in 8Bit Schritten)	[0-65535]
Identification	16 Bit	Datagram Kennung (Nummer)	[0-65535]
Flags	3 Bit	Indikator zur Fragmentierung	Bit 0 : Reserviert Bit 1: DF (Don't Fragment) Bit 2 : MF (More Fragments) → Last Fragment
Fragment Offset	13 Bit	Daten Fragment Offset (in 64Bit)	[0-8191]
Time to Live	8 Bit	Anzahl Hoppings, bis Paket zerstört wird	[0-255]
Protocol	8 Bit	Folgeprotokoll	Bsp: TCP =6 / UDP = 17 (RFC 3232)
Header Checksum	8 Bit	Checksumme (Addition Einerkomplement)	(Wird meist nicht geprüft... aber bei jedem Hop inkrementiert)
Source Adress	16 Bit	IP Adresse des Senders	(network byte order)
Destination Adress	32 Bit	IP Adresse des Empfängers	(network byte order)
Options	– Bit	RFC-791	zB. IPSec / Internet Timestamp

Tabelle 17.1: IPv4 Datagramm Header

IPv6 Datagram Header

Der IPv6 Header ist immer 40 Byte lang. Interessant dabei, das es viel weniger optionale Felder aufweist als der IPv4 Header. Die Adressen sind mit 128 Byte, vier mal grösser als bei IPv4. IPv6 Datagramme können auch aus 3 Teilen bestehen. Einem Header, einem optionalen extended Header und dem Datenteil. Der extended Header ähnelt dem IPv4 optional Header Eintrag.

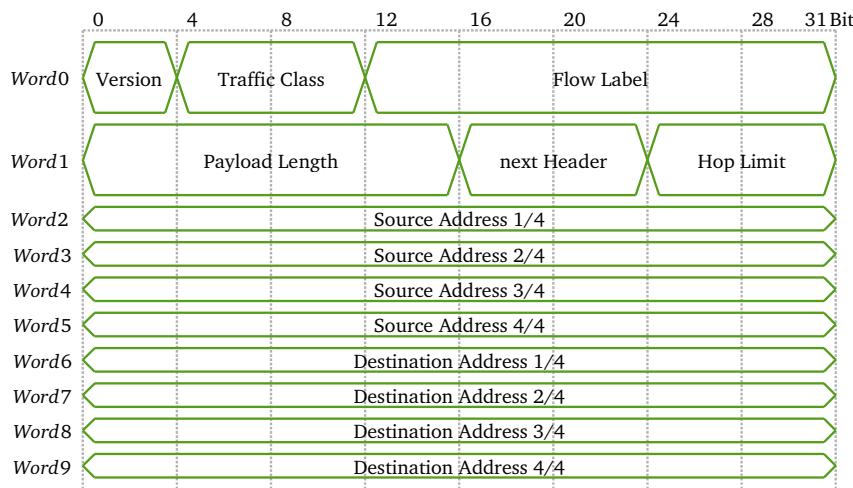


Abbildung 17.16: IPv6 Datagram Header

Header Field	Grösse	Beschreibung	Inhalt
Version	4 Bit	Protokoll Version	IPv4 / IPv6
Traffic Class	8 Bit	Quality of Service (Priorität)	Bits 0-5: DSCP (Differentiated Services Code Point) Bits 6-7: ECN (Explicit Congestion Notification – IP-Flusskontrolle)
Flow Label	20 Bit	QoS / Real-Time routing hint	Forciert gleichbleibendes Routing. Alle Pakete mit gleichem Flow Label gleichbehandeln
Payload Length	16 Bit	Länge der Nutzdaten (in Byte)	[0-65535]
Next Header	8 Bit	Protokoll Info (wie IPv4)	[0-255]
Hop Limit	8 Bit	Time to Live (wie IPv4)	[0-255]
Source Adress	128 Bit	IPv6 Adresse des Senders	
Destination Adress	128 Bit	IPv6 Adresse des Empfängers	

Tabelle 17.2: IPv6 Datagram Header

Die Fragmentierung wird in IPv6 ein wenig anders gehandhabt. Die Fragmente haben einen eigenen Header mit dem Eintrag `next Header=44 → fragment extension header`. Nach dem normalen IPv6 Header folgt ein *fragment extension header* und danach die *fragmentierten Daten*. Im *Fragment Header* gibt es ein *next Header*, *Fragment Offset*, *M Feld* und *Identification Number* Feld. Ist das *M Feld* =0 ist es das letzte Fragment. Alle Fragmente besitzen die gleiche *Identification Number*.

17.4.2 IP Adresse

Wir schauen uns noch die IP Adresse als solches an (Aber zur Zeit nur IPv4)

IPv4 IP Adresse

Die IPv4 Adresse besteht aus 32 Bit. Man schreibt die Zahl in 8 Bit Blöcke mit einem Punkt als Separator.

Wichtig

Die IP Adresse besteht aus einem **Netz-Teil** und einem **Host-Teil**

Die Trennung erfolgt durch eine zweite 32 Bit Nummer, welche als Subnetzmaske bezeichnet wird.

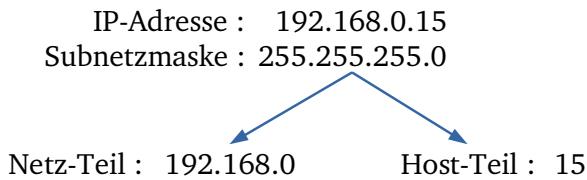


Abbildung 17.17: Beispiel IPv4 Adresse

Die Subnetzmaske verwenden wir als «binäre Maske» und können so den Netzteil und den Host-Teil trennen. Wird die IP Adresse mit der Subnetzmaske AND verknüpft, erhalten wir den Netzteil. Der Rest ist der Host-Teil.

Was ist nun der Vorteil von der Trennung von Netz und Host-Teil?

In einem Subnetz sind alle Geräte, welche die gleiche Netz-Adresse haben im gleichen Netz. Sie können nur miteinander kommunizieren und besitzen die selbe Broadcast Domain. Im gleichen Teilnetz darf die Hostadresse nur **1** mal vorkommen. Damit Sie mit Geräten aus einem anderem Subnetz kommunizieren können, müssen Sie über einen Router, welche die Netze «zusammen routet».

Besondere Netzwerkadressen

Einige Netzwerkadressen sind für spezielle Zwecke reserviert. (RFC 6890) Hier nur ein kleiner Auszug.

Adressblock /Subnetzmaske	Verwendung	RFC
0.0.0.0 / 255.0.0.0	Das vorliegende Netzwerk	RFC 1122
10.0.0.0 / 255.0.0.0	Private Address Space	RFC 1918
172.16.0.0 / 255.240.0.0		
192.168.0.0 / 255.255.0.0		
224.0.0.0 / 240.0.0.0	Multicast Adress Space	RFC 5771
127.0.0.0 / 255.0.0.0	Loopback (lokaler Computer)	RFC 1122
192.88.99.0 / 255.255.255.0	IPv6 zu IPv4 Relay	RFC 3068

Tabelle 17.3: IPv4 Besondere Netzwerkadressen

Die Adresse 0 und die höchste Adresse im Subnetz sind immer reserviert. Die Adress 0 bezeichnet das eigene Netzwerk und die höchste Adresse ist die Broadcast-Adresse des Subnetzes.

Je nach Wahl der Subnetzmaske stehen einem eine gewisse Anzahl IP Adressen zur freien Vergabe:

$$k = \text{max}(\text{Hostaddress}) - \text{Subnetzmaske} - 1 \quad (17.1)$$

Sollte man kennen

Die IP Adressen 192.168.0.0 / 127.0.0.0 / 0.0.0.0 deren Bedeutung und die dazugehörigen Subnetzmasken sollte man kennen
Ausserdem das die 0 Adresse das eigene Netz darstellt und die höchste Adresse die Broadcast-Adresse

Aufgabe 17.64

Frage

Trennen Sie den Netz- und Host-Teil aus folgenden IP-Adressen und Subnetzmasken:

Antwort

IP-Adresse Subnetzmaske	192.168.0.1 255.255.0.0	Netz-Teil : Host-Teil :	192.168 0.1
IP-Adresse Subnetzmaske	178.15.9.100 255.0.0.0	Netz-Teil : Host-Teil :	178 15.9.100
IP-Adresse Subnetzmaske	1.1.1.1 255.255.255.252	Netz-Teil : Host-Teil :	1.1.1.xx 1

Aufgabe 17.65

Frage

Sie haben ein Server auf Ihrem PC gestartet und möchten diesen testen. Sie schreiben daher die IP Adresse direkt in die «Location Bar» (→ URL) ihres Browsers. Welche Adresse müssen Sie eingeben? Welche könnten Sie ebenfalls verwenden? Ihre IP Adresse ist 192.168.0.110:

Antwort

direct : 192.168.0.110
Loopback : 127.0.0.0

Aufgabe 17.66

Frage

Sie möchten eine Broadcast Nachricht in ihrem Subnetz absetzen. An welche IP-Adresse müssen sie diese senden? Ihre IP-Adresse ist: 192.168.0.110, Subnetzmaske 255.255.0.0:

Antwort

direct : 192.168.255.255
own network : 255.255.255.255

Aufgabe 17.67

Frage

Ebenfalls für die folgende IP-Adresse: 192.168.0.110, Subnetzmaske 255.255.240.0:

Antwort

direct : 192.168.15.255
own network : 255.255.255.255

17.4.3 Wieso MAC-- und IP--Adressen?

Wieso eigentlich MAC- und IP-Adressen? Hätte man dies nicht einfach mit nur einer Adresse realisieren können?

Stellen wir uns ein kleines Netzwerk vor:

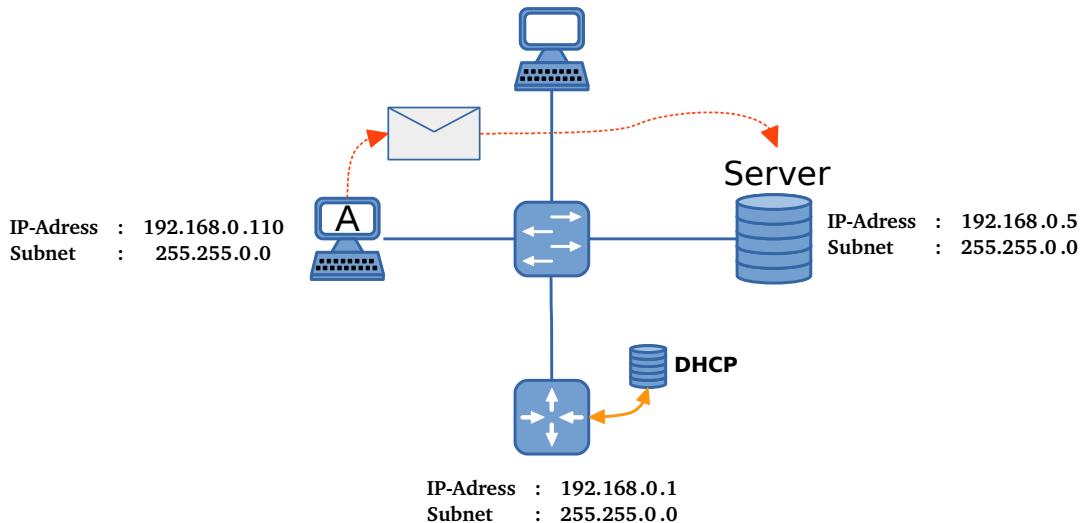


Abbildung 17.18: kleines Netzwerk

In Abbildung 17.18 ist ein kleines Netzwerk abgebildet. Der PC «A» möchte eine Nachricht an den Server senden. Damit die Nachricht nun zum Server gesendet werden kann, muss die Nachricht diesen einen Switch passieren. Der PC «A» baut sich nun sein Nachrichtenpacket zusammen:

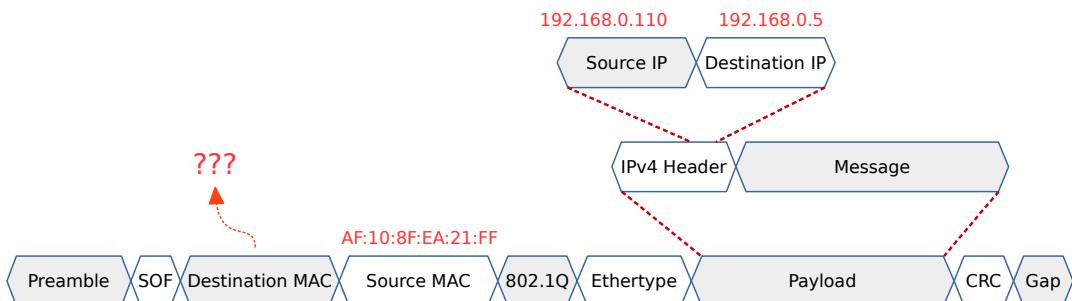


Abbildung 17.19: IP-Packet

Als Source IP, tragen wir die IP des PC «A» ein. Als Destination IP die IP des Servers ein. Aber für den OSI-Layer 2 Ethernet Dataframe müssen wir noch die MAC Adressen eintragen. Aber woher kennen wir die MAC Adresse des Servers?

Der PC «A», muss irgendwoher wissen, wie die Zuordnung der IP-Adressen und der MAC Adressen funktioniert. Die Antwort gibt uns der *DHCP Server*, welcher in diesem Beispiel auf dem Router läuft. Mit dem sog. *Address Resolution Protocol (ARP)* erfährt PC «A» von der MAC Adresse des Servers und kann nun die Nachricht fertig zusammenstellen.

Woher kennt PC «A» die IP / MAC Adresse des Routers? Diese wurde dem PC «A» bei der Vergabe der eigenen IP-Adresse mitgeteilt. (Hierzu später mehr).

Wir sehen an diesem kleinen Beispiel, dass wir zwar jetzt die Grundlagen zur IP-Kommunikation kennen, aber damit in der Praxis ein Netzwerk funktioniert, müssen wir die richtigen MAC/IP Adressen kennen. Dies ist nicht immer trivial.

Wieso es Sinn macht, neben der MAC Adresse noch die IP-Adressen zu verwalten ist noch nicht klar, daher geht es nun mit einem weiteren Beispiel weiter.

Wir haben nun ein klein wenig komplizierteres Netzwerk:

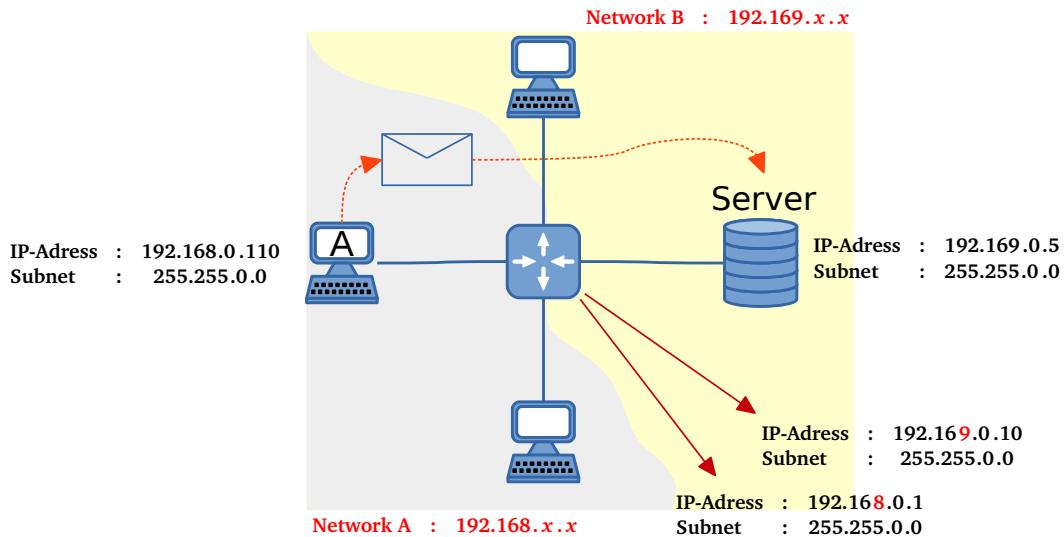


Abbildung 17.20: komplizierteres Netzwerk

In Abbildung 17.20 ist nun ein komplizierteres Netzwerk. Wir haben zwischen Server und PC «A» einen Router. Der Server befindet sich außerdem in einem anderen Subnetz. Der Router muss die Nachricht von PC «A» weiter routen.

Ist der Empfänger in unserem Subnetz?

Der Sender muss zunächst überprüfen ob der Nachrichten-Empfänger im gleichen Subnetz ist. Dies erkennt er über seine eigene Subnetzmaske und der Ziel-IP.

Die Antwort ist Nein. Er ist in einem anderen Subnetz. Der Sender muss nun seine Nachricht an seinen Gateway, in diesem Fall den Router senden. Als MAC Zieladresse wird der Router eingetragen.

Sehen wir uns das IP-Packet an:

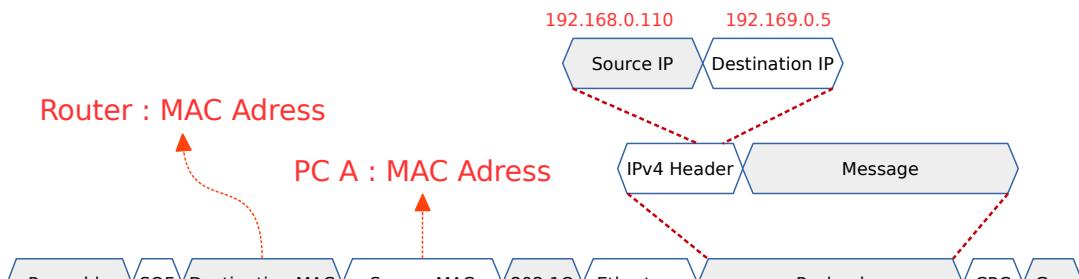


Abbildung 17.21: IP-Packet 1

Weiterleiten → Anpassen der Source MAC

Der Router sendet die Nachricht nun an den Server weiter. Dazu muss er ein neues Layer 2 Datagramm aufbauen. (Er muss ebenfalls den IPv4 Header anpassen und alle Checksummen neu berechnen)

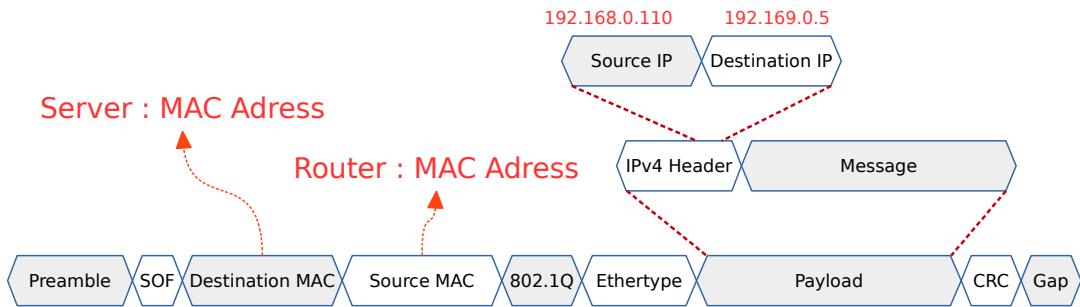


Abbildung 17.22: IP-Packet 2

MAC und IP

Wir sehen hier die Notwendigkeit dieser zweifachen Hierarchie. Wenn wir in einem Netzwerk einen Knoten brauchen um unsere Nachricht weiter zu leiten, müssen wir das Paket an den Knoten senden und an einem zweiten Ort die Endadresse hinterlegen. Das direkte Versenden von Knoten zu Knoten ist eine OSI-Layer 2 Angelegenheit. Das Paket über einen Knoten zu routen ist die Aufgabe des Layer 3. Und damit man die beiden Adressen auch ja nicht verwechselt haben sie einen eigenen Namen. Was durchaus Sinn macht.

Dieses Beispiel ist ein wenig abrupt zu Ende. Dies daher, da beim TCP/IP Protokoll Layer 3 und 4 nicht vollständig voneinander isoliert sind. Wir müssen erst den Layer 4 von TCP/IP verstehen damit wir ein interessanteres Beispiel durchspielen können.

18

OSI Layer 4 : Transport Layer

Inhalt des Kapitel

18.1	Funktionen des Layers 4 (Transport Layer)	246
18.2	Transport Layer Services	247
18.2.1	Verbindungsorientierter Dienst	247
18.2.2	Verbindungs-Aufbau / -Abbau	248
18.2.3	Daten sortieren	249
18.2.4	Flusskontrolle	249
18.3	Layer 4 Protokolle bei Ethernet	249
18.3.1	User datagram protocol (UDP)	249
18.3.2	Transmission control protocol (TCP)	250
18.3.3	Vergleich UDP /TCP Header Grösse	251
18.3.4	UDP / TCP Ports	251
18.3.5	Anwendungszenarien Ethernet	252
18.3.6	Network address translation NAT	253
18.3.7	Static NAT (port forwarding)	256

Zum Inhalt

In diesem Kapitel betrachten wir die Aufgaben des OSI Layer 4. Dieser hat die Aufgabe eine Punkt-zu-Punkt Verbindung aufzubauen und gegebenenfalls abzusichern.

18.1 Funktionen des Layers 4 (Transport Layer)

OSI Layer 4 wird auch *Transport Layer* oder *Transportschicht* genannt und beschreibt wie eine *Punkt-zu-Punkt* Verbindungen über ein Netz **aufgebaut** und **abgesichert** werden. Ebenfalls Aufgabe des Transport Layer ist die *Flusskontrolle auf Verbindungsebene*. Nicht zu vergessen ist die sog. *same-order-delivery*, der Network Layer garantiert uns nicht die richtige Reihenfolge der Datenpakete. Dies wieder zu ordnen ist die Aufgabe des Transport Layers.

Ein interessanter Aspekt des Transport Layers ist, dass dieser aus einem *Verbindungslosen Dienst eines Paketvermittelndes Netzes*, einen *Verbindungsorientierten Dienst* bereitstellen kann.

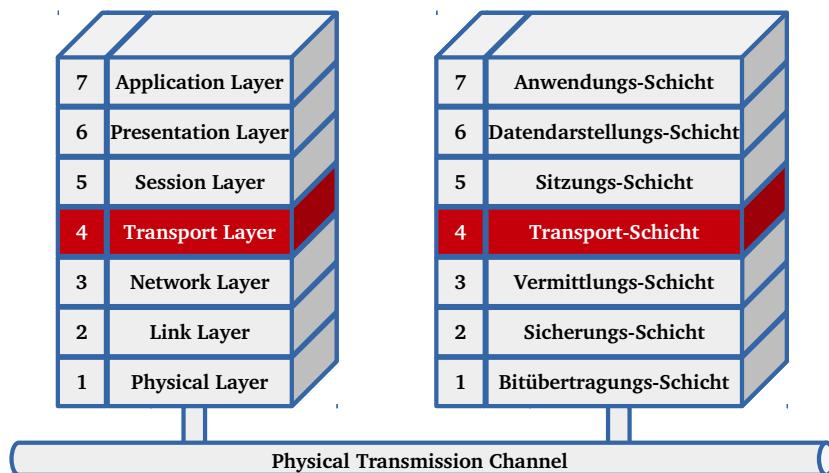


Abbildung 18.1: OSI Layer 4 (Transport Layer)

Wie sie in der Beschreibung sehen, kommen wieder die gleichen Konzepte (*Flusskontrolle*, *Punkt-zu-Punkt Verbindung*, etc) zum Einsatz.

Wieso diese Redundanz?

Tatsächlich gehen wir hier nicht genau das selbe Problem an, wie in den vorangegangenen Layern. Wir haben bis zum *Network Layer* wie ein Datenpaket von einem *Knoten* zu einem anderen *Knoten* gelangt. Aber was wenn wir ein Datenpaket über viele *Knoten* versenden möchten, es aber nicht zum Ziel gelangt? (zB. Wegen eines Leitungsunterbruchs). Wir müssen nun sicherstellen, dass das Datenpaket über eine andere Route durch das Netz geschickt wird. Hierzu müssen wir aber oberhalb des *Network Layer* nochmals eine *Flusskontrolle* aufbauen.

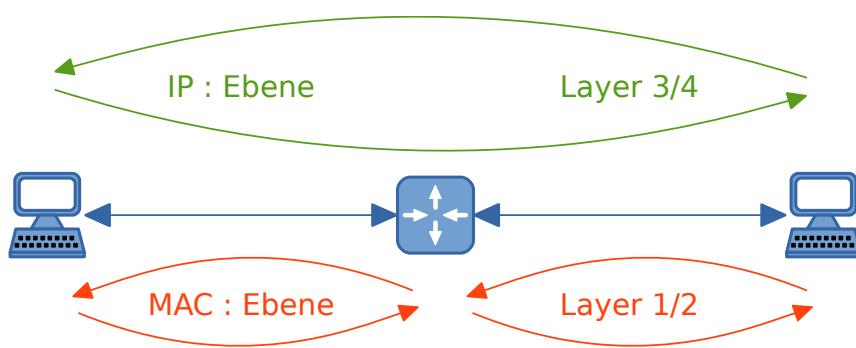


Abbildung 18.2: Vergleich MAC / IP Ebene

In Abbildung 18.2 ist dies schematisch abgebildet. Zwischen zwei Knoten bewegen wir uns auf der MAC Ebene und können Daten nur von Knoten zu Knoten senden. Die Flusskontrolle in Layer 2 sichert uns also nur die Verbindung von zwei benachbarten Knoten. Wenn wir über einen Knoten hinweg routen,

müssen wir eine weitere Flusskontrolle realisieren, die uns die Verbindung über mehrere Knoten sichert. Diese wird bei TCP/IP im Layer 4 realisiert.

Ein weiteres Beispiel: Wir senden eine Nachricht über mehrere Knoten. Woher wissen wir, dass die Nachricht nicht von einem Knoten verändert wird?

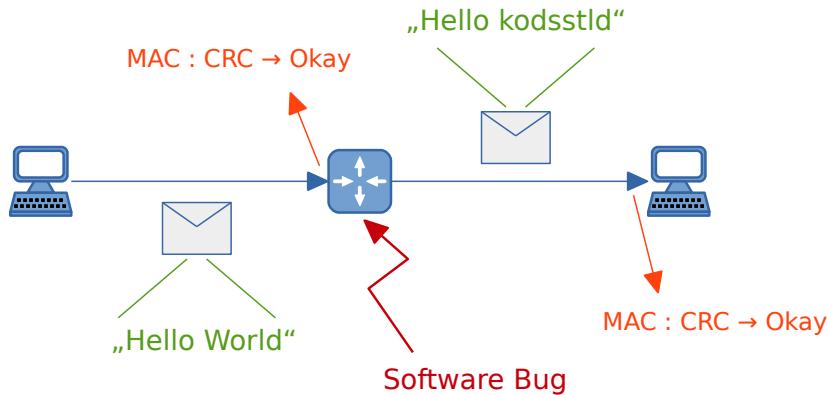


Abbildung 18.3: Vergleich MAC / IP Ebene (CRC)

In Abbildung 18.3 ist solch ein Fall abgebildet. Angenommen wir verschicken eine Nachricht, aber ein Knoten auf dem Weg hat ein Software Bug und verändert die Daten, aber berechnet den Layer 2 CRC richtig. Wie erkennen wir das? Der Sender muss eine Checksequenz für unsere Layer 3/4 Daten hinzufügen, damit der Empfänger die korrupten Daten erkennen kann.

Wir werden nun kurz Besprechen, welche Services der Layer 4 bereitstellt und danach betrachten wir die Implementation bei Ethernet.

18.2 Transport Layer Services

Da wir die Grundlagen bereits aus den vorhergehenden OSI-Layern kennen, gehen wir nicht mehr so tief ins Detail.

18.2.1 Verbindungsorientierter Dienst

Ethernet und damit «Das Internet» basiert auf einem Verbindungslosen Dienst über ein Paketvermittlendes Netz. Damit sind die gesendeten Datenpakete «out-of-order», Sie können verloren gehen. Im Grunde haben wir auch nur eine Simplex Verbindung, da die Datenpakete mit der Zieladresse versehen werden und nach dem «fire-and-forget» Prinzip verschickt werden. Ohne zu wissen ob diese ankommen.

Bei einem Verbindungsorientierten Dienst wissen wir immer ob die Verbindung steht oder ob diese abgebrochen ist. (zB. Leerzeichen beim Telefon) Die Datenpakete sind «same-order».

Damit der Transport Layer aus einem Verbindungslosen Dienst ein Verbindungsorientierten Dienst anbieten kann muss er folgende Dienste kombiniert anwenden:

- Verbindungsauflösen/-abbau, auch Zustandsabfrage muss realisiert sein. (Wiederaufbau nach Abbruch)
- Daten **same-order** sortieren
- Flusskontrolle inkl. **feedback error control** (zB. ACK / NACK, ARQ etc)
- Manchmal ist eine zusätzliche Sicherungsschicht implementiert, so dass die Daten nochmals mit einer **error detection / correction** geschützt sind.

18.2.2 Verbindungs--Aufbau / -Abbau

Wie muss man sich nun so ein Verbindungsauftbau vorstellen? Nehmen wir hierzu die OSI Layer 4 Implementation von dem ISO/IEC 8073/ITU-T Recommendation X.224 :

Im X224 sind folgende Datenpaket Typen definiert:

Type	Beschreibung
Connection Request	Signal für den Verbindungsauftbau
Connection Confirm	Bestätigung für den Verbindungsauftbau
Disconnection Request	Signal für den Verbindungsabbau
Disconnection Confirm	Bestätigung für den Verbindungsabbau
Data	Daten
Expedited Data	«beschleunigte» Daten (nur Class 1/4)
Data Acknowledge	ACK für Datenpaket
Expedited Acknowledge	ACK für «beschleunigtes» Datenpaket (nur Class 1/4)
Reject	Ablehnungs-Signal (nur Class 1)
Error	Error Signal

Tabelle 18.1: X224 : Datenpaket Type

Wie wird eine Verbindung aufgebaut : Man sendet das *Connection Request Signal* und wartet auf ein *Connection Confirm Signal*. Fertig. Der Abbau erfolgt analog mit den *Disconnection Signalen*.

Man mag jetzt einschreiten und sagen: «Das ist doch viel zu einfach». Die eigentliche «Magie» liegt nicht im Verbindungsauft und -abbau sondern im Kontext, was wir mit dem Verbindungsauftbau erreichen wollen.

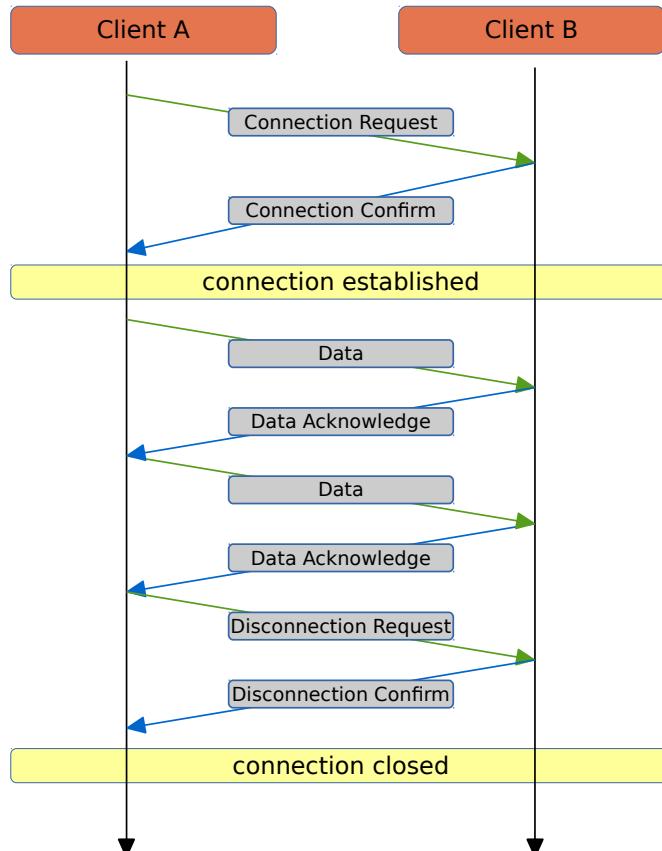


Abbildung 18.4: X224 : Verbindungsauft und abbau

Mit diesen einfachem «signalling» können wir sicherstellen, dass die Gegenstelle exisitiert und bereit ist die Daten zu übertragen. Und wir können nach der Transaktion die Ressourcen der Gegenstelle wieder frei geben, so dass diese nicht überlastet wird. Das ist auch schon alles.

18.2.3 Daten sortieren

Die Daten werden vom «Network Layer» **out-of-order** gesendet. Im Transport Layer werden die Daten daher in sog. *Segmente unterteilt* und *durchnummieriert*. Im Transport Layer des Empfängers wieder die Segmente wieder sortiert und zusammengesetzt. Falls nur ein Segment fehlt und es auch mithilfe der Flusskontrolle nicht innerhalb der vorgegeben Zeit eintrifft, wird das ganze Datenpaket verworfen.

18.2.4 Flusskontrolle

Die Flusskontrolle auf dem Transport Layer überwacht die Übertragung der Segmente und erwartet eine Quittierung für den Erhalt der einzelnen Segmente. Fehlt die Quittierung eines Segments, wird dieses erneut gesendet bis die **max. Anzahl versuche** oder ein **Timeout** auftritt. (Siehe: Kapitel 16 : Flusssteuerung, nur mit Segmenten anstatt Paketen) Je nach Protokoll kann auch der Empfänger tätig werden und ein einzelnes Segment nochmals anfordern.

18.3 Layer 4 Protokolle bei Ethernet

Bei Ethernet haben wir zwei Layer 4 Protokolle. Zum einen das **User datagram protocol (UDP)** und das **Transmission control protocol (TCP)**. Das **UDP** Protokoll ist ein *Verbindungsloses Protokoll*, während **TCP** eine *Verbindungsorientiertes Protokoll* implementiert.

18.3.1 User datagram protocol (UDP)

UDP implementiert eine Verbindungslose Übertragung und hat folgende Eigenschaften:

- Verbindungsloser Dienst
- Unzuverlässiger Transport (kann Packete verlieren, keine Flusssteuerung)
- **out-of-order-delivery**
- Data Checksum optional
- Port Adressierung
- Für Broadcast geeignet

Vorteil

Wenig **Latenz**. Für Real-Time Systeme kann es sinnvoller sein, UDP Pakete zu verlieren, als auf verspätet Pakete zu warten.

Nachteil

Wichtig : UDP

Pakete können verloren gehen. Pakete sind nicht geordnet.

Unterschied zum OSI Modell

Wie TCP verfügt auch UDP über Ports, welche nach OSI-Modell nicht in diese Schicht gehören.

Ein Witz

Ich kenn ja nen tollen UDP Witz, aber ich weiß nicht, ob der ankommt...

18.3.2 Transmission control protocol (TCP)

TCP/IP ist der Grundstein des modernen Internets. Es wird im Transport Layer eingesetzt und hat folgende Eigenschaften:

- Verbindungsorientierter Dienst
- Zuverlässiger Transport
- **same-order-delivery**
- Data Checksumme
- Flusskontrolle
- Port Adressierung
- Überlastkontrolle / Congestion Control (wird nicht weiter erklärt...)

Vorteil

Zuverlässiger Transport, **same-order-delivery**, Flusskontrolle

Nachteil

Viel **Overhead**, also schlechtes Protokolldaten zu Nutzdaten Verhältnis. Relativ hohe Latenz, da der Transport Layer nur vollständige Segmente als **empfangen** deklariert.

Unterschied zum OSI Modell

TCP/IP hält sich nicht an das OSI-Referenzmodell. Daher wird in TCP auch ein Multiplexing durchgeführt, welches eigentlich im Session Layer durchgeführt wird.

Für was braucht es dieses Multiplexing? Nehmen wir an, Sie hören mit ihrem PC zum Beispiel ein Internet Radio und gleichzeitig surfen sie auf einer Webpage. Während die Daten der Webpage geladen werden, müssen die Daten des Internet Radio's in regelmässigen Abständen empfangen werden können. Wie unterscheiden wir nun, welche Daten zu welchem Programm gehören?

TCP/IP hat hierfür sog. **Ports**. Dies ist ein Feld im TCP Datagram / Segment. Mit diesen **Ports** können die Datenströme den Applikationen zugeordnet werden. Bei TCP/IP werden Ports auch gebraucht um Standard Dienste nach aussen zugänglich zu machen. Es sind somit nicht alle Ports frei wählbar.

TCP

Zuverlässiger Transport, **same-order-delivery**, Flusskontrolle, aber viel «Overhead»

18.3.3 Vergleich UDP /TCP Header Grösse

Vergleichen wir noch kurz den UDP und TCP Header. Dies nur als Größen Vergleich, das man sieht wie viel «Overhead» wir bis an hin schon verursachen:

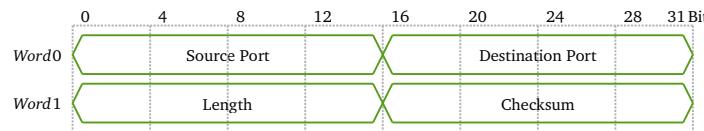


Abbildung 18.5: UDP Header

Der UDP Header besteht aus 2 Word (8 Byte)

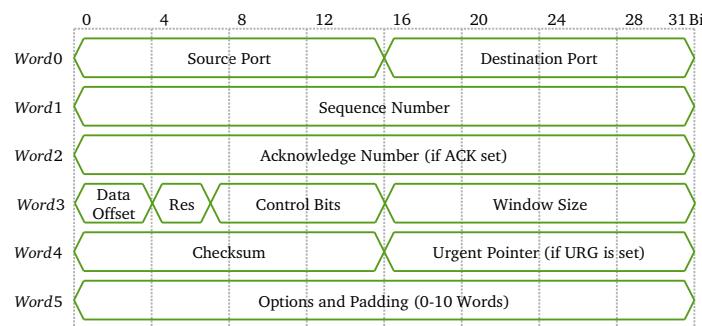


Abbildung 18.6: TCP Header

Der TCP-Header besteht aus min. 5 Word, bis max. 15 Word (20 – 60 Byte)

Wenn wir das ganze per IPv4 versenden kommen nochmals mind. 5 Word (20 Byte) für den IP-Header hinzu . Und wird dies über ein Ethernet-Frame übertragen kommen nochmals 42 Byte für den Link-Layer-Header hinzu.

Alleine für ein einfaches UDP Paket brauchen wir mind. 70 Byte Header. Für ein TCP Paket sind es 82 – 122 Byte.

18.3.4 UDP / TCP Ports

UDP und TCP verwenden die Ports für verschiedene Dinge. Wie bereits erwähnt, jedes Programm auf unserem PC / Smartphone etc. welches auf das Internet zugreift, erhält vom Betriebssystem eine Port Nummer zugewiesen. Damit können wir einzelne Datenpakete den jeweiligen Programmen zuweisen.

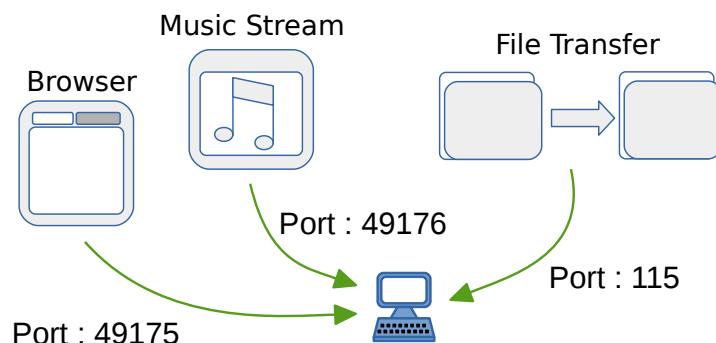


Abbildung 18.7: UDP / TCP Ports Example

In Abbildung 18.7 ist dies kurz Dargestellt. Jede Applikation, welche eine Ethernet Verbindung aufbaut, erhält eine Portnummer zugewiesen.

Wir haben zum einen sog. **ephemeral Ports** (ephemeral : engl. Kurzlebig, flüchtig). Diese werden vom Betriebssystem für kurze Sessions vergeben. Aber wir haben auch eine ganze Reihe, bereits festgelegte Dienste mit **fixen** Portnummern. Diese werden auch als *Well-known Ports* Bezeichnet:

Port	Dienst	Beschreibung	Port	Dienst	Beschreibung
7	TCP/UDP	Echo Protocol	67/68	TCP/UDP	DHCP
9	TCP	Discard Protocol	80	TCP/UDP	HTTP
9	UDP	Wake-on-LAN	220	TCP/UDP	IMAP
20	TCP/UDP	FTP data transfer	443	TCP/UDP	HTTPS
21	TCP/UDP	FTP control (command)	502	TCP/UDP	Modbus
22	TCP/UDP	Secure Shell : ssh/sftp	520	UDP	Routing Information Protocol (RIP)
23	TCP/UDP	Telnet	666	TCP/UDP	Doom (First Person Shooter)
25	TCP/UDP	SMTP	953	TCP/UDP	Domain Name System (DNS)

Tabelle 18.2: Well-known TCP / UDP Ports
https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers

Es gibt noch viele weitere Ports. Eine weitere Gruppe wird *Registered Ports* genannt und werden von der IANA verwaltet.

18.3.5 Anwendungszenarien Ethernet

Wir bearbeiten nun ein paar Beispiele, damit wir uns vorstellen können, wie Ethernet funktioniert.

Als erstes Beispiel sehen wir uns eine Ethernet Verbindung aus einem privaten Netzwerk zu einem Webserver im gleichen Netzwerk an. Wir benutzen für solch eine Anfrage das HTTP Protokoll.

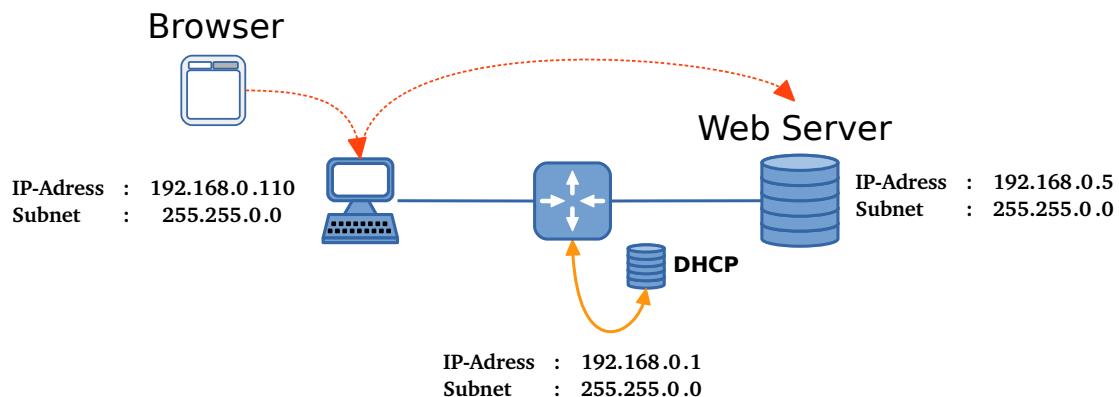


Abbildung 18.8: WebServer Anfrage im gleichen Netzwerk

In Abbildung 18.8 ist unser Beispielnetz abgebildet. Wir gehen davon aus, dass unser PC einen Browser startet und eine Webpage beim Web-Server anfragt. Der PC kennt die IP Adresse des Web Servers und für den Browser wurde ein *ephemeral Port* mit der Nummer 50132 zugewiesen. Damit unser Beispiel nicht zu schwierig wird, benutzen wir ein UDP Packet (HTTP benutzt in normalerweise TCP). Wie sieht nun das ganze UDP Packet aus?

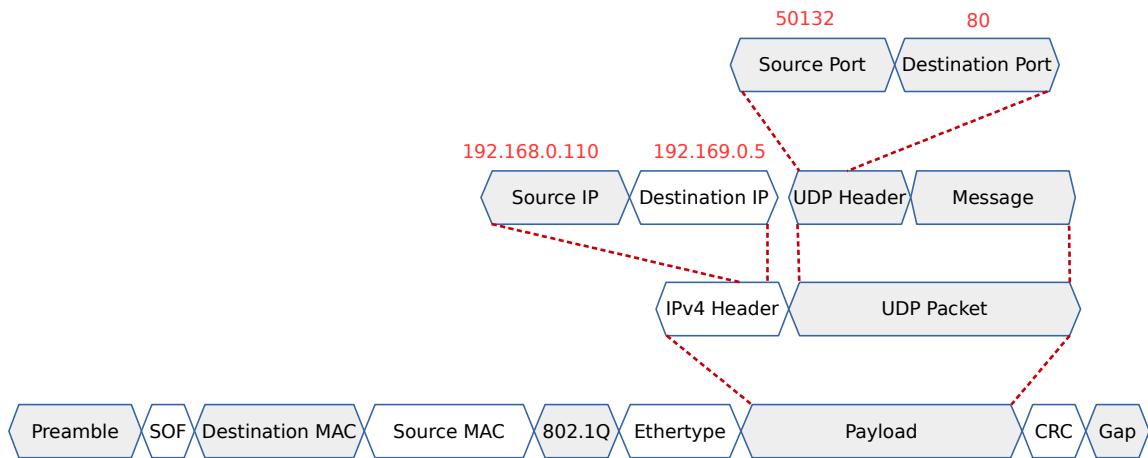


Abbildung 18.9: WebServer Anfrage im gleichen Netzwerk : UDP Paket

In Abbildung 18.9 ist nun das UDP Packet abgebildet. Wir sehen nun im Payload Teil des IP-Packets, unser UDP-Packet. Der UDP-Header besteht unter anderem aus den Source und Destination Ports. Für das Source-Port wird der Port des Browsers verwendet. Als Destination-Port wird das HTTP zugewiesene **Port 80** verwendet (Siehe : Tabelle 18.2)

Der Webserver wird nun ein UDP-Packet mit der Antwort zusammenstellen. Natürlich mit den Inversen IP's / Ports.

18.3.6 Network address translation NAT

Wir betrachten nun wie das ganze funktioniert, wenn der Webserver nicht im gleichen Netzwerk sitzt, sondern am anderen Ende vom Internet.

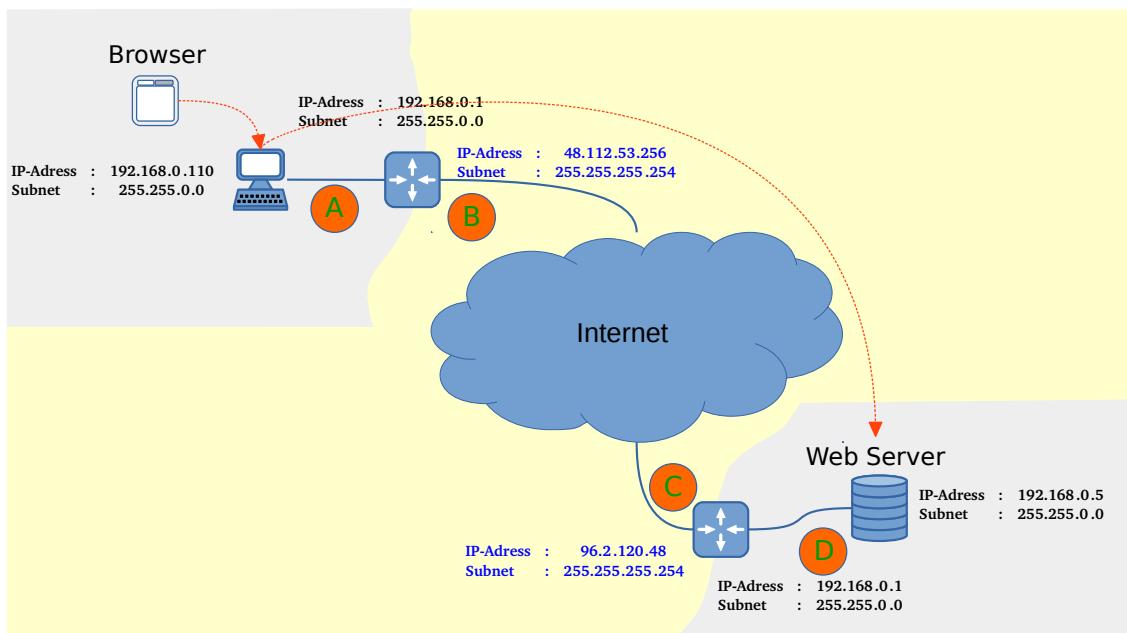


Abbildung 18.10: WebServer Anfrage im Netzwerk

In Abbildung 18.10 ist unser Beispiel abgebildet. Was als erstes auffallen sollte. Der Server sowie der PC

sind in einem privaten Netzwerk. Beide IP Adressen beginnen mit der IP Adresse 192.168.0.xxx.
Wir können die beiden nun miteinander Kommunizieren?
Die Antwort ist: Sie tun dies über Mittelsmänner. Die Router!

Punkt A

Sehen wir uns das Paket bei der Markierung A an:

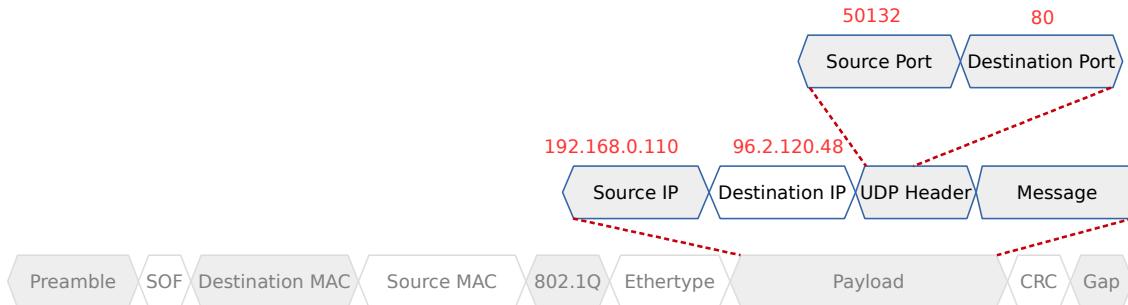


Abbildung 18.11: WebServer Anfrage im Netzwerk : A) UDP Paket

Der PC kennt nur die WAN: IP Adresse des Ziel Web-Servers. Daher ist die Destination IP = 96.2.120.48. Wir möchten eine HTTP Anfrage an den Server durchführen, daher Destination Port = 80.

Punkt B

Der Router nimmt das IP-Paket entgegen und erkennt, dass die Destination-IP ausserhalb des privaten Netzwerks liegt.

Was nun passiert, nennt sich **Network Address Translation (NAT)**. Der Router ersetzt die Source-IP im IP-Paket mit seiner eigenen WAN-IP. Ausserdem wählt er ein *ephemeral Port* und ersetzt das Source-Port im UDP Packet.

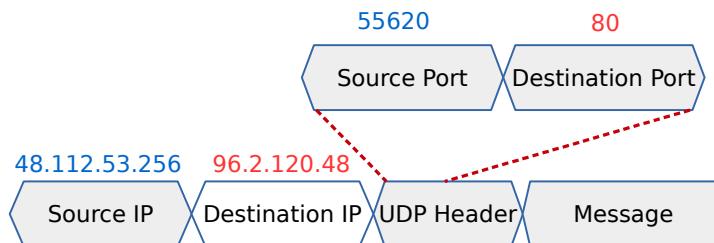


Abbildung 18.12: WebServer Anfrage im Netzwerk : B) UDP Paket

In Abbildung 18.12 ist das IP-Paket abgebildet (Der Layer 2 Teil wurde zur besseren Übersicht weg gelassen) Wir sehen, dass der Source-Port und die Source-IP verändert wurden.

Der Router führt intern eine Tabelle, so dass er immer die IP / Port Zuordnung kennt und nachführt.

Local Network	WAN Network
192.168.0.110:50132	48.112.53.256:55620

Tabelle 18.3: Beispiel Router NAT Tabelle

Punkt C

Am Destination Router wird das Paket empfangen und analysiert. Der Router weiss, dass er jede Web Server Anfrage an die IP 192.168.0.5 senden muss (dies nennt man Static NAT, dazu später mehr). Also verändert er das UDP-Paket erneut, so das bei der Destination-IP die Web Server-IP eingetragen wird.

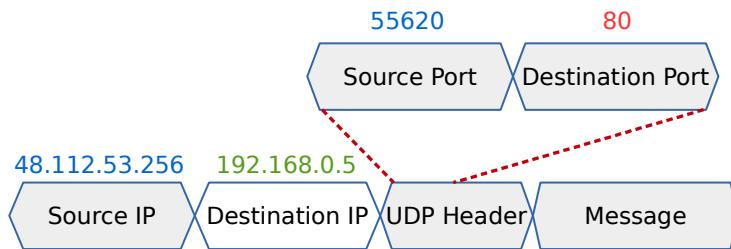


Abbildung 18.13: WebServer Anfrage im Netzwerk : C) UDP Paket

Damit ist nun nur das Destination Port unangetastet geblieben.

Punkt D

Die HTTP Anfrage wird beim Web-Server empfangen und bearbeitet. Die Antwort wird an die ehemalige Source-IP und Source-Port geschickt.

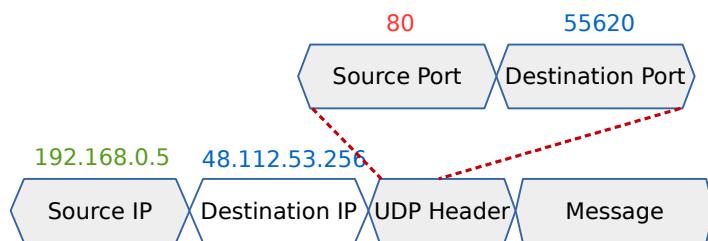


Abbildung 18.14: WebServer Anfrage im Netzwerk : D) UDP Paket

Damit beginnt es von neuem. Der Router des Web-Servers macht eine NAT Umsetzung und schickt das Paket zum PC zurück.

Wieso so kompliziert?

Zum einen ist uns bereits am Anfang aufgefallen, dass die beiden privaten Netzwerk ähnlich konfiguriert sind. Es muss also irgendwie unterschieden werden, welche Nachricht vom Netz her kommt und welche Nachrichten im privaten Netzwerk verschickt werden.

Zum zweiten haben wir eine Abschottung vom Internet zu unserem privaten Netzwerk. Damit ist es schwierig von aussen (vom Internet her) auf die interne Struktur eines privaten Netzwerks zu schliessen. Zum dritten können wir ein privates Netzwerk so konfigurieren wie wir wollen.

18.3.7 Static NAT (port forwarding)

Wir haben im vorgehenden Beispiel **Static NAT** gebraucht. Und zwar hat der Destination Router gewusst, an welche IP im privaten Netzwerk die HTTP Anfrage hingeschickt werden muss. Machen wir hierzu eine kleines Beispiel:

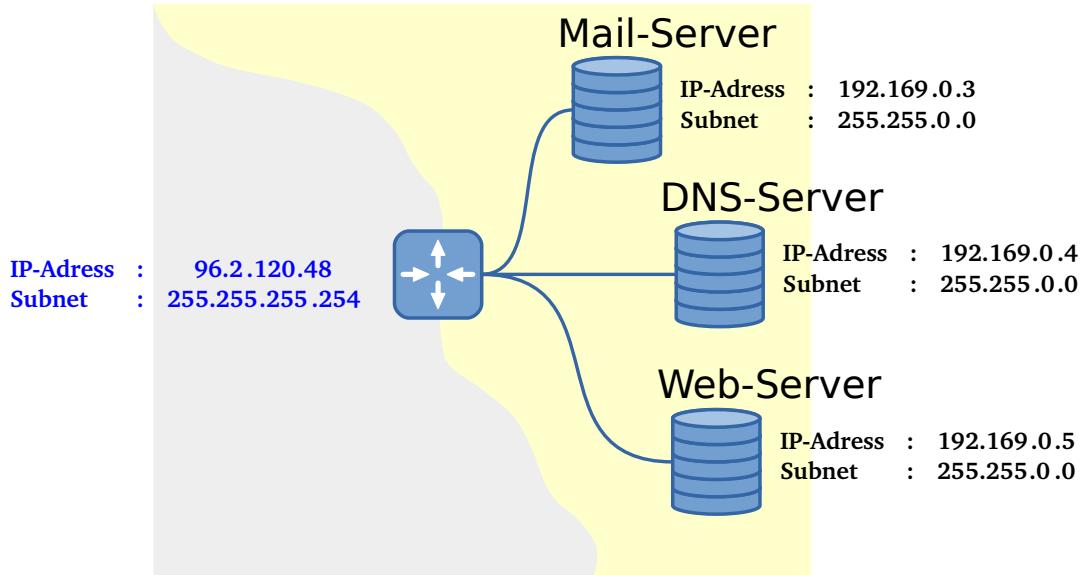


Abbildung 18.15: Static NAT Beispiel

In Abbildung 18.15 ist ein kleines Netzwerk abgebildet. Wir haben einen Mail- / DNS- und einen Web-Server. Wir möchten, dass diese Server aus dem Internet erreichbar sind. Hierzu müssen wir den Router so konfigurieren, dass er die Ports der einzelnen Server Dienste an die richtigen Server weiter leitet. Wie wir aus Tabelle 18.2 wissen, ist der IMAP Maidienst auf **Port 220**, der DNS Dienst auf **953** und HTTP auf **80**. Damit muss der Router folgendermassen Konfiguriert werden:

Protocol	Port	IP-Address
TCP / UDP	220	192.168.0.3
TCP / UDP	953	192.168.0.4
TCP / UDP	80	192.168.0.5

Tabelle 18.4: Static NAT Beispiel : Router Konfiguration

19

Gebäudeinstallation / Automation

Inhalt des Kapitel

19.1 Standardisierung	258
19.2 Zonen der Strukturellen Verkabelung	259
19.3 Verteilung nach ISO/IEC 11801	259

Zum Inhalt

In diesem Kapitel werden wir noch kurz besprechen, welche Standards für die Gebäudeinstallation / Automations-Anlagen gelten und wieso diese überhaupt benötigt werden.

19.1 Standardisierung

In der Vergangenheit war die Verkabelung von Gebäuden und Industrieanlagen nicht Standardisiert. Als Folge davon wurden die Verkabelung meist nur auf den momentanen Bedarf und Dienst ausgerichtet. Viele Firmen setzten auf nicht zukunftsträchtige Technologien, welche zwar für den momentanen Bedarf ausreichten, aber schon nach kurzer Zeit nicht mehr den Anforderungen genügten. **Eine typische Fehlinvestition.**

Um dies zu minimieren und zukunftsträchtige Technologien zu spezifizieren wurde die **ISO/IEC 11801** Norm realisiert. Diese wurde 1995 freigegeben und unter anderem 2002 und 2010 überarbeitet. Zur Zeit ist eine weitere Überarbeitung im Gange, welche diverse andere Standards ablösen wird.

ISO / IEC	Ersetzt	Titel	Beschreibung
ISO/IEC 11801-1	ISO/IEC 11801	Part 1: General requirements	Generische Anforderungen an Twisted Pair und optische Übertragungsleiter
ISO/IEC 11801-2	ISO/IEC 11801	Part 2: Office premises	Verkabelung von Geschäftsgebäuden
ISO/IEC 11801-3	ISO/IEC 24702	Part 3: Industrial premises	Verkabelung von Industriegebäuden mit Automations, Prozess Kontroll ud Monitoring Applikationen
ISO/IEC 11801-4	ISO/IEC 15018	Part 4: Homes	Verkabelung von Wohnhäusern inkl. 1.2GHz Links für CATV/SATV Applikationen
ISO/IEC 11801-5	ISO/IEC 24764	Part 5: Data centres	Verkabelung für High performance Netzwerke im Bereich der Rechenzentren
ISO/IEC 11801-6	-	Part 6: Distributed Building	Verkabelung für verteilte Wireless Netzwerke im Bereich Gebäudeautomation und IoT (Internet of Things)

Tabelle 19.1: ISO/IEC 11801:2015

Im Zusammenhang mit der **ISO/IEC 11801** spricht man auch von der sog. *Strukturierten Verkabelung*.

19.2 Zonen der Strukturellen Verkabelung

Die Strukturelle Verkabelung wurde auf 3 Zonen aufgeteilt.

Primärbereich Der Primärbereich umfasst die Gelände-Verkabelung und wird heutzutage mit 62.54/125 µm Glasfasern mit einer max. Länge von 2000 m.

Sekundärbereich Der Sekundärbereich umfasst die Gebäude-Verkabelung. Hier kommt ebenfalls 62.54/125 µm Glasfasern mit einer max. Länge von 500 m zum Einsatz.

Tertiärbereich Der Tertiärbereich umfasst die Etagen-Verteilung, welche vom Etagenverteilier bis zur Anschlussdose reicht. Zum Einsatz kommen Twisted Pair Kabel und auch Glasfaser mit einer max. Länge von 100 m.

19.3 Verteilung nach ISO/IEC 11801

Abkürzung	Beschreibung	Zone
SV	Standortverteilier	Primärkabel
GV	Gebäudeverteilier	Sekundärkabel
EV	Etagenverteilier	Tertiärbereich
KV	Kabelverteilier (optional)	Tertiärbereich
TA	Telekommunikations Anschlussdose	-

Tabelle 19.2: Abkürzungen ISO/IEC 11801

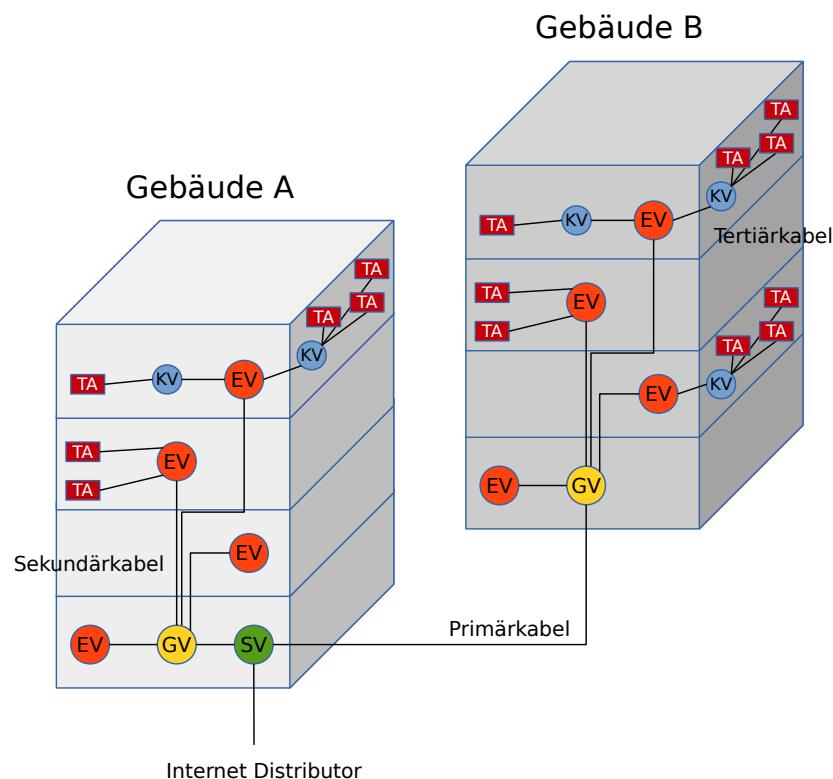


Abbildung 19.1: ISO/IEC 11801 Gebäudeverkabelung

20

Epilog

Leider ist der Kurs mit diesem Kapitel zu Ende.

Gerne hätte ich mit Ihnen noch weitere Themen erarbeitet, damit Sie noch besser auf Ihren zukünftigen Lebensabschnitt vorbereitet sind.

Wir haben zusammen die Grundlagen der Kommunikationstechnik erarbeitet:

- elektrotechnische Grundlagen
 - Eigenschaften eines Leiters
 - Reflexionen
 - Dämpfung
 - single-ended vs. differential signalling
 - Skin-Effekt
- Grundlagen der Informationstechnologie
 - ISO/OSI Referenzmodell
 - Serielle Schnittstelle UART / RS-232
 - Leitungscodes
 - Fehlerschutz und Fehlerkorrektur
 - Parity-Bit
 - CRC
 - Hamming-Code
 - BCH
- Standardisierung
- Grundlagen der Netzwerktechnologie
 - Netzwerktopologien
 - Aufbau eines Dataframe
 - Multiplexing
 - Flusssteuerung
 - Paket-Routing

Wichtig

Diverse Themen resp. Konzepte welche wir besprochen haben, lassen sich auch in anderen Fach-Gebieten verwenden!

Ich hoffe ich konnte Ihre Erwartungen an diesen Kurs erfüllen und wünsche Ihnen alles gute für Ihren weiteren Lebensweg.

Widmung

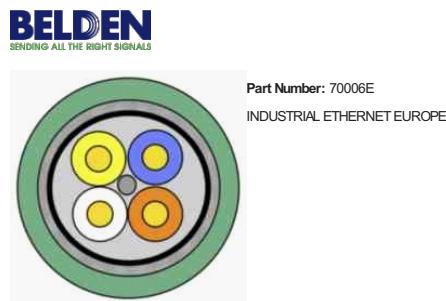
Ich widme dieses Skript meiner wunderbaren Partnerin Claudia:

Ich danke dir für deine Geduld, Unterstützung und Verständnis, welches dieses Skript erst möglich machte.

Anhang

A Datenblätter

A.1 Datasheet : Twisted Quad Ethernet Cable



Product Description

2x2AWG22 (QUAD)

Technical Specifications

Construction and Dimensions

Conductor:

AWG	Stranding	Material	Nominal Diameter	No. of Conductors
22	Solid	Bare copper	0.65 in	4

Insulation:

Material	Nominal Diameter	Diameter +/- Tolerance
Polyethylene	1.5 mm	0.05 mm

Color Chart 1:

Number	Color
Quad 1	White & Blue & Yellow & Orange

Innerjacket:

Material	Nominal Diameter
PVC Bedding	3.9 mm

Cabling 1:

Lay Length
1 quad with center filler

Part # 70006E | Page 1 of 4 | December 30, 2016

Abbildung A.1: Datasheet : 70006E (1)

http:

//belden.catalog.belden.com/en/Products/TechDataSheetPdf?product=PF_70006E

Outershield 1:

Type	Material	Min. Coverage [%]
Tape	Aluminum / Polyester	
Braid	Tinned copper	85 %

Outerjacket 1:

Material	Color	Nominal Diameter	Diameter +/- Tolerance
PVC	Green (RAL 6018)	6.5 mm	0.2 mm

Electrical Characteristics**Conductor DCR:**

Max. Conductor DCR
57.1 Ohm/km

Impedance:

Frequency [MHz]	Nominal Characteristic Impedance	Nominal Characteristic Tolerance
1 - 100	100 Ohm	15 Ohm

Delay:

Max. Delay	Nominal Velocity of Propagation (VP) [%]
530 ns/100m	68 %

High Freq:

Frequency [MHz]	Max. Insertion Loss (Attenuation)	Min. NEXT [dB]	Min. ACRF (ELFEXT) [dB]	Min. SRL (Structural Return Loss)
1 MHz	2.1 dB/100m	65.3 dB	64 dB	
4 MHz	4.1 dB/100m	56.3 dB	52 dB	23 dB
10 MHz	6.5 dB/100m	50.3 dB	44 dB	25 dB
16 MHz	8.3 dB/100m	47.2 dB	40 dB	25 dB
31.25 MHz	11.7 dB/100m	42.9 dB	34 dB	23.6 dB
62.5 MHz	17 dB/100m	38.4 dB	28 dB	21.5 dB
100 MHz	22 dB/100m	35.3 dB	24 dB	20.1 dB

Voltage:

Voltage Rating [V]
300 V

Transfer Impedance:

Frequency [MHz]	Transfer impedance
10 MHz	Max. 10 mOhm/m

Part # 70006E | Page 2 of 4 | December 30, 2016

Abbildung A.2: Datasheet : 70006E (2)
http://belden.catalog.belden.com/en/Products/TechDataSheetPdf?product=PF_70006E

A.2 Datasheet : Coaxial Ethernet Cable

Detailed Specifications & Technical Data

METRIC MEASUREMENT VERSION



9907 Coax - Coaxial Cable - Thinnet 10Base2 Ethernet

For more Information
please call

1-800-Belden1



General Description:

20 AWG stranded (19x32) .037" tinned copper conductor, foam polyethylene insulation, Duobond® II (100% coverage) plus an overall tinned copper braid shield (93% coverage), PVC jacket.

Usage (Overall)

Suitable Applications: Thin Ethernet

Physical Characteristics (Overall)

Conductor

AWG:

#	Coax	AWG	Stranding	Conductor Material	Dia. (mm)
1		20	19x32	TC - Tinned Copper	0.9398

Total Number of Conductors: 1

Insulation

Insulation Material:

Insulation Material	Dia. (mm)
FHDEPE - Foam High Density Polyethylene	2.5908

Outer Shield

Outer Shield Material:

Layer #	Outer Shield Trade Name	Type	Outer Shield Material	Coverage (%)
1	Bonded Duofoil®	Tape	Bonded Aluminum Foil-Polyester Tape-Aluminum Foil	100
2		Braid	TC - Tinned Copper	93

Outer Jacket

Outer Jacket Material:

Outer Jacket Material
PVC - Polyvinyl Chloride

Overall Cable

Overall Nominal Diameter: 4.699 mm

Mechanical Characteristics (Overall)

Operating Temperature Range:	-40°C To +80°C
UL Temperature Rating:	60°C (UL AWM Style 1354)
Bulk Cable Weight:	34.229 Kg/Km
Max. Recommended Pulling Tension:	200.169 N
Min. Bend Radius/Minor Axis:	50.800 mm

Applicable Specifications and Agency Compliance (Overall)

Applicable Standards & Environmental Programs

NEC(UL) Specification:	CL2, CM
CEC/C(UL) Specification:	CM
AWM Specification:	UL Style 1354 (30 V 60°C)
EU Directive 2011/65/EU (ROHS II):	Yes
IEEE Specification:	IEEE802.3 10Base2
Other Standards:	ISO8802.3 10Base2
EU CE Mark:	Yes
EU Directive 2000/53/EC (ELV):	Yes
EU Directive 2002/95/EC (RoHS):	Yes
EU RoHS Compliance Date (mm/dd/yyyy):	01/01/2004

Page 1 of 3

08-26-2016

Abbildung A.3: Datasheet : 9907 (1)
<http://www.belden.com/techdatas/metric/9907.pdf>

Detailed Specifications & Technical Data



METRIC MEASUREMENT VERSION

9907 Coax - Coaxial Cable - Thinnet 10Base2 Ethernet

EU Directive 2002/96/EC (WEEE):	Yes				
EU Directive 2003/11/EC (BFR):	Yes				
CA Prop 65 (CJ for Wire & Cable):	Yes				
Mil Order #39 (China RoHS):	Yes				
Customer Part Number Reference Specification:	DEC Part No. 17-01248-00				
RG Type:	58A/U				
Flame Test					
UL Flame Test:	UL1685 UL Loading				
Plenum/Non-Plenum					
Plenum (Y/N):	No				
Plenum Number:	82907, 89907				
Electrical Characteristics (Overall)					
Nom. Characteristic Impedance:					
Impedance (Ohm) Tolerance (Ohms)					
50 +/- 2					
Nom. Capacitance Conductor to Shield:					
Capacitance (pF/m)					
83.3374					
Nominal Velocity of Propagation:					
VP (%)					
80					
Nominal Delay:					
Delay (ns/m)					
4.16687					
Nom. Conductor DC Resistance:					
DCR @ 20°C (Ohm/km)					
28.8728					
Maximum Loop Resistance:					
Resistance (Ohm/km)					
50.0024					
Nominal Outer Shield DC Resistance:					
DCR @ 20°C (Ohm/km)					
19.0298					
Nom. Attenuation:					
Freq. (MHz) Attenuation (dB/100m)					
1 1.41083					
10 4.2653					
50 9.54771					
100 13.7802					
200 20.0141					
400 29.2009					
700 39.7001					
900 45.6059					
1000 48.5588					
Max. Operating Voltage - UL:					
Voltage Description					
300 V RMS	UL AWM Style 1354				
30 V RMS					
Notes (Overall)					
Notes: Tape to bond at overlap area only. Tape is not designed to bond to dielectric core.					
Put Ups and Colors:					
Item #	Putup	Ship Weight	Color	Notes	Item Desc
9907 E4XU1000	1,000 FT	24.000 LB	GRAY, LIGHT DEC		RG-58 TYPE COAX
9907 E4X1000	1,000 FT	25.000 LB	GRAY, LIGHT DEC	C	RG-58 TYPE COAX
9907 E4X1640	1,640 FT	41.000 LB	GRAY, LIGHT DEC	C	RG-58 TYPE COAX
9907 E4X2500	2,500 FT	62.500 LB	GRAY, LIGHT DEC	C	RG-58 TYPE COAX
9907 E4X3280	3,280 FT	82.000 LB	GRAY, LIGHT DEC	C	RG-58 TYPE COAX
9907 E4X500	500 FT	12.500 LB	GRAY, LIGHT DEC		RG-58 TYPE COAX

Abbildung A.4: Datasheet : 9907 (2)

<http://www.belden.com/techdatas/metric/9907.pdf>

A.3 Dipol-Antenne

16 March 2008

W1034 Datasheet Version 1.0.

Wireless External Antenna for 2.4 GHz Applications, Tapered Design

Pulse Part Number: W1034

Features

- Attractive, tapered design
- For WLAN devices using WiFi (802.11b/g), Bluetooth® and ZigBee™
- Omni-directional radiation pattern provides broad 360° coverage
- One-quarter wavelength dipole configuration
- Connection and color options easily integrate with OEM designs

Color Options

- Black*
- Gray (Pantone cool gray 8C)
- Gray (Pantone 429C)
- Gray (Pantone cool gray 7C)

Connector Options

- Reverse SMA (Male)*
- SMA (Male)

* Default Configuration – Please contact Pulse Applications Engineering for assistance in ordering colors and connectors

Weight.....19.5 grams
Carton.....20/bag; 500/carton
Dimensions: Inches
mm
Unless otherwise specified, all tolerances are $\pm \frac{.010}{.25}$

Electrical Specifications @ 25 °C

Note: This part number is lead-free and RoHS compliant. No additional suffix or identifier is required.

Antenna Part No.	Frequency [GHz]	Gain [dBi]	Impedance [Nom]	VSWR	Polarization	Electrical Length	Radiation	Color
W1034	2.4 – 2.5	2.0	50 Ω	≤ 2.0	Vertical	1/4, dipole	Omni	Black

Pulse Antennas
Takatie 6
90440 Kempele, Finland
Tel: +358 207 935 500
Fax: +358 207 935 501
www.pulseeng.com/antennas

External Antennas Sales Contacts

USA	858 674 8100	Shanghai	86 21 32181071
UK	44 1483 401 700	China	86 769 5538070
France	33 3 84 35 04 04	Taiwan	886 2 26980228
Singapore	65 6287 8998		

Abbildung A.5: Dipol Antenne (1)
<http://www.mouser.com/ds/2/336/W1028.W1030-188421.pdf>

W1034 Datasheet Version 1.0.

March 2008 17

Wireless External Antenna for 2.4 GHz Applications, Tapered Design

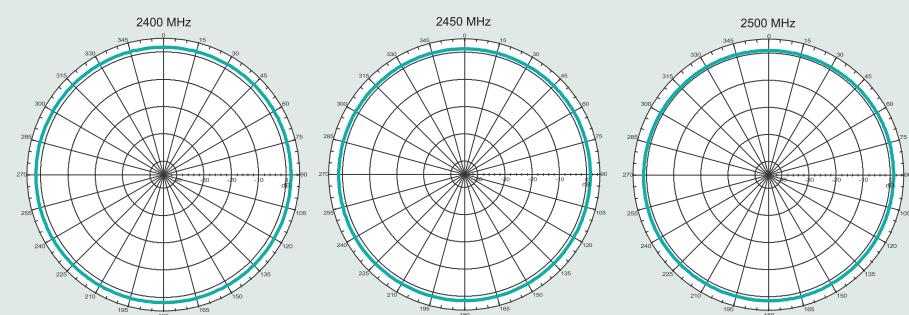
Pulse Part Number: W1034

Application Notes

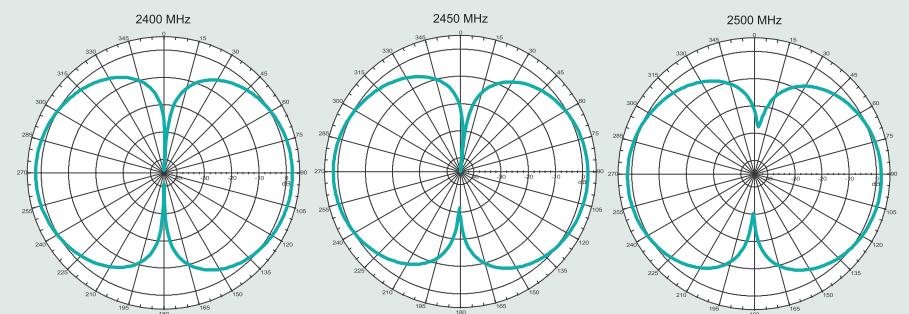
Omni-directional antennas provide a uniform, donut-shaped, 360° radiation pattern. The omni-directional pattern is suitable for point-to-multipoint broadcasting in all directions. This antenna is primarily used for WLAN applications. However, it can also be used for a variety of other applications within the specified frequency range. When used as an access point, the antenna is ideally located at the center of the coverage area.

Gain Performance W1034

Horizontal Position



Vertical Position



© 2008 All Rights Reserved.

Abbildung A.6: Dipol Antenne (2)
<http://www.mouser.com/ds/2/336/W1028.W1030-188421.pdf>

Animation Dipol-Antenne

<https://www.youtube.com/watch?v=HGPMror2syC>

B Aufgaben-Verzeichnis

Aufgabe 1.1 : verbale Kommunikation	4
Aufgabe 1.2 : Redundanz in der deutschen Sprache	6
Aufgabe 2.3 : Leitungsmaterial	13
Aufgabe 2.4 : Widerstandsbaufom	13
Aufgabe 2.5 : Kapazität	14
Aufgabe 2.6 : Kapazität erhöhen	14
Aufgabe 2.7 : Induktivität	14
Aufgabe 2.8 : mech. Eigenschaften einer Induktivität	14
Aufgabe 2.9 : Digital vs. Analog	16
Aufgabe 2.10 : Verschiebung von Ladung	17
Aufgabe 2.11 : NAND-Gate umbauen	25
Aufgabe 2.12 : NAND-Gate umbauen (2)	25
Aufgabe 2.13 : De-Glitch	28
Aufgabe 2.14 : Hex, Binäre, Dezimal Umrechnung	32
Aufgabe 3.15 : Zwischenfrage	40
Aufgabe 3.16 : Bit and Bitrates	43
Aufgabe 3.17 : Serial Bit and Bitrates	43
Aufgabe 3.18 : kopieren	44
Aufgabe 3.19 : Tertiär Signal	46
Aufgabe 3.20 : welches Signal fehlt bei RS-232?	47
Aufgabe 3.21 : idle Signalpegel bei RS-232	49
Aufgabe 3.22 : RS-232 Baudrate	49
Aufgabe 3.23 : RS-232 Verbinden von zwei Teilnehmern	50
Aufgabe 3.24 : RS-232 Verbinden von mehr als zwei Teilnehmern?	50
Aufgabe 3.25 : UART Quiz	53
Aufgabe 3.26 : UART Quiz (2)	53
Aufgabe 4.27 : Standards Definition	59
Aufgabe 4.28 : Standard Gremium	59
Aufgabe 4.29 : Welches Gremium ist das?	59
Aufgabe 4.30 : ISO/OSI Modell	69
Aufgabe 4.31 : Protokoll Stack	69
Aufgabe 5.32 : Dämpfung eines kaskadierten Leitungssystems	76
Aufgabe 5.33 : Skin Effekt	79
Aufgabe 5.34 : allg. Zweidrahtsystem	80
Aufgabe 5.35 : Filtereigenschaften eines allg. Zweidrahtsystem	81
Aufgabe 5.36 : Signallaufzeit in einem Leiter	91
Aufgabe 5.37 : Signallaufzeit in einem Leiter (2)	91
Aufgabe 6.38 : max Sendedistanz	99
Aufgabe 9.39 : Galvanische-Trennung mit NRZ /NRZI Polar Leitungscode	129
Aufgabe 9.40 : HDB3 Sequenz	131
Aufgabe 9.41 : AMI Bipolar Sequenz	132
Aufgabe 9.42 : HDB3 Sequenz	133
Aufgabe 9.43 : Manchester Code	133
Aufgabe 9.44 : NRZI-Polar	136
Aufgabe 9.45 : NRZ-Unipolar	136
Aufgabe 10.46 : Parity-Bit	143
Aufgabe 10.47 : Parity-Bit	143
Aufgabe 10.48 : CRC-5-USB	148
Aufgabe 10.49 : CRC-4	149

Aufgabe 12.50	171
Aufgabe 12.51	171
Aufgabe 12.52	175
Aufgabe 12.53	175
Aufgabe 13.54	182
Aufgabe 13.55	182
Aufgabe 13.56	183
Aufgabe 13.57	186
Aufgabe 13.58	186
Aufgabe 13.59	187
Aufgabe 15.60	213
Aufgabe 17.61	231
Aufgabe 17.62	232
Aufgabe 17.63	232
Aufgabe 17.64	239
Aufgabe 17.65	240
Aufgabe 17.66	240
Aufgabe 17.67	240

Tabellenverzeichnis

1.1 Japanische Kanji's und ihre deutsche Wort Übersetzung	7
2.1 Formel-Zeichen / -Einheiten : Ohmsches Gesetz	10
2.2 Formel-Zeichen / -Einheiten : Ohmscher Widerstand	10
2.3 Formel-Zeichen / -Einheiten : Kapazität	11
2.4 Formel-Zeichen / -Einheiten : Induktivität	12
2.5 div. Materialeigenschaften	13
2.6 div. Logik-Standards	20
2.7 Hex-Dezimal Umrechnung	31
2.8 Bits and Bytes	33
2.9 Präfixe bei Binären Zahlen	33
2.10 Treiber Schaltungen	34
2.11 Pull-Up / Pull-Down Schaltungen	34
2.12 Detektor Schaltungen	35
3.1 RS-232 definerte Baudraten	52
4.1 Organisation von Standardisierungsgremien	57
4.2 Wichtige Standardisierungsgremien	57
4.3 Internet Standardisierung	58
4.4 ISO/OSI Referenzmodell	62
4.5 Bestandteile des TCP/IP Protokoll Stacks	67
4.6 diverse Internet Protokolle	67
5.1 Formel-Zeichen / -Einheiten : Skin Effekt	78
5.2 ISO / IEC-11801 : Bezeichnungsschema	82
5.3 Formel-Zeichen / -Einheiten : Koaxial Kabel	86

5.4 Formel-Zeichen / -Einheiten : Ausbreitungsgeschwindigkeit	90
6.1 Formel-Zeichen / -Einheiten : Wireless- / Drahtlos-Kommunikation	100
6.2 ISM Frequenzbänder	100
6.3 WLAN Standards	101
6.4 RFID Frequenzbänder	102
6.5 NB-IoT Protokolle / Technologien	102
7.1 Impedanz Standards	107
7.2 Reflexionsfaktor : Interpretation und Wertebereich	112
7.3 Formel-Zeichen / -Einheiten : Reflexionsfaktor / Impedanz	113
9.1 Vergleich der diskutierten Codes	133
9.2 4B5B Codierung	135
10.1 Formel-Zeichen / -Einheiten : Binär symmetrische Kanal Modell	138
10.2 Verschiedene CRC-Generatorpolynome [W12]	146
11.1 Netzwerk Ausdehnung	162
12.1 Felder im Dataframe - Zweck	168
13.1 Hamming-Codes für $d_{min} = 3$	181
13.2 BCH-Codes : [O1]	188
16.1 USB Handshake Packet PID's	219
17.1 IPv4 Datagram Header	237
17.2 IPv6 Datagram Header	238
17.3 IPv4 Besondere Netzwerkadressen	239
18.1 X224 : Datenpaket Type	248
18.2 Well-known TCP / UDP Ports https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers	252
18.3 Beispiel Router NAT Tabelle	254
18.4 Static NAT Beispiel : Router Konfiguration	256
19.1 ISO/IEC 11801:2015	258
19.2 Abkürzungen ISO/IEC 11801	259

Abbildungsverzeichnis

1.1 Kommunikation	2
1.2 Einfaches Kommunikationsmodell	2
1.3 Beispiel eines Sender- / Empfänger-Systems	3
2.1 sehr einfaches Modell eines Ohmschen Widerstandes	10
2.2 einfacher Plattenkondensator (Rand-Effekte vernachlässigt)	11
2.3 einfache Induktivität (parallele Leiter)	12
2.4 Analog Digital Wandlung	16
2.5 Analogen (Thermisches) Rauschen	18
2.6 Mit Rauschen überlagertes Signal	18

2.7	Restauriertes Signal	19
2.8	Digitales Signal unterteilt nach dem TTL-Standard [W40]	20
2.9	NOT Gate («Nicht» Gatter)	21
2.10	Buffer Gate («Buffer»)	21
2.11	NAND Gate («Nicht-Und» Gatter)	22
2.12	AND Gate («Und» Gatter)	22
2.13	NOR Gate («Nicht-Oder» Gatter)	23
2.14	OR Gate («Oder» Gatter)	23
2.15	XOR Gate («Exklusiv-Oder» Gatter)	24
2.16	XNOR Gate («Exklusiv-Nicht-Oder» Gatter)	24
2.17	Schema und Wahrheitstabelle : Glitches (1)	26
2.18	Timing Diagram : Glitches (2)	26
2.19	D-FlipFlop	27
2.20	D-FlipFlop Register	29
2.21	D-FlipFlop Schiebe-Register	30
2.22	Binär- / Hexadecimal-Umrechnung	31
2.23	Direktivität	35
3.1	Einfaches Sender- / Empfänger-Systems (LPT / COM)	38
3.2	parallele Datenübertragung	38
3.3	parallele Datenübertragung : Beispiel Timing Diagram	39
3.4	serielle Datenübertragung	39
3.5	Beispiel eines Symbols in der parallelen / seriellen Datenübertragung	41
3.6	Quaternär Signal	45
3.7	Parallel Port Female D-Sub25 Stecker https://upload.wikimedia.org/wikipedia/commons/f/fa/Parallel_computer_printer_port.jpg	46
3.8	Parallel Port Female D-Sub25 Pinout	46
3.9	Serial Port Male / Female D-Sub9 Stecker http://media.digikey.com/Photos/CnC%20Tech/720-10020-00300.jpg	47
3.10	Serial Port Female D-Sub9 Pinout	47
3.11	RS-232 Leitungscode	48
3.12	UART Leitungscode	51
4.1	Standardisierungsgremien	58
4.2	Schichtenmodell	60
4.3	Datenübertragung im Schichtenmodell	61
4.4	ISO/OSI Modell	62
4.5	OSI Layer 1-3 : Simple	64
4.6	OSI Layer 1-3 : Tx	65
4.7	Verschachtelung	65
4.8	OSI Layer 1-3 : Rx	66
4.9	TCP/IP Protokoll Stack	67
4.10	Beispiel Schichtenmodell	68
4.11	Austausch von Schichten im OSI Layer Modell	68
5.1	Bitraten-Dreieck	73
5.2	Signal-zu-Rauschleistungs Verhältnis	73
5.3	Frequenzgang eines einfachen RC-Tiefpass	74
5.4	(Simple) Übersicht verschiedener Leitungssysteme	74
5.5	Ersatzschaltbild eines realen Leiters	75
5.6	Kaskadieren einzelner Leitungsstücke	76
5.7	LC-Filter	77
5.8	sehr einfaches Modell eines Ohmschen Widerstandes	77
5.9	Skin Effekt : Eindringtiefe	78
5.10	Allgemeines Zweidrahtleitung	79
5.11	Twisted-Pair	81
5.12	Auszug aus Application Note Lattice : TN1752 : (Seite 7) http://www.latticesemi.com/view_document?document_id=47960	82

5.13	...	83
5.14	LVDS (Differenzielle Stromgekoppelte Endstufe)	84
5.15	Balun	84
5.16	$\frac{\lambda}{4}$ Transformer (quarterwave transformer)	85
5.17	Drei $\frac{\lambda}{4}$ Transformer für unterschiedliche Frequenzbereiche http://www.atechfabrication.com/images/coax_balun_002.jpg	85
5.18	Koaxial Kabel	86
5.19	HJ8-50B, Air Dielectric Coaxial Cable http://www.commscope.com/catalog/imagesCache/0000023/t006_r00740_v5.jpg	87
5.20	Aufbau eines Lichtwellenleiters	88
5.21	Monomode Faser	89
5.22	Multimode Faser	89
6.1	EM Spektrum https://upload.wikimedia.org/wikipedia/commons/thumb/1/15/Electromagnetic_spectrum_c.svg/1024px-Electromagnetic_spectrum_c.svg.png	94
6.2	Strahlungsverhalten von EM-Wellen bei tiefen Frequenzen	94
6.3	Strahlungsverhalten von EM-Wellen bei hohen Frequenzen	95
6.4	Strahlungsverhalten von EM-Wellen welche von der Ionosphäre reflektiert werden	95
6.5	Antenne = Aufgeklappter Schwingkreis	96
6.6	Energieverteilung einer sich ausbreitenden EM-Welle	96
6.7	Freiraum Übertragungsleistung	97
6.8	einfaches Beispiel einer Antennen-Strahlungscharakteristik	98
6.9	Ein TP-LINK Router mit 4x4 MIMO Technologie https://static.digitecgalaxus.ch/Files/6/7/1/3/8/9/4/AD7200_un_V1_1185_large_2.00_20160125105911.jpg?fit=inside%7C464:368&output-format=progressive-jpeg	101
7.1	Telegrafien-Modell und -Gleichung	106
7.2	Schema : abgeschlossene Leitung	108
7.3	Signalform : abgeschlossene Leitung	108
7.4	Schema : offene Leitung	109
7.5	Signalform : offene Leitung	109
7.6	Schema : kurzgeschlossene Leitung	110
7.7	Signalform : kurzgeschlossene Leitung	110
7.8	Schema : beliebig abgeschlossene Leitung	111
7.9	Signalform : beliebig abgeschlossene Leitung	111
7.10	Schema : Reflexionsfaktor	112
7.11	Time-Domain Reflektometer SQ7270 https://upload.wikimedia.org/wikipedia/commons/0/08/OTDR_-_Yokogawa_AQ7270_-_1.jpg	114
8.1	even Parity-Bit	119
8.2	UART im OSI-Modell	121
9.1	Einfache Darstellung einer Asynchronen Verbindung	124
9.2	Timing-Diagram einer ideal synchronisierten Übertragung	124
9.3	Timing-Diagram einer unsynchronisierten Übertragung (fatal)	124
9.4	Takt-Rückgewinnung	126
9.5	Empfänger Kopplung	126
9.6	Leitungscode - Übersicht	127
9.7	NRZ Unipolar	128
9.8	NRZ (Bi-) Polar	128
9.9	RZ Unipolar	129
9.10	AMI Bipolar	130
9.11	HDB3 (odd parity = [000V])	130
9.12	HDB3 (even parity = [B00V])	131
9.13	Manchester Code	132
9.14	2B1Q Code	134
9.15	MLT-3 Codierung	134

10.1	BSK Model : Binary Symmetric Channel Model	138
10.2	Single-Bit Error	139
10.3	Burst Error	139
10.4	Theorie zur Fehlerdetektion	140
10.5	Beispiel : Repetition Code : Sendesignal	141
10.6	Beispiel : Repetition Code : Empfangssignal	141
10.7	Parity-Bit : even / odd	142
10.8	Beispiel : even parity-bit (inkl. Versagen)	142
10.9	Bitstrom mit Infodaten und Checksumme / Redundanz	144
10.10	CRC Berechnung : CRC-4 mit $g(x) = x^4 + x^1 + 1$	145
10.11	CRC Hardware-Umsetzung	147
11.1	Netzwerk Topologie	152
11.2	Bus Topologie	153
11.3	Stern Topologie	154
11.4	Linien Topologie	155
11.5	Ring Topologie	155
11.6	Baum Topologie	156
11.7	Mesh Topologie	157
11.8	Simplex	158
11.9	Halb-Duplex	158
11.10	(Voll-) Duplex	159
11.11	RS-232 : Verkabelung	160
11.12	Shared Medium : Shared-Simplex	160
11.13	Shared Medium : Shared-Halb-Duplex	161
11.14	Shared Medium : Shared-Duplex	161
11.15	Beispiel I2C	162
11.16	Beispiel RS-485	163
12.1	Übersicht Datenfluss	166
12.2	Asynchron	166
12.3	Isochron	167
12.4	Plesiochron	167
12.5	Synchron	168
12.6	Allgemeiner Dataframe	168
12.7	UART Dataframe	172
12.8	I2C Dataframe	172
12.9	Ethernet Dataframe	173
12.10	Bit-Stuffing Beispiel : Nach 5 · 1 wird eine 0 eingeschoben	174
13.1	Forward Error Correction - Übersicht	178
13.2	Hamming Code (3,1)	179
13.3	Hamming Distanz	180
13.4	Hamming Code (7,4)	183
13.5	Hamming(7,4) Codewort	184
13.6	BCH Code : Beispiel Multiplikationsmethode	188
13.7	BCH Code : Syndrom berechnen	189
13.8	non-recursive Conv.Code $R_C = \frac{1}{2}$	190
13.9	recursive Conv.Code $R_C = \frac{1}{2}$	190
13.10	Zustandsdiagramm für $R_C = \frac{1}{2}$ non-recursive Trellis	191
13.11	Trellis Diagram	192
13.12	Trellis Encoder (Beispiel)	193
13.13	Trellis Diagram	193
14.1	Beispiel Polling	197
14.2	Beispiel Token-Passing	198
14.3	Ausgangslage CSMA/CD	199
14.4	CSMA/CD Medium verkehr «mithören» Ear Icon : Icon made by Freepik from www.flaticon.com	200

14.5 CSMA/CD shared-medium busy Timer / Transmitter Icon : Icon made by Freepik from www.flaticon.com	200
14.6 CSMA/CD gleichzeitiges Senden Transmitter Icon : Icon made by Freepik from www.flaticon.com	201
14.7 CSMA/CD Collision Crash Icon : Icon made by Freepik from www.flaticon.com	201
14.8 CSMA/CD recovery	201
14.9 CSMA/CD Ablaufschema	202
15.1 Multiplexer / Demultiplexer	206
15.2 FDM : Radiofrequenzen Zürich	206
15.3 Time Division Multiplexing	207
15.4 TDM : Asymmetrische Datenraten	208
15.5 CDM Orthogonal Code	208
15.6 CDM : Modulation	209
15.7 CDM : Demodulation	209
15.8 CDM : Spektrale Verschmierung	210
15.9 3 Bit Pseudo-Random Generator	210
15.10 ACDM : Modulation	211
15.11 ACDM : Demodulation	211
15.12 SDM : GSM Zellenverteilung	212
15.13 SDM : WLAN Frequenz Kanal Verteilung 2.4GHz	213
15.14 WDM : Aufbau (stark vereinfacht)	214
15.15 Multiplex Verfahren TDM / FDM / CDM	215
15.16 Multiplexing : TDM / FDM / CDM Combined	215
16.1 ACK / NACK einfaches Beispiel	218
16.2 I2C Protokoll	218
16.3 USB Packet Types	219
16.4 CTS / RTS einfaches Beispiel	220
16.5 RS232 : RTS / CTS	220
16.6 Stop-and-wait AQR : Ablaufschema	221
17.1 OSI Layer 3 (Network Layer)	226
17.2 unicast	226
17.3 unicast	226
17.4 broadcast	227
17.5 anycast	227
17.6 geocast	227
17.7 Vermittlung von Daten	228
17.8 Beispiel Leitungsvermittlung (Verbindungsorientiert)	229
17.9 Beispiel Paketvermittlung	230
17.10 Repeater	233
17.11 Bridge / Switch	233
17.12 Bridge / Switch : bad example	234
17.13 Bridge / Switch : good example	234
17.14 Router	235
17.15 IPv4 Datagram Header	236
17.16 IPv6 Datagram Header	237
17.17 Beispiel IPv4 Adresse	238
17.18 kleines Netzwerk	241
17.19 IP-Packet	241
17.20 komplizierteres Netzwerk	242
17.21 IP-Packet 1	242
17.22 IP-Packet 2	243
18.1 OSI Layer 4 (Transport Layer)	246
18.2 Vergleich MAC / IP Ebene	246
18.3 Vergleich MAC / IP Ebene (CRC)	247
18.4 X224 : Verbindungsau- und abbau	248

18.5	UDP Header	251
18.6	TCP Header	251
18.7	UDP / TCP Ports Example	251
18.8	WebServer Anfrage im gleichen Netzwerk	252
18.9	WebServer Anfrage im gleichen Netzwerk : UDP Paket	253
18.10	WebServer Anfrage im Netzwerk	253
18.11	WebServer Anfrage im Netzwerk : A) UDP Paket	254
18.12	WebServer Anfrage im Netzwerk : B) UDP Paket	254
18.13	WebServer Anfrage im Netzwerk : C) UDP Paket	255
18.14	WebServer Anfrage im Netzwerk : D) UDP Paket	255
18.15	Static NAT Beispiel	256
19.1	ISO/IEC 11801 Gebäudeverkabelung	259
A.1	Datasheet: 70006E(1) http://belden.catalog.belden.com/en/Products/TechDataSheetPdf?product=PF_70006E	263
A.2	Datasheet: 70006E(2) http://belden.catalog.belden.com/en/Products/TechDataSheetPdf?product=PF_70006E	264
A.3	Datasheet : 9907 (1) http://www.belden.com/techdatas/metric/9907.pdf	265
A.4	Datasheet : 9907 (2) http://www.belden.com/techdatas/metric/9907.pdf	266
A.5	Dipol Antenne (1) http://www.mouser.com/ds/2/336/W1028.W1030-188421.pdf	267
A.6	Dipol Antenne (2) http://www.mouser.com/ds/2/336/W1028.W1030-188421.pdf	268

