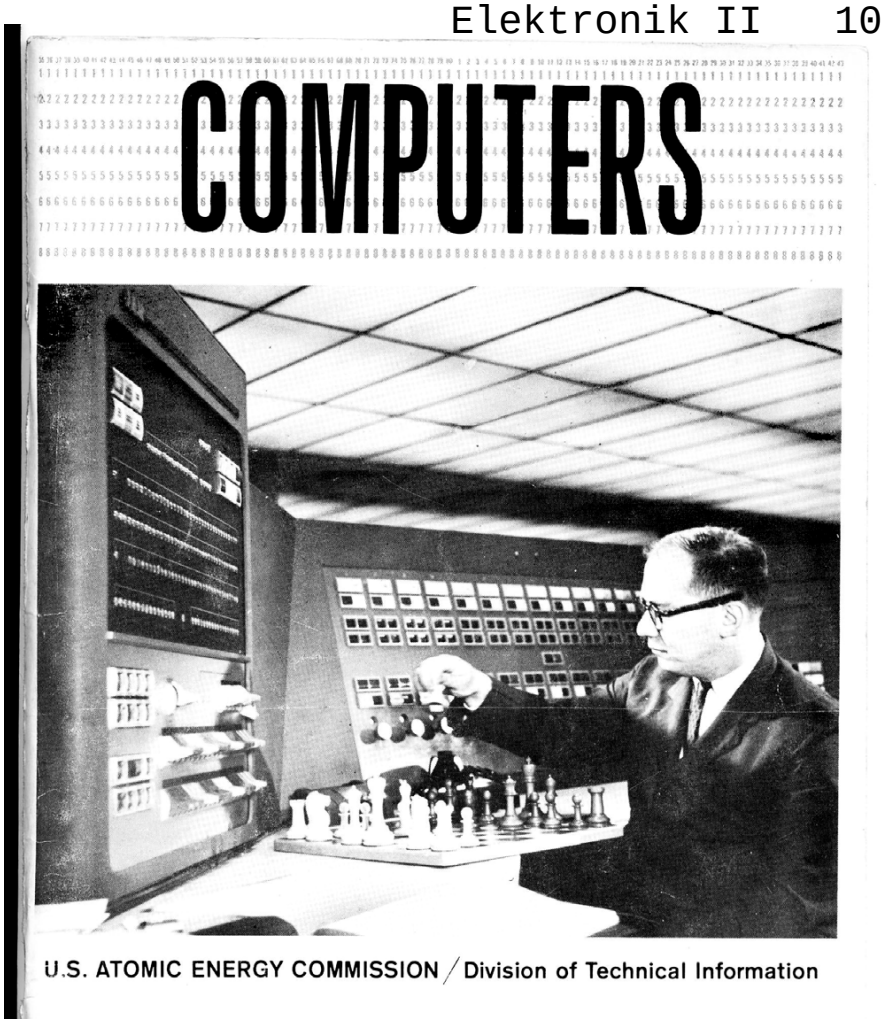


November 1968

I. Einführung in die Digitalelektronik

Grundbegriffe, Wahrheitstabellen:

Die Begriffe „analog“ und „digital“ stammen aus der Rechentechnik:

Analog-Rechner benötigt zur Darstellung von Zahlenwerten eine Analogie-Größe (z.B. eine Spannung)

Maßstab	1 = 1 V	1 = 10 µV
Anzeigen	1,36 = 1,36 V	10530 = 0,1053 V

Analoge Messgeräte (Zeigerinstrumente)

Digital-Rechner (lat. *digitus*, der Finger)

Rechnen mit Finger: Finger vorhanden = 1
Finger nicht vorhanden = 0

Digitalelektronik baut auf der Notwendigkeit auf, das Mengenproblem bei der Bewältigung von Rechenaufgaben zu lösen.

Historie:

1623 Wilhelm Schickard (Tübingen) baut erste Rechenmaschine.

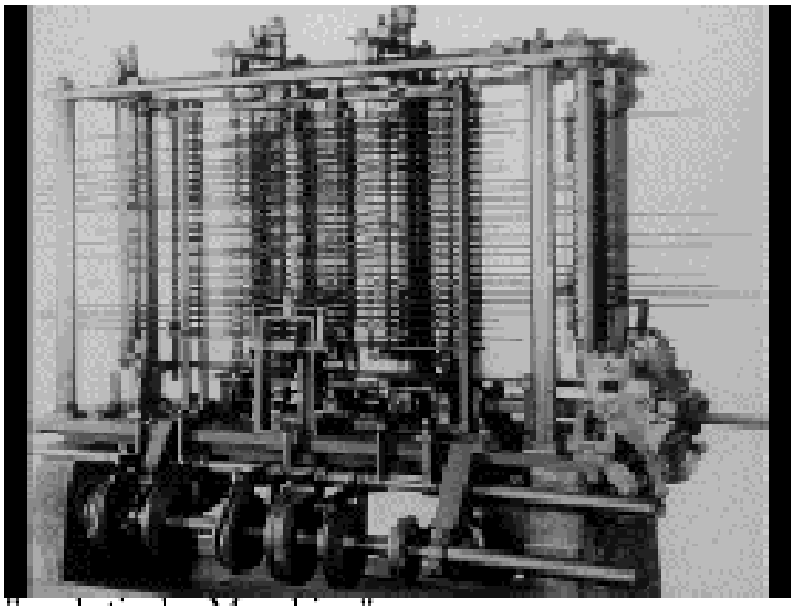
1642 Blaise Pascal: Erstes mechanisches Rechenwerk für die Addition und Subtraktion mit durchlaufendem Zehnerübertrag

1671 Gottfried Wilhelm Leibniz: Multiplikation

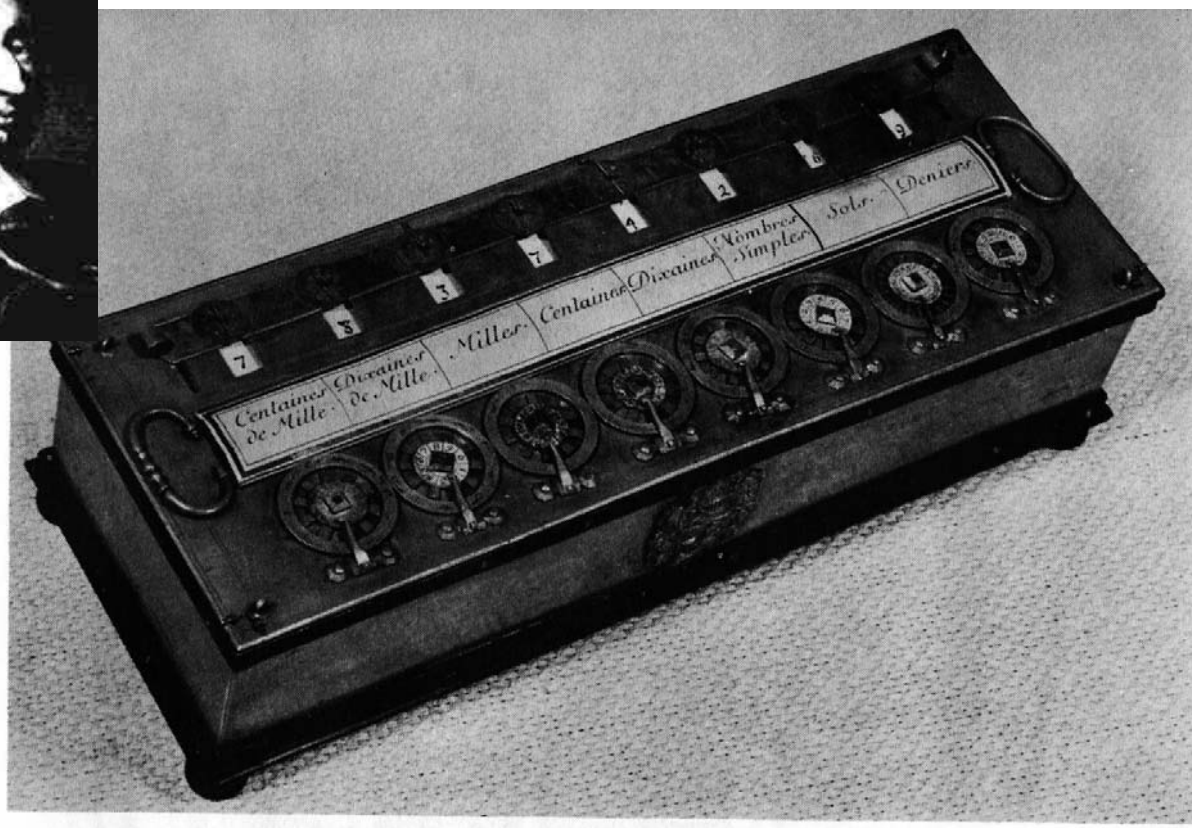
1825 Charles Babbage: "analytische Maschine" --> "Programm-Speicherung".

Leider nur theoretisch (inkl. dualer Logik und Ansätze Boole´scher Algebra).

Maschine funktionierte allerdings nicht.



"analytische Maschine"



Blaise Pascal designed this mechanical computer in 1642. It is a digital, decimal machine, operated with a stylus. Numbers are carried to adjacent wheels by gears inside the machine.

Zahlensysteme

Ziffern <-> Ziffernwert

Zahlen <-> Zahlenwert

Zahlen: Folge von Ziffern

Jedes Zahlensystem hat so viele Ziffern, wie die Basis des Zahlensystems groß ist.

kleinste Ziffer: 0

größte Ziffer: B –1

z.B: Dezimalsystem

B = 10, Ziffern 0...9

Zahl: $297 = 2 * 10^2 + 9 * 10^1 + 7 * 10^0$

$B^0 = 10^0 = 1$
$B^1 = 10^1 = 10$
$B^2 = 10^2 = 100$

Beim Dezimalsystem ist die Wertigkeit durch den Stellenwert erkennbar:

MSD						LSD
10^3	10^2	10^1	10^0	10^{-1}	10^{-2}	10^{-3}
1000	100	10	1	1/10	1/100	1/1000

Vergleich zwischen den vier wichtigsten Zahlensystemen	Oktalsystem	... 8^3	8^2	8^1	8^0	...	(8 Ziffern)
	Hexadezimalsystem	... 16^3	16^2	16^1	16^0	...	(16 Ziffern)
	Dualsystem	... 2^3	2^2	2^1	2^0	...	(2 Ziffern)

Dezimal-System	Hexa-dezimal	Oktal-System	Dual-System
0	0	0	0000
1	1	1	0001
2	2	2	0010
3	3	3	0011
4	4	4	0100
5	5	5	0101
6	6	6	0110
7	7	7	0111
8	8	10	1000
9	9	11	1001
10	A	12	1010
11	B	13	1011
12	C	14	1100
13	D	15	1101
14	E	16	1110
15	F	17	1111
16	10	20	10000

Datenorganisation:

Bits, Bytes

- digitale Verarbeitung von Daten:

  i.a. zwei Zustände

  (Schalterstellung, Spannungen, Ströme,Frequenzen, Magnetisierung...)

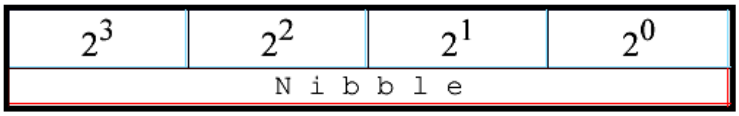
- optimal:

  zwei entgegengesetzte Zustände (an, aus)

  <=> kleinste Einheit:

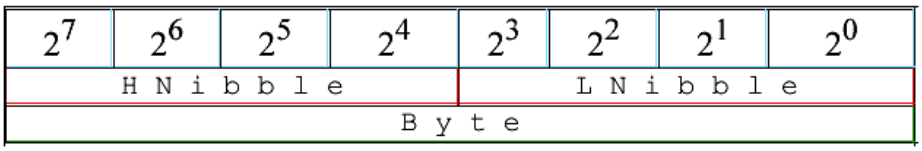
  BIT

- Zusammenfassung von 4 bits: **Nibble**



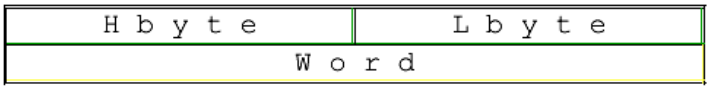
=> 16 Werte darstellbar

- Zusammenfassung von 8 bits (2 Nibbles) : **Byte**



=> Mit einem byte lassen sich 256 Werte darstellen

- Zusammenfassung von 2 byte (4 Nibbles, 16 bit) : **Word**



Desweiteren:

- Doubleword (DD- Format) 2^{32} Werte
- Quadword (DQ- Format) 2^{64} Werte
- Ausnahme: DF (defines farword): sechs Byte
- Speziell für Coprozessoren (Numerik):
  Zusammenfassung von zehn Bytes, DT (defines Tenword)
  <-> 80 bit- Format für Rechenoperationen mit Gleitpunktformat

Verarbeitung reeller Zahlen: Umrechnung zwischen den Zahlensystemen

$$1\,5\,2\,5\,3_{\text{Dez}} = \underset{2^{13}}{1}\,1\,1\,0\,1\,1\,1\,0\,0\,1\,0\,1\,0\,\underset{2^{0}}{1}_{\text{Dual}}$$

Wie können nun die Zahlen im Dualcode dargestellt werden?

- **Festpunktformat**:

  Bits 0 1 1 1 0 . 0 0 1
  Wert 8+4+2 Binärpunkt 1/8 = 14.125

Nachteil: Abstand zwischen größter und kleinster darstellbarer Zahl ist recht klein (im Beispiel: 2^5 - 1 ... 2^{-3}).

Zur Darstellung kleinerer Zahlen muß größte darstellbare Zahl eingeschränkt werden.

Die Position des Dezimalpunktes liegt für alle Zahlen fest!

- **Gleitpunktformat**:

  Einige bits des verwendeten Formats werden dazu verwendet, die Position des Binärpunktes in Bezug auf die übrigen bits anzuzeigen!

  Beispiel für Gleitpunkt in 8 bit-Wert

  Position des GP "Exponent" | eigentliche Zahl "Mantisse"

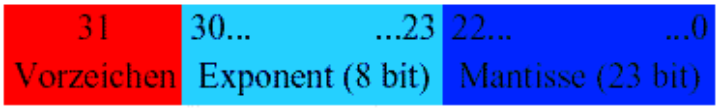
  Beispiel: 000 -> x.xxxx
  011 -> xxxx.x

  kleinste darstellbare Zahl = 000 00001 = 0.0001 = 1/16
  größte darstellbare Zahl = 111 11111 = 11111000.0 = 248

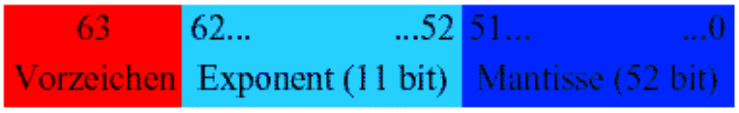
Ende 16.4.2008

Normung nach IEEE-754

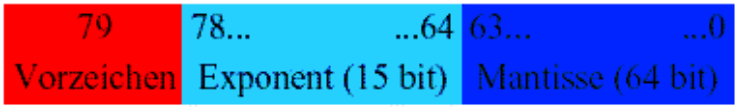
32 bit



64 bit



Double precision



=> **stets eine Approximation von R!**

Der Dual Code bildet die Grundlage für jeden 4 bit- Code. Mit dem 4 bit- Code kann man 16 Wertigkeiten darstellen.

Darstellung des Dezimalsystems im 4 bit-Code: **B**inär **C**odierte **D**ezimalzahl (BCD)

Dezimal: 0 9
Dual- 4: 0000 .... 1111 (sechs Redundanzen da Übertrag auf nächste Stelle bei 9!)

Redundanzen sind unerwünscht, da sie zusätzlichen Speicherplatz benötigen.
Die Bewertung von Codes geschieht nach den Redundanzen:

BCD: benötigt 3.32 bit => Redundanz 0.68 bit

Ein wichtiger Punkt für die Boole´sche Algebra ist die Komplementierbarkeit von Codes. (siehe Bool´sche Algebra). Das Zweierkomplement einer Hex-Zahl ist die Ergänzung auf die Basis 16!

Dies geht *nicht* beim einfachen BCD- Code.

⇨ Aiken-Code, Stibitz-Code (Excess 3) (komplementär und symmetrisch)

Im Einzelfall werden noch andere Codes verwendet.
Stichwort: n- schrittige Codes, 1- schrittige Codes

Beispiel:

	BCD	Gray-Code
7	0111	0100
8	1000	1100
	3-schrittig	1- schrittig

Aufbau des Aiken- und des Stibitz-(Exzeß-3-) Codes im Vergleich zum BCD-Code

Beispiel für Aiken-Code		
Dezimal-ziffer	Aiken-codiert	Binär-codiert
0	**0 0 0 0**	0 0 0 0
1	**0 0 0 1**	0 0 0 1
2	**0 0 1 0**	0 0 1 0
3	**0 0 1 1**	0 0 1 1
4	**0 1 0 0**	0 1 0 0
5	**1 0 1 1**	0 1 0 1
6	**1 1 0 0**	0 1 1 0
7	**1 1 0 1**	0 1 1 1
8	**1 1 1 0**	1 0 0 0
9	**1 1 1 1**	1 0 0 1

Beispiel für Stibitz-Code			
Dezimal-ziffer	Stibitz-codiert	BCD-codiert	Binär-codiert
0	**0 0 1 1**	0 0 0 0	0 0 0 0
1	**0 1 0 0**	0 0 0 1	0 0 0 1
2	**0 1 0 1**	0 0 1 0	0 0 1 0
3	**0 1 1 0**	0 0 1 1	0 0 1 1
4	**0 1 1 1**	0 1 0 0	0 1 0 0
5	**1 0 0 0**	0 1 0 1	0 1 0 1
6	**1 0 0 1**	0 1 1 0	0 1 1 0
7	**1 0 1 0**	0 1 1 1	0 1 1 1
8	**1 0 1 1**	1 0 0 0	1 0 0 0
9	**1 1 0 0**	1 0 0 1	1 0 0 1

- **Prüfbits**

Sie dienen der Fehlererkennung und Fehleranalyse bei der Übertragung bzw Speicherung von digitalen Daten.

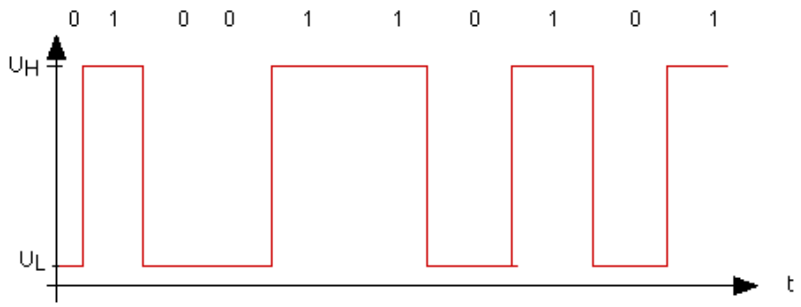
  - Erkennen der Fehler
  - Lokalisieren der Fehler
  - Sei tolerant gegen Mehrfachfehler

Beispiele: Paritätsbit bei ASCII Code (Quersumme der Binärzahl)

Hamming-Code (drei Prüfbits) erkennt Position eines erkannten Fehlers

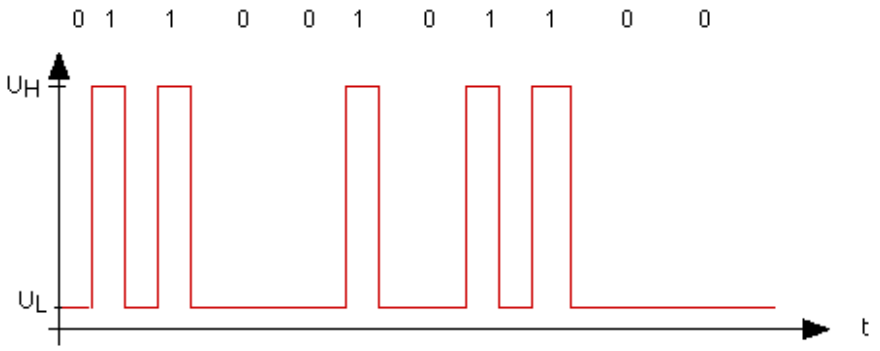
Serielle Datenübertragung und Leitungscodes

- NRZ- Codes (non return to zero)



  - nicht gleichspannungsfrei
  - keine Taktinformation

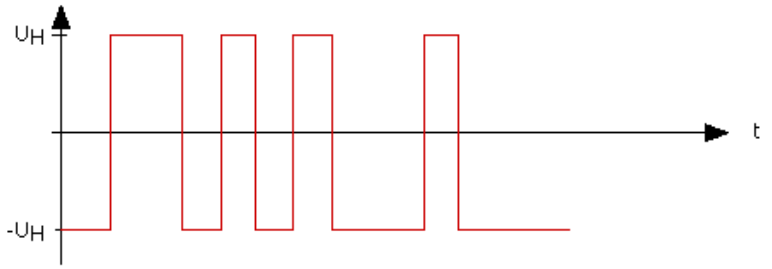
- RZ- Codes



  - Taktrückgewinnung einfacher

- CMI (coded mark inversion)
  - Doppelstromumtastung

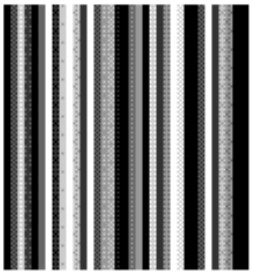
"1":	Sender	positiver Strom	Empfänger
"0":	Sender	negativer Strom	Empfänger



- weitere, kompliziertere Codes möglich

Ziel:
- geringer Gleichspannungsanteil
- Taktrückgewinnung
- Bandbreite des "Kabels" gering

=> Schnittstellen, Bussysteme



- Bar- Codes

diverse Standards, beruhen auf Reflexionsunterschieden

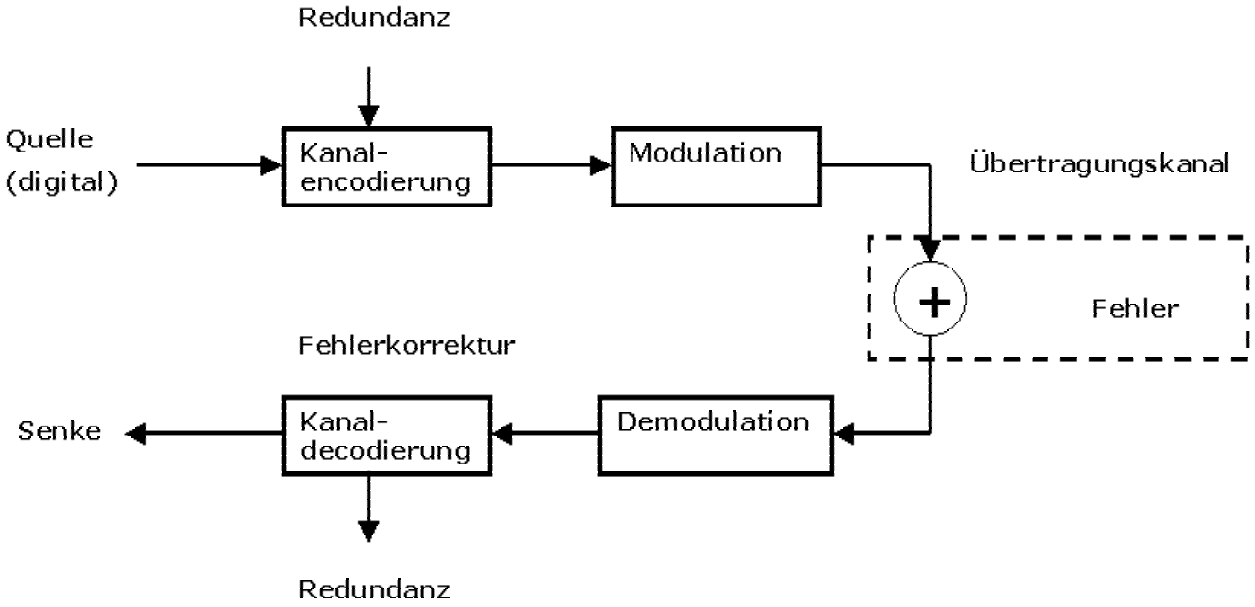
- Abtastung per sichtbaren Laser, IR, usw.
- Wahl der Druckfarbe!
- mögliche Größe des Barcodes

Kanalkodierung

Als Kanalkodierung bezeichnet man das Verfahren, digitale Daten bei der Übertragung über gestörte Kanäle durch Hinzufügen von **Fehlererkennungs- und korrekturzeichen** gegen Übertragungsfehler zu schützen.

Im Gegensatz dazu reduziert die **Quellenkodierung** die Ursprungsmenge digitaler Daten, indem es Irrelevanzen und Redundanzen entfernt.

Die Grundlagen für die Kanalcodierung stellt die **Kodierungstheorie** bereit.



Den Daten am Eingang eines **Übertragungskanals** werden Redundanzzeichen hinzugefügt und man dekodiert die Daten an seinem Ausgang.

Wenn die Zusatzinformationen lediglich auf einen Fehler hindeuten und eine Neuübertragung der Daten erfordern, spricht man von Rückwärtsfehlerkorrektur.

Genügt die Redundanzinformation, den Fehler zu korrigieren, handelt es sich um eine **Vorwärtsfehlerkorrektur**.

Je nach Verfahren entspricht der Kanalkodierung einer Verbesserung des Signal-Rausch-Verhältnises um mehrere dB.

Ein Datencode wird durch seine Rate gekennzeichnet: R = k / n, wobei k die Anzahl der Zeichen eines Wortes am Eingang des Kodierers ist und n die Anzahl am Ausgang. Es werden also k Symbole am Eingang auf n Symbole am Ausgang abgebildet. Eine kleine Rate (großes n) bedeutet einen höheren Anteil an der **Datenübertragungsrate**.

Durch geschicktes Verketten von Codes und das Streuen von Fehlern kann diese Rate oft erheblich vergrößert werden. Die Codes einer Compact Disc erreichen zum Beispiel eine Rate von 3 / 4.

siehe auch: Kryptografie

Wozu benötigt man Kanalkodierung?

- Rauschen, Abschattungen, Echos, Dopplereffekt usw. stören die Übertragung
- Störungen verursachen Fehler im Bitstrom
- Fehler wirken sich besonders gravierend bei komprimierten Daten aus

==> **Kanalkodierung zum Schutz vor Übertragungsfehlern**

- Weniger Sendeleistung für gleiche Übertragungsqualität nötig
- Bessere Qualität bei gleicher Sendeleistung

Funktionsweise der Kanalkodierung:

Der Kodierer fügt einem Informationswort ein oder mehrere Prüfbits hinzu.

Informationswort U	Prüfteil P
k Bit	(n-k) Bit

Redundanz erlaubt Fehlererkennung und/oder Fehlerkorrektur.

Beispiel 1: Wiederholungscode bzw. Repetition code (4,3)-RC

U_0	X	U_0	U_0	U_0

- Drei Fehler können erkannt werden
- Bei einfacher "Mehrheitsentscheidung" kann ein Fehler korrigiert werden.

Beispiel 2: Single Parity Check Code mit 2 Infobit und einem Prüfbit (3,2)-SPC

U_0	U_1	X	P_0

- Bei diesem SPC wird auf **gerade Parität** ergänzt (d.h. Anzahl der Einsen in Codewort ist immer gerade).
- P0=(U0+U1) mod 2
- Ein Fehler kann erkannt, aber nicht korrigiert werden

Verkettung von Codes, Produktcodes

U_0	U_1	P_0
U_2	U_3	P_1
U_4	U_5	P_2
U_6	U_7	P_3
P_4	P_5	

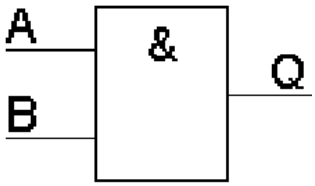
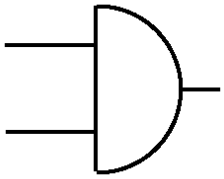
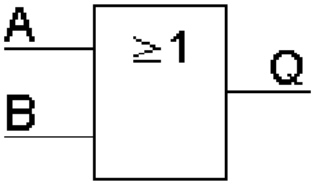
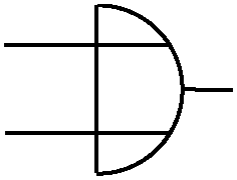
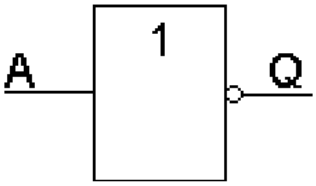
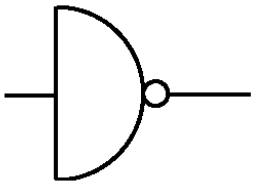
- Anordnung der Informationsbits in einer Matrix U.
- Horizontale Codierung C: Zeilenweise Erzeugung der Prüfbits P_0...P_3
- Vertikale Codierung C': Spaltenweise Erzeugung der Prüfbits P_4...P_5

- Codes C und C' können unterschiedlich sein, z.B. 3,2-SPC und 5,4-SPC
- SPC-Codes können nur einen Fehler erkennen
- Produktcode kann drei Fehler mit Sicherheit erkennen und maximal einen Fehler korrigieren

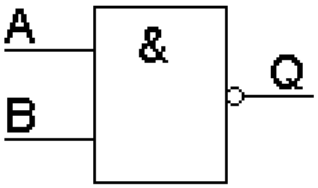
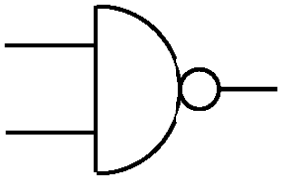
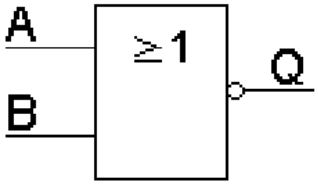
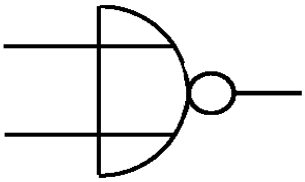
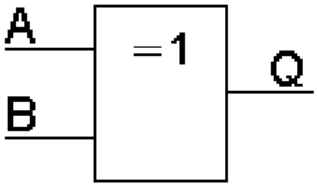
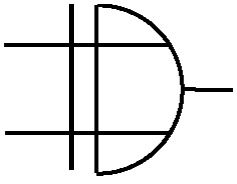
1	0	1
0	1	0
0	0	0
1	1	0
0	1	

- Fehlererkennung: Summe in Spalte und Zeile ungerade
- Fehler befindet sich am Schnittpunkt der fehlerhaften Spalte und Zeile

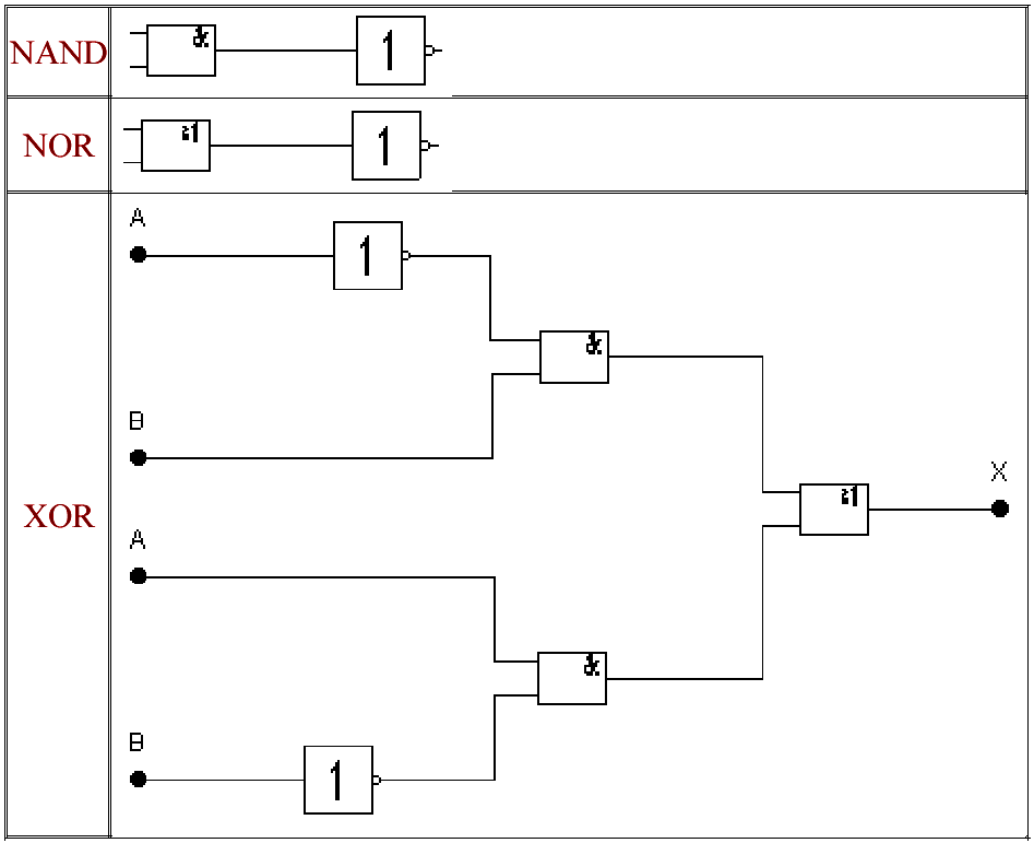
Digitale Schaltkreise: Grundschaltungen

Typ	Symbol	Symbol (alte Norm)	Schaltfunktion	Funktionstabelle		
				Eingang		Ausgang
				A	B	X
AND			X=A*B	0 0 1 1	0 1 0 1	0 0 0 1
OR			X=A+B	0 0 1 1	0 1 0 1	0 1 1 1
NOT			$X=\overline{A}$	0 1		1 0

Weitere Grundschaltungstypen, die aus den ersten drei abgeleitet werden können:

Typ	Symbol	Symbol (alte Norm)	Schaltfunktion	Funktionstabelle		
				Eingang		Ausgang
				A	B	X (Q)
NAND			$X=\overline{A*B}$	0 0 1 1	0 1 0 1	1 1 1 0
NOR			$X=\overline{A+B}$	0 0 1 1	0 1 0 1	1 0 0 0
XOR			$X=\overline{A}*B+A*\overline{B}$	0 0 1 1	0 1 0 1	0 1 1 0

Ersatzschaltungen:



Testgeräte der Digitalelektronik

Ziel einer Messung
(Test, Fehleranalyse)

- Erkenne
  High / Low Pegel
- Überprüfe
  Wahrheitstabelle
- Beobachte
  zeitlichen Verlauf