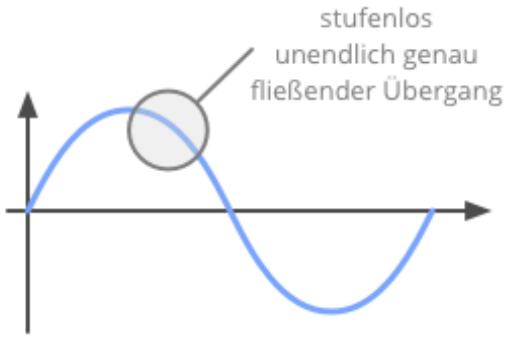
FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Grundlagen der Informationstechnik
Digitale Daten

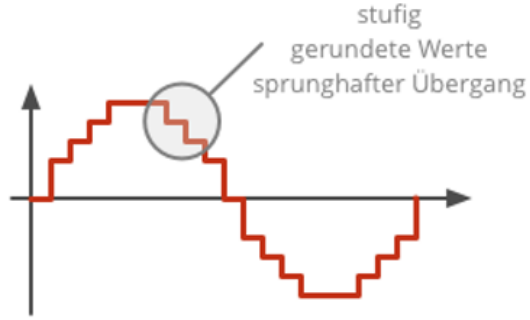
Peter Holzkämper

Digitale Daten
Grundlagen













Analoge Signale sind stufenlos und liefern theoretisch unendlich genaue Informationen (kontinuierliche Werte)

Digitale Signale sind mehr oder weniger „verlustbehaftet“ (diskrete Werte)

Quelle: https://13db.de/wissen/digitalsignale/

Digitale Daten
Grundlagen



Bei der analogen Uhr können wir zwischen den Sekunden die Uhrzeit ablesen. Wir haben wieder ein kontinuierliches Signal, welches wir unendlich genau beobachten können.

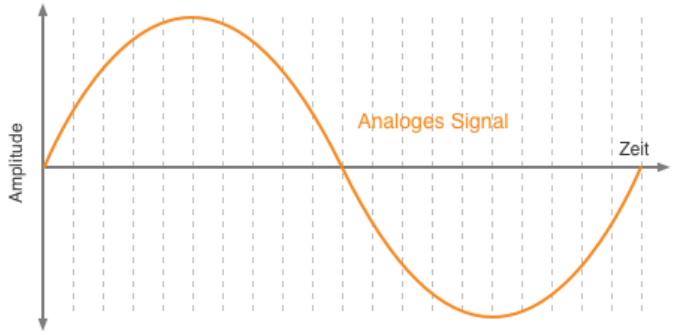


Bei der digitalen Uhr erhalten wir dagegen nur jede Sekunde ein neues Signal. Es handelt sich dabei um ein diskretes Signal. Es gibt keine Zwischenwerte, die wir auswerten können.

Quelle: https://13db.de/wissen/digitalsignale/

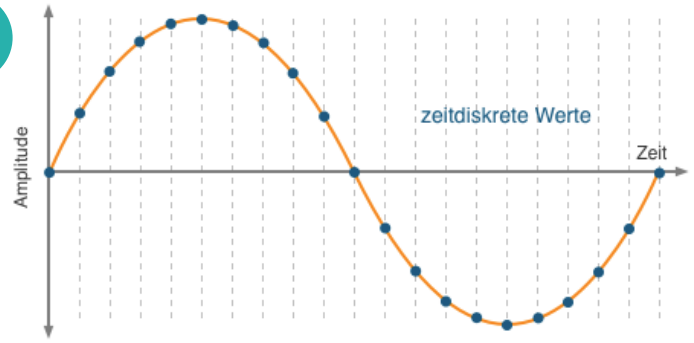
Digitale Daten
Grundlagen

1 Analogsignal vor der zeitlichen Abtastung

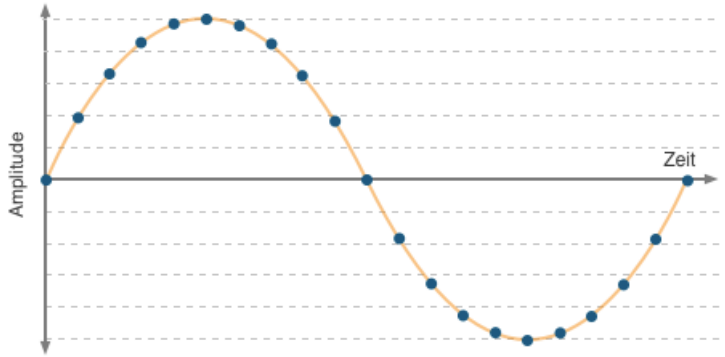


Nach der zeitlichen Abtastung (vertikale Linien – z.B. 48.000/s) ergibt sich ein zeitdiskretes Signal

2

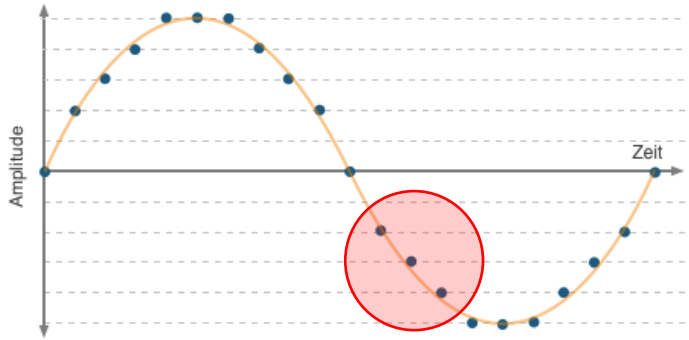


3 Mit 16 Bit stehen uns 65.536 Werte zur Verfügung (horizontale Linien)



Nach der Quantisierung werden die Werte den möglichen Quantisierungsstufen „zugeordnet“

4



Quelle: https://13db.de/wissen/digitalsignale/

1 Mit der Anzahl von n Bits (binary digit) kann man Z-Zustände erzeugen:

$$Z = 2^n$$

2

n	Z
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1.024
11	2.048
12	4.096
13	8.192
14	16.384
15	32.768
16	65.536

Beispiele:

Variable B	Variable A
0	0
0	1
1	0
1	1

Variable Z	Variable Y	Variable X
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

3 $a^n = a \cdot a \cdot a \ \ldots a$

4 $a^0 = 1$ für a ≠ 0

$127^0 = 1$

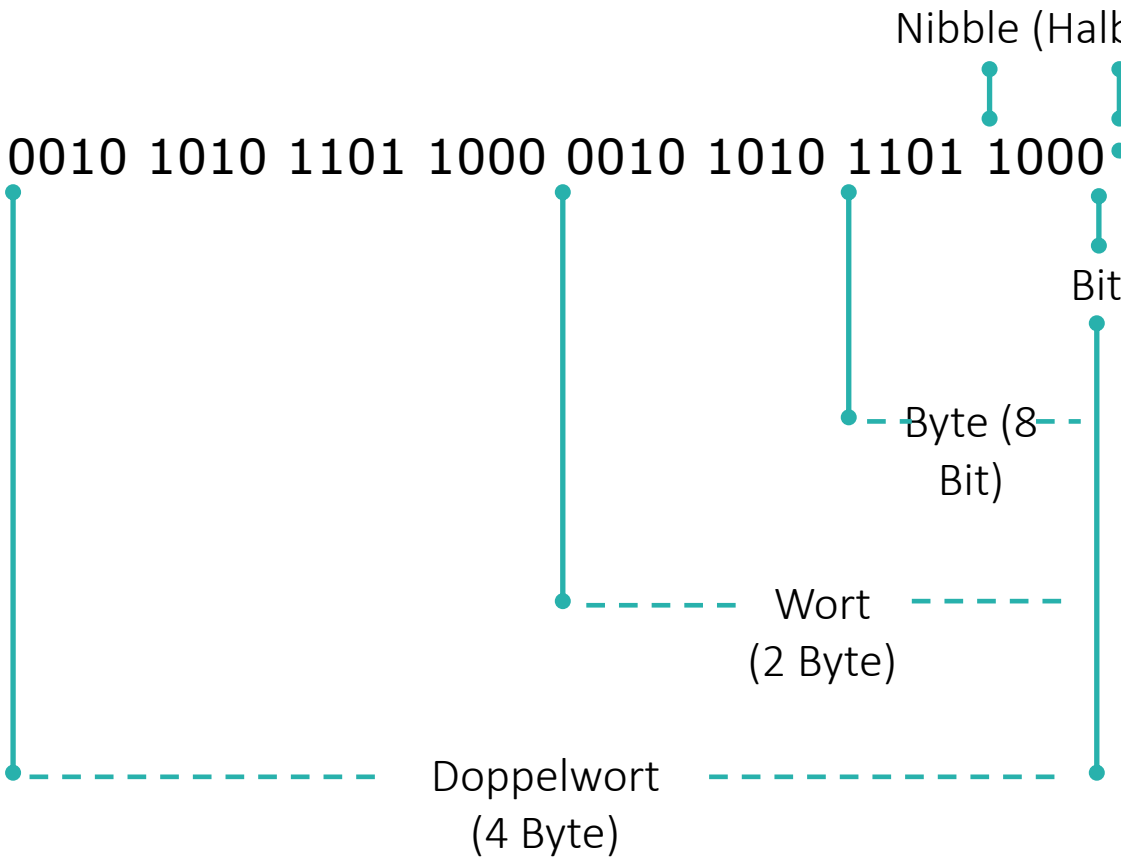
$74^0 = 1$

5 $1 \cdot 2^{-1} = \frac{1}{2^1} = \frac{1}{2} = 0{,}5$

$1 \cdot 2^{-2} = \frac{1}{2^2} = \frac{1}{4} = 0{,}25$

…

Digitale Daten
Bits und Bytes | Aufteilung Bit, Byte, Wort, Doppelwort



Das Byte ist eine Maßeinheit der Digitaltechnik und der Informatik, welches meist für eine Folge aus 8 Bit steht.

Historisch gesehen war ein Byte die Anzahl der Bits zur Kodierung eines einzelnen Schriftzeichens im jeweiligen Computersystem und daher das kleinste adressierbare Element in vielen Rechnerarchitekturen.

Digitale Daten
Bits und Bytes | Speichergrößen und Einheiten

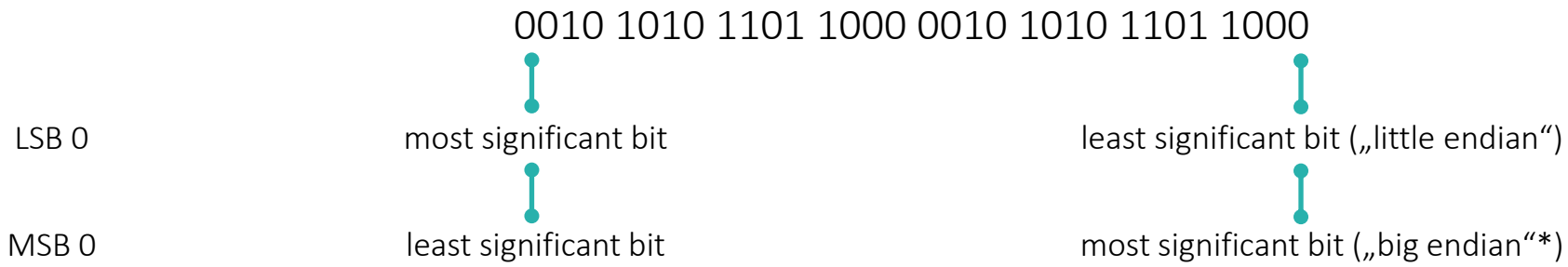
Für Datenspeicher mit binärer Adressierung ergeben sich technisch Speicherkapazitäten basierend auf Zweierpotenzen (2^n Byte).
Normalerweise werden jedoch dezimale Angaben zur Speichergröße verwendet (https://de.wikipedia.org/wiki/Byte#Vergleichstabelle)

Dezimalpräfixe			Unterschied gerundet	Binärpräfixe gemäß IEC		
Name	**Symbol**	**Anzahl Bytes**[G 1]		**Name**	**Symbol**	**Anzahl Bytes**
Kilobyte	kB[G 2]	$1\,000 = 10^3$	2,4 %	Kibibyte	KiB[G 3]	$1\,024 = 2^{10}$
Megabyte	MB	$1\,000\,000 = 10^6$	4,9 %	Mebibyte	MiB	$1\,048\,576 = 2^{20}$
Gigabyte	GB	$1\,000\,000\,000 = 10^9$	7,4 %	Gibibyte	GiB	$1\,073\,741\,824 = 2^{30}$
Terabyte	TB	$1\,000\,000\,000\,000 = 10^{12}$	10,0 %	Tebibyte	TiB	$1\,099\,511\,627\,776 = 2^{40}$
Petabyte	PB	$1\,000\,000\,000\,000\,000 = 10^{15}$	12,6 %	Pebibyte	PiB	$1\,125\,899\,906\,842\,624 = 2^{50}$
Exabyte	EB	$1\,000\,000\,000\,000\,000\,000 = 10^{18}$	15,3 %	Exbibyte	EiB	$1\,152\,921\,504\,606\,846\,976 = 2^{60}$
Zettabyte	ZB	$1\,000\,000\,000\,000\,000\,000\,000 = 10^{21}$	18,1 %	Zebibyte	ZiB	$1\,180\,591\,620\,717\,411\,303\,424 = 2^{70}$
Yottabyte	YB	$1\,000\,000\,000\,000\,000\,000\,000\,000 = 10^{24}$	20,9 %	Yobibyte	YiB	$1\,208\,925\,819\,614\,629\,174\,706\,176 = 2^{80}$

*Quelle: Wikipedia

Digitale Daten
Bits und Bytes | MSB und LSB

Die Bitwertigkeit legt den Stellenwert eines einzelnen Bits fest, den es durch seine Position innerhalb einer Binärzahl (auch Dualzahl genannt) haben soll. Wichtig ist diese Festlegung auch bei der seriellen Datenübertragung, sowie für parallele Busse in der Datenverarbeitung.



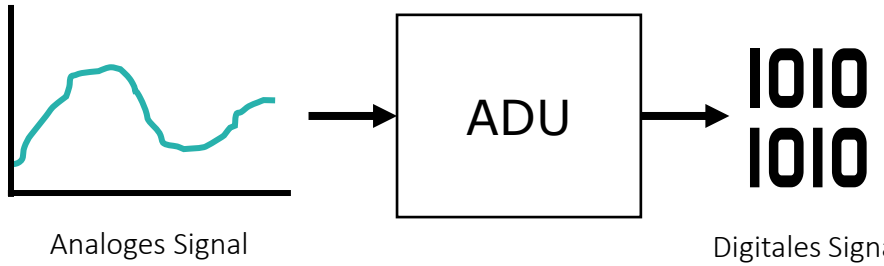
Beginnt die Zählung mit 0 für das niedrigstwertige Bit (LSB) auf der rechten Seite und läuft nach links, so spricht man von „LSB_0“.

Beginnt die Zählung mit 0 für das höchstwertige Bit (MSB) auf der linken Seite und läuft nach rechts, so spricht man von „MSB_0“.

*Das Big-Endian-Format wird von Mainframe-Prozessoren, einigen RISC-Architekturen und von Minicomputern benutzt. In TCP/ IP-Netzwerken erfolgt die Byte-Anordnung im Big-Endian-Format. Nach dem Big-Endian-Verfahren arbeiten u.a. der MC68000 von Motorola, Scalable Processor Architecture (SPARC), IBM PowerPC.

Quelle: https://de.wikipedia.org/wiki/Bitwertigkeit

Digitale Daten
Bits und Bytes | Anwendungsbeispiel ADU





- Wie groß ist die minimale Auflösung für eine Eingangsspannung von 1V?

- Wie groß wäre die minimale Auflösung für eine Eingangsspannung von 10V, wenn wir einen 16-Bit A/D-Wandler eingesetzt hätten?

Technische Daten

Allgemeines

Modell	A/D-Wandler
Typ	12-Bit
Auflösung	12-Bit
Ausführung	1-Kanal, SPI
Technologie	Low Power CMOS
Bauform	DIP-8

Sonstiges

Temperaturbereich	-40 ... +85 °C

Elektrische Werte

Frequenz	1,6 MHz
Versorgungsspannung	+2,7 ... +5,5 VDC
Eingangsstrom	100 µ A

Ausführung

Abtastrate	±1/2 LSB

Digitale Daten
Analog-Digital-Umsetzung

Beispiel:



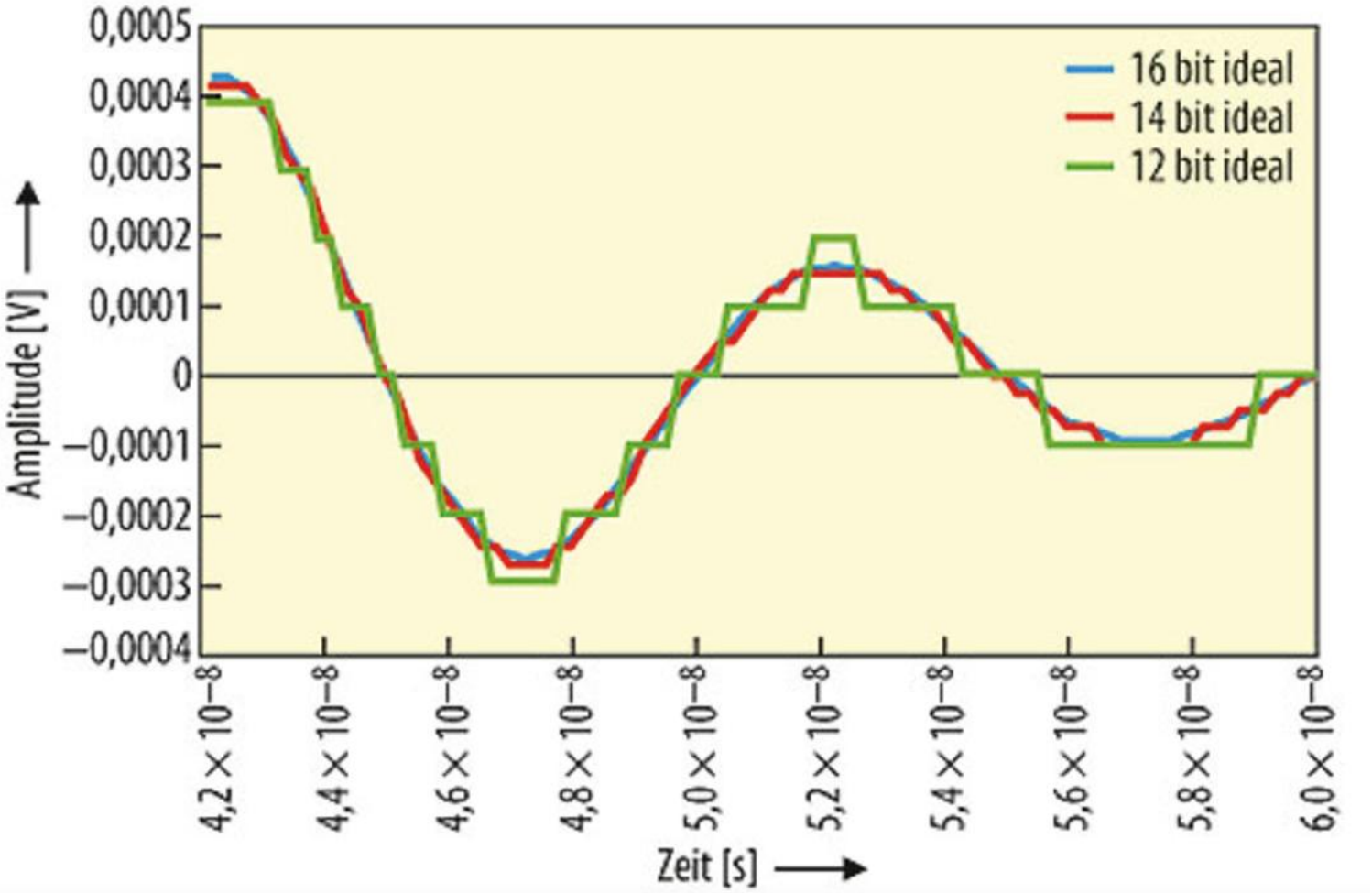
Wie groß ist der kleinste Spannungswert, der bei einem Eingangsspannungsbereich von ± 10V, dargestellt werden könnte?

https://www.beckhoff.com/de-de/produkte/i-o/ethercat-klemmen/el3xxx-analog-eingang/el3001.html

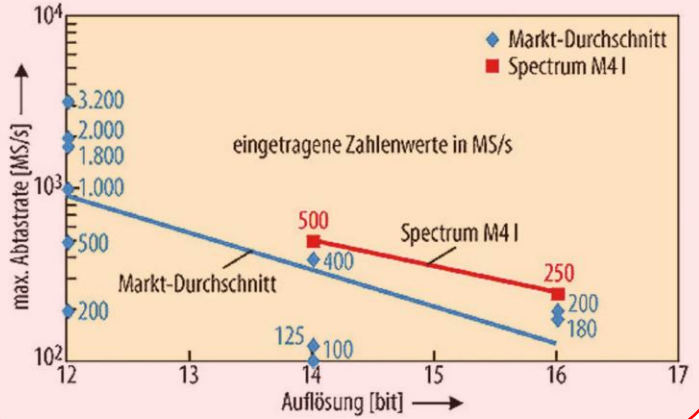
Digitale Daten
Bits und Bytes | Übung ADU

Auflösung / Bit	Minimaler Spannungsschritt für ±500 mV Eingangsbereich
8	?
10	?
12	?
14	?
16	?

Digitale Daten
Bits und Bytes | Anwendungsbeispiel ADU



Zu beachten ist, dass eine höhere Auflösung zu einer langsameren Abtastrate führt.



Quelle: https://www.elektroniknet.de/messen-testen/was-hohe-digitizer-aufloesung-bringt.113641.html

Digitale Daten
Zahlensysteme

Zahlensystem	Basis	Wertebereich (Ziffern)	Beispiel
Dezimal	10	{0,1, 2,..., 8, 9}	73_{10}
Dual	2	{0, 1}	1001001_{2}
Oktal	8	{0, 1, 2, 3, 4, 5, 6, 7}	273_{8}
Hexadezimal	16	{0,..., 9, A, B, C, D, E, F}	49_{16}

Ein Positionssystem mit der Basis B ist ein Zahlensystem, in dem eine Zahl in x Potenzen von B zerlegt wird.*

$$n = \sum_{i=0}^{N-1} b_i \times B^i$$

B=Basis des Zahlensystems ($\mathrm{B} \in N, B \geq 2$)
b=Ziffern $b_i \in N_0, 0 \leq b_i < B$)
N=Anzahl der Stellen

Beispiel	Berechnung
$n=2017_{(10)}$	$2 \cdot 10^3 + 0 \cdot 10^2 + 1 \cdot 10^1 + 7 \cdot 10^0$
$n=11001_{(2)}$	$1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$

*Quelle: Grundlagen der Informatik / Pearson Verlag

Beispiele:

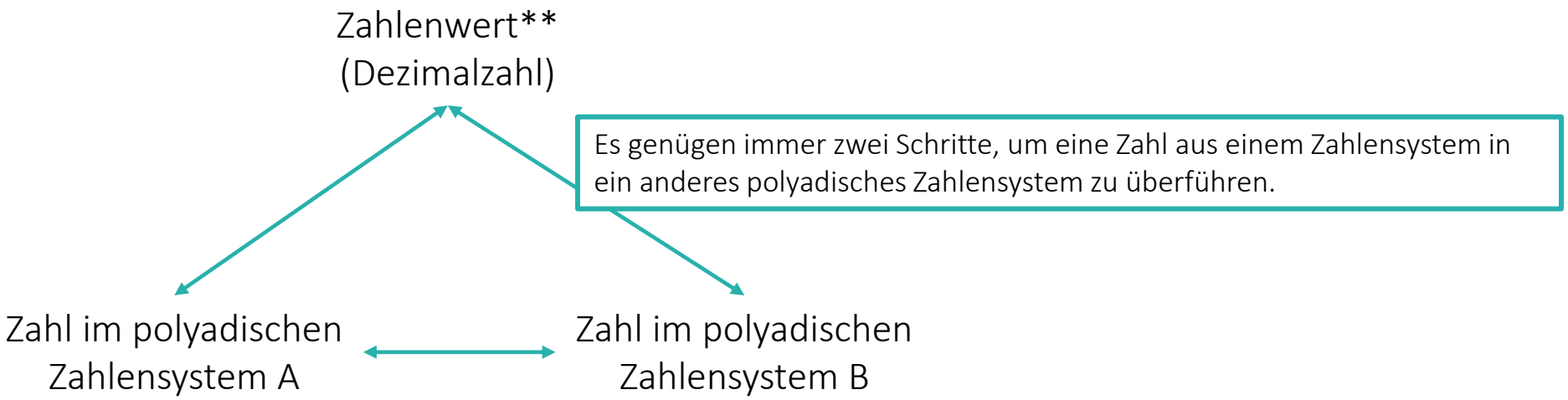
Der Zahlenwert ergibt sich als Summe der Produkte und Ziffer und dem Stellenwert!

$10011_{(2)}$	?	?
$201_{(3)}$	?	?
$23_{(8)}$	?	?
$19_{(10)}$	?	?
$13_{(16)}$	?	?

Digitale Daten
Konvertieren zwischen den Zahlensystemen

Ein Stellenwertsystem, Positionssystem oder polyadisches Zahlensystem ist ein Zahlensystem, bei dem die (additive) Wertigkeit eines Symbols von seiner Position, der Stelle, abhängt.

Beispielsweise besitzen im weitverbreiteten Zehnersystem für den Beispiels-Wert „127“ die Ziffer „1“ den Wert 1 · 100, dazu addiert sich für die Ziffer „2“ der Wert 2 · 10 sowie für die „7“ 7 · 1 – die Symbole „1“, „2“ und „7“ besitzen eine Wertigkeit, die davon abhängt, an welcher Position/Stelle sie in der Zahl stehen.*

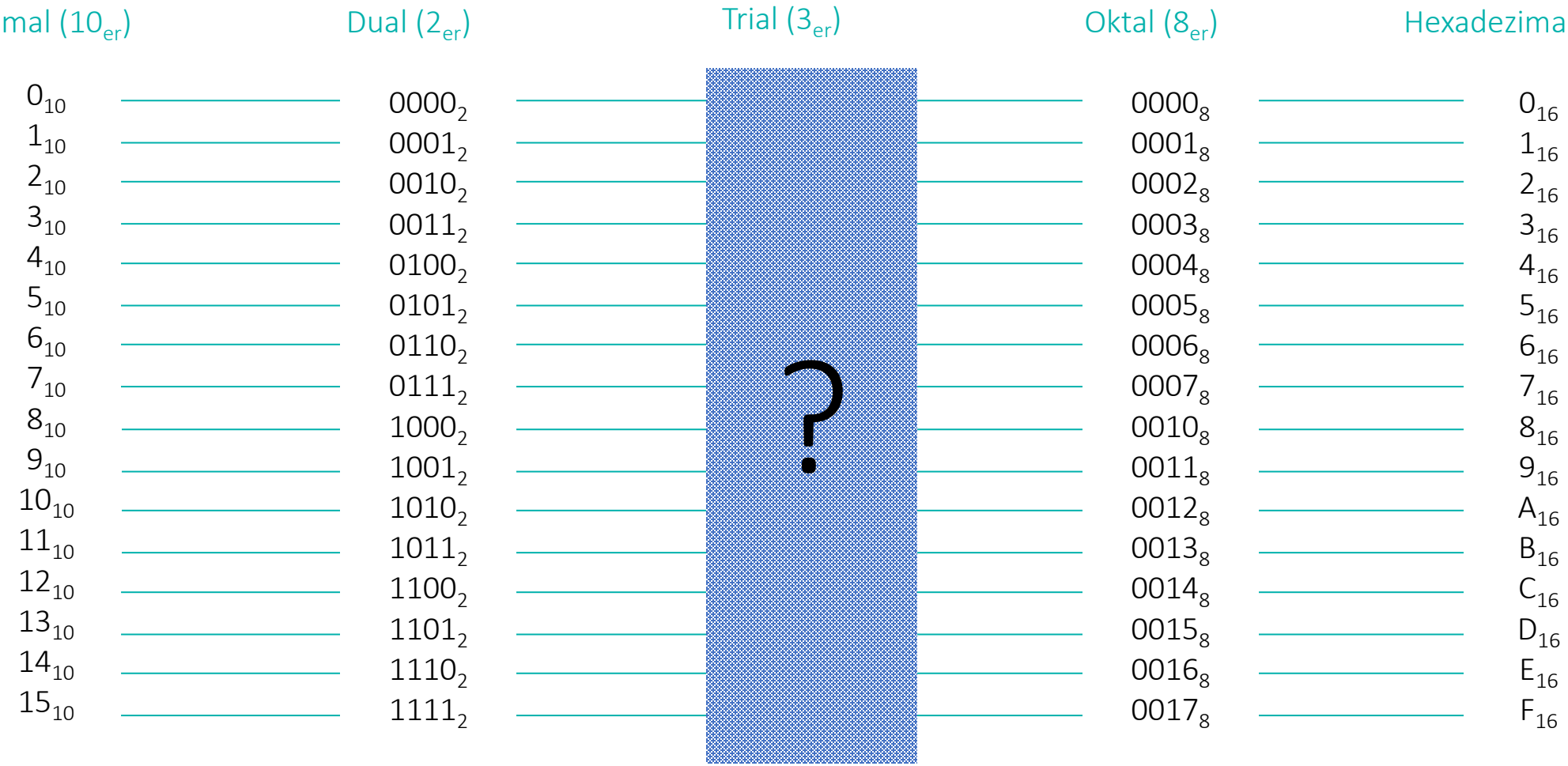


*Quelle: https://de.wikipedia.org/wiki/Stellenwertsystem
**Quelle: Handbuch der Elektronik / Digitaltechnik

Digitale Daten
Dual-, Oktal- und Hexadezimalsystem

Dezimal (10_{er})	Dual (2_{er})	Oktal (8_{er})	Hexadezimal (16_{er})
0_{10}	0000_2	0000_8	0_{16}
1_{10}	0001_2	0001_8	1_{16}
2_{10}	0010_2	0002_8	2_{16}
3_{10}	0011_2	0003_8	3_{16}
4_{10}	0100_2	0004_8	4_{16}
5_{10}	0101_2	0005_8	5_{16}
6_{10}	0110_2	0006_8	6_{16}
7_{10}	0111_2	0007_8	7_{16}
8_{10}	1000_2	0010_8	8_{16}
9_{10}	1001_2	0011_8	9_{16}
10_{10}	1010_2	0012_8	A_{16}
11_{10}	1011_2	0013_8	B_{16}
12_{10}	1100_2	0014_8	C_{16}
13_{10}	1101_2	0015_8	D_{16}
14_{10}	1110_2	0016_8	E_{16}
15_{10}	1111_2	0017_8	F_{16}

Digitale Daten
Dezimal, Dual-, Trial-, Oktal- und Hexadezimalsystem



Stellen Sie die folgenden Zahlen in ihrer Summenschreibweise dar und geben Sie ihre entsprechenden Werte im Dezimalsystem an:

$(312)_4 = ?$

$(1202)_{16} = ?$

$(ab1)_{12} = ?$

$(101011)_2 = ?$

$(705)_8 = ?$

$(ABC)_{16} = ?$

$(1111)_2 = ?$

$(127)_8 = ?$

*Quelle: Grundlagen der Informatik / Pearson Verlag

Variante 1: Subtraktion von höchsten 2er-Potenzen:

$$
\begin{aligned}
73 - 2^6 = 9 &\rightarrow 1 \\
9 - 2^5 < 0 &\rightarrow 0 \\
9 - 2^4 < 0 &\rightarrow 0 \\
9 - 2^3 = 1 &\rightarrow 1 \\
1 - 2^2 < 0 &\rightarrow 0 \\
1 - 2^1 < 0 &\rightarrow 0 \\
1 - 2^0 = 0 &\rightarrow 1
\end{aligned}
$$

Das Ergebnis wird von oben nach unten abgelesen

→ (73_{10}) = 1001001_2

Digitale Daten
Konvertieren zwischen den Zahlensystemen | Dezimal nach Dual

Variante 2: Horner-Schema oder Division mit Rest:

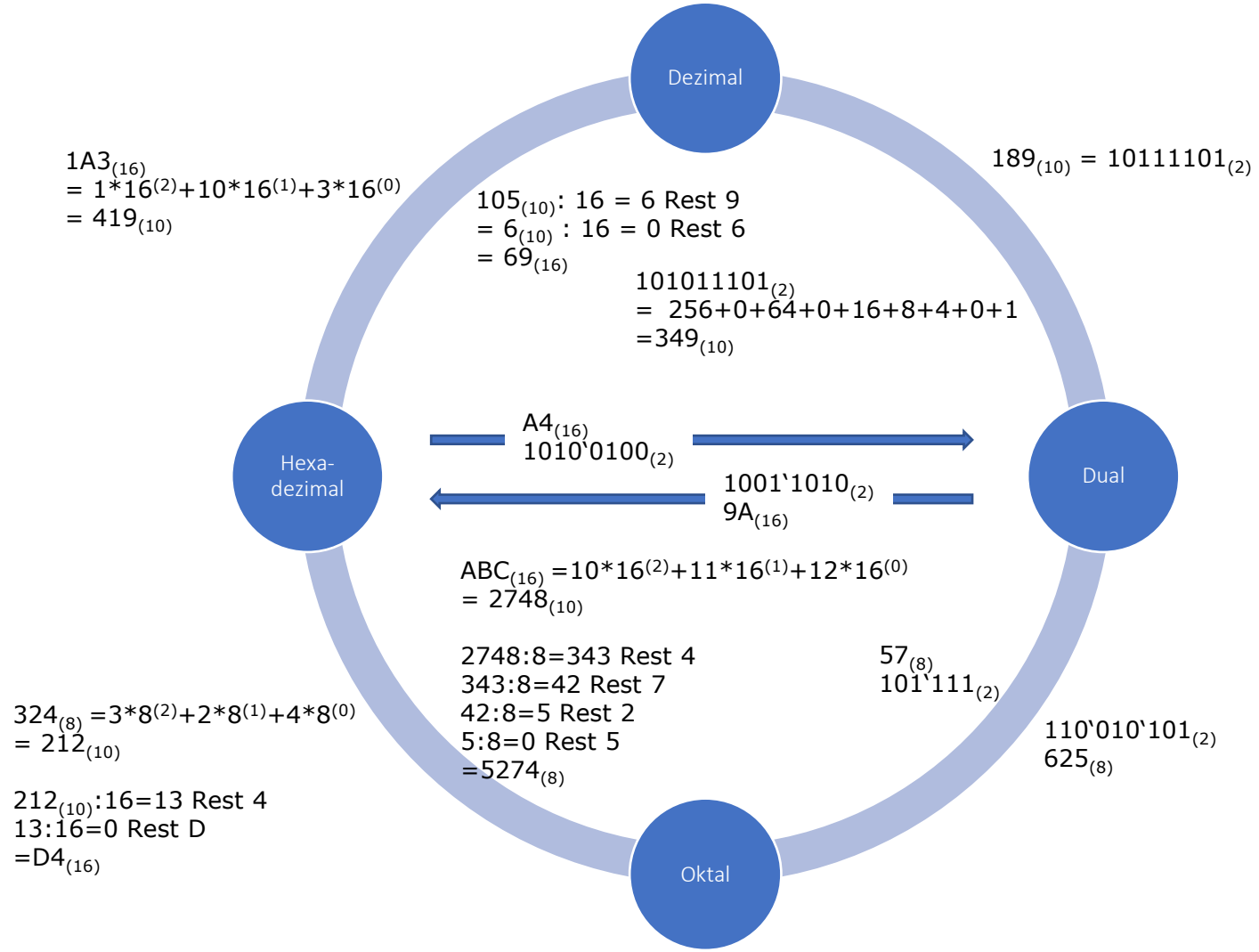
Rechnung	Ergebnis	Rest
73:2	36	1
36:2	18	0
18:2	9	0
9:2	4	1
4:2	2	0
2:2	1	0
1:2	0	1



Das Ergebnis wird von unten nach oben abgelesen!

→ (73_{10}) = 1001001_2

Digitale Daten
Konvertierung zwischen den Zahlensystemen



Digitale Daten
Konvertieren zwischen den Zahlensystemen

Binär	Dezimal	Hexadezimal
0010 1010	?	?
?	37	?
1010 1100	?	?

Digitale Daten
Rechenoperationen / Addition

Addition:
Die Addition der Zahler erfolgt ziffernweise von rechts nach links. Erreicht oder übersteigt die Summe der Ziffern den Wert der Basis so entsteht ein Übertrag auf die nächste Stelle. In der gerechneten Stelle wird dabei nur die Ziffer für den Zahlenwert geschrieben, um den der Wert der Basis überschritten wurde.

dezimal	dual	oktal	hexadezimal
...8 +..$_{1}$9 ...7	...1 +..$_{1}$1 ...0	...7 +..$_{1}$6 ...5	...8 +..$_{1}$A ...2
8 + 9 =17 =10+7 =1Ü+7	1+1 =2 =2 + 0 =1Ü+0	7+6 =13 =>13:8=1R5 =1Ü+5	8+A =18 =16+2 =1Ü+2

Für die duale Addition gilt:
0+0=0
0+1=1
1+0=1
1+1=0 Übertrag 1
1+1+1(vom Übertrag)=1 Übertrag 1

Beispiel

dezimal
$(45)_{10}$ = 0101101 $(54)_{10}$ = 0110110 1111---- $(99)_{10}$ = 1100011

Digitale Daten
Darstellung negativer Zahlen

Negative Zahlen werden u.a. durch Bildung des

2-er Kompliments gebildet

Feste Anzahl von Bits (hier z.B. 8 Bit)

01001001 positive Zahl 73_{10}

1. Schritt Bits invertieren (aus einer 0 wird eine 1, aus einer 1 wird eine 0)

10110110

2. Schritt „1“ addieren

+1

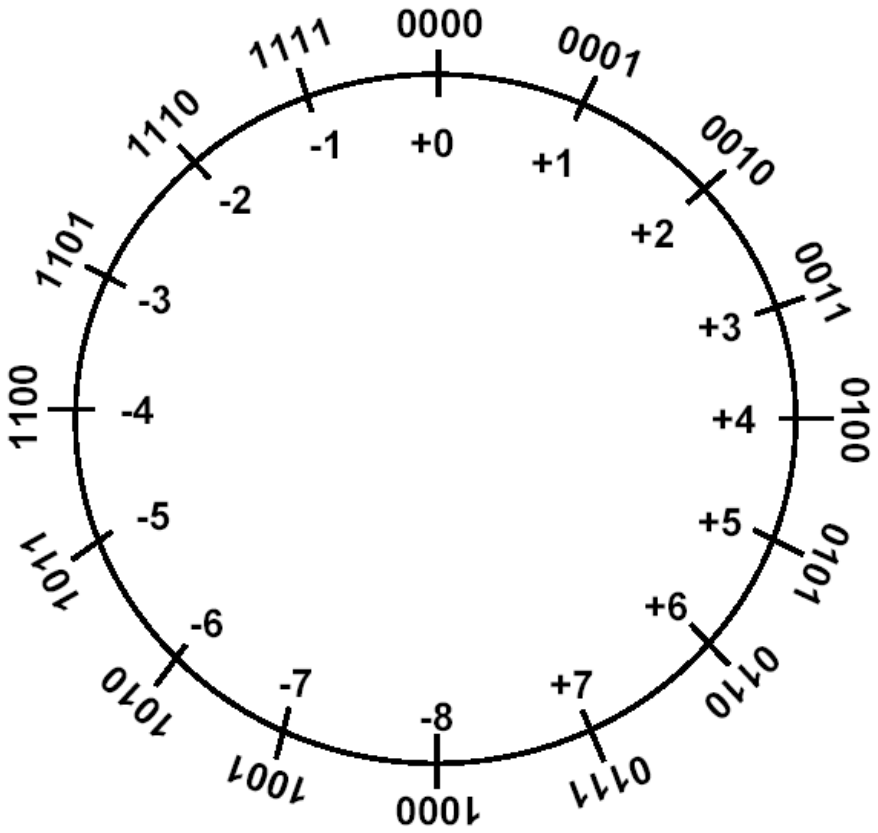
Erstes Bit 1: Kennzeichen für eine negative Zahl

10110111 negative Zahl -73

Der Überlauf wird ignoriert, falls er nach 2. Schritt vorhanden ist

Digitale Daten
Darstellung negativer Zahlen

Das 2er-Komplement kann wie folgt grafisch dargestellt werden (hier für 4 Bits). Bei 4 Bits ergibt sich ein Wertebereich von -8 bis +7.



Beispiel

$(2)_{10} = 0010$

?

$(7)_{10} = 0111$

?

Digitale Daten
Rechenoperationen / Subtraktion

Subtraktion:
In digitalen Rechenwerken wird die Subtraktion häufig auf eine Addition zurückgeführt. Dazu wird die zu subtrahierende Zahl durch ihr Zweierkomplement ersetzt und anschließend addiert. Dadurch können Addition und Subtraktion mit demselben Addierwerk ausgeführt werden, sodass kein separates Subtraktionswerk erforderlich ist.

Beispiel (4 Bits)

1 $(4)_{10} - (5)_{10}$

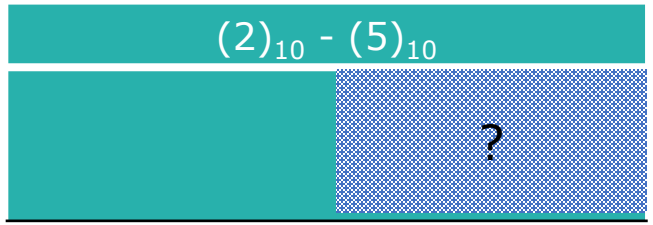
$(4)_{10} = 0100$

2 $-(5)_{10}$

$(5)_{10} = 0101$
1010
$+1$
1011

3 $(4)_{10} + (-5)_{10}$

$(4)_{10} = 0100$
$-(5)_{10} = 1011$
1111



Merksatz:
Die Subtraktion $A - B$ wird durch die Addition $A + (-B)$realisiert, wobei $-B$ im Binärsystem durch das Zweierkomplement von B dargestellt wird.

1 Mit der Anzahl von n Bits (binary digit) kann man Z-Zustände erzeugen:

$$Z = 2^n$$

3 $a^n = a * a * a \ \ldots a$

4 $a^0 = 1$ für a ≠ 0

$127^0 = 1$

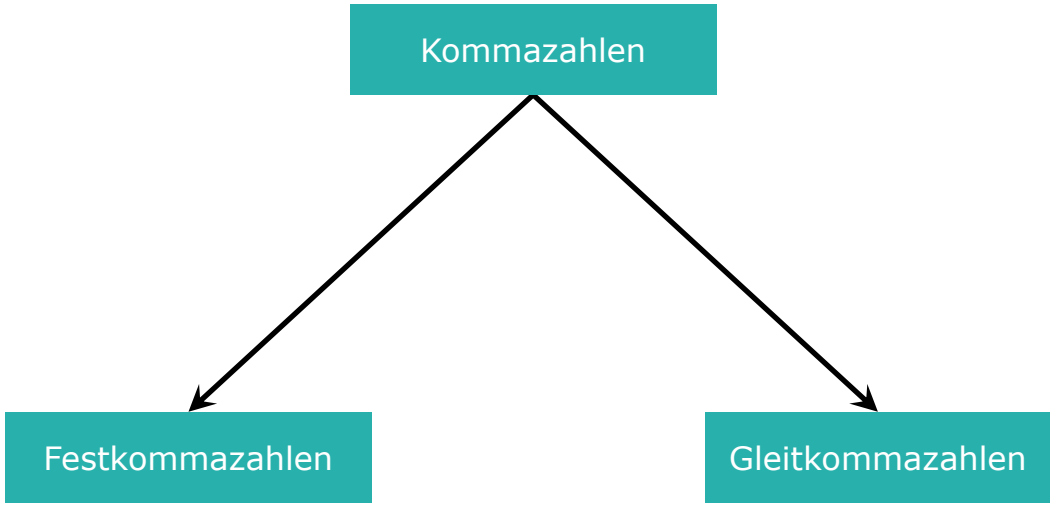
$74^0 = 1$

2

n	Z
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1.024
11	2.048
12	4.096
13	8.192
14	16.384
15	32.768
16	65.536

5 $a^{-n} = \frac{1}{a^n}$

Digitale Daten
Darstellung von Kommazahlen



Darstellung von irrationalen Zahlen?

3,1415926
2,7182818
1,732
1,414
…

Ein anderes Problem:

1 / 3 = $0,\overline{3}$

0,3333333333 * 3 = 0,999999999

Bei einer Festkommazahl ist die Position des Kommas **fest vorgegeben**. Es wird also vorher festgelegt, wie viele Bits für den ganzzahligen und wie viele für den Nachkommateil verwendet werden.

$$0101{,}1000_2 = 5{,}5_{10}$$

Bei einer Gleitkommazahl kann das Komma gedanklich verschoben werden. Die Zahl wird ähnlich wie bei der wissenschaftlichen Schreibweise dargestellt.

$$12345 = 1{,}2345 \cdot 10^4$$

$$1011{,}01_2 = 1{,}01101_2 \cdot 2^3$$

Digitale Daten
Darstellung von Kommazahlen | Festkommazahl

Bei Festpunktzahlen steht der Punkt (das Komma) immer an einer bestimmten festgelegten Stelle.

2	1	0		-1	-2	-3	-4
1	0	0	.	1	0	0	1

$$1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 0 \cdot 2^{-3} + 1 \cdot 2^{-4}$$
$$= 4{,}5625_{(10)}$$

Vorgehensweise:

1. Umwandlung des ganzzahligen Anteils
2. Umwandlung der Nachkommastellen
3. Addition der beiden Ergebnisse

Digitale Daten
Darstellung von Kommazahlen

Beispiel: Umwandelung von $73{,}4_{(10)}$

$73_{(10)}=1001001_{(2)}$ (siehe Folie 22)

Für $0{,}4_{(10)}$ Anwendung der Multiplikationsmethode mit Rest:

$0{,}4 \cdot 2 = 0{,}8 \longrightarrow 0$ (erstes Bit nach dem Komma)
$0{,}8 \cdot 2 = 1{,}6 \longrightarrow 1$
$0{,}6 \cdot 2 = 1{,}2 \longrightarrow 1$
$0{,}2 \cdot 2 = 0{,}4 \longrightarrow 0$
$0{,}4 \cdot 2 = 0{,}8 \longrightarrow 0$

Das Ergebnis wird von oben nach unten abgelesen

Wenn die Periode gefunden wurde, erfolgt der Abbruch:
Addition von $73_{(10)}$ und $0{,}4_{(10)}$

Ergebnis: $1001001{,}0110_{(2)}$

Digitale Daten
Darstellung von Kommazahlen | Festkommazahl

9,87654321 (große Genauigkeit)
987654321 (großer Wertebereich)
98765,4321 (Mischung?)

Die Nachteile:

- Es ist nur möglich einen beschränkten Wertebereich abzudecken.
- Die Stelle des Kommas muss festgelegt werden, was es schwierig macht, wenn man mit unterschiedlichen Wertebereichen (sehr groß / sehr klein)* arbeiten möchte.
- Die Ergebnisse von arithmetischen Operationen sind durch Rundungsfehler eventuell nicht korrekt darzustellen. Diesen Rundungsfehler gilt es möglichst klein zu halten.
- Bei Berechnungen, bei denen dividiert und multipliziert wird, kann es zu erheblichen Ungenauigkeiten kommen.

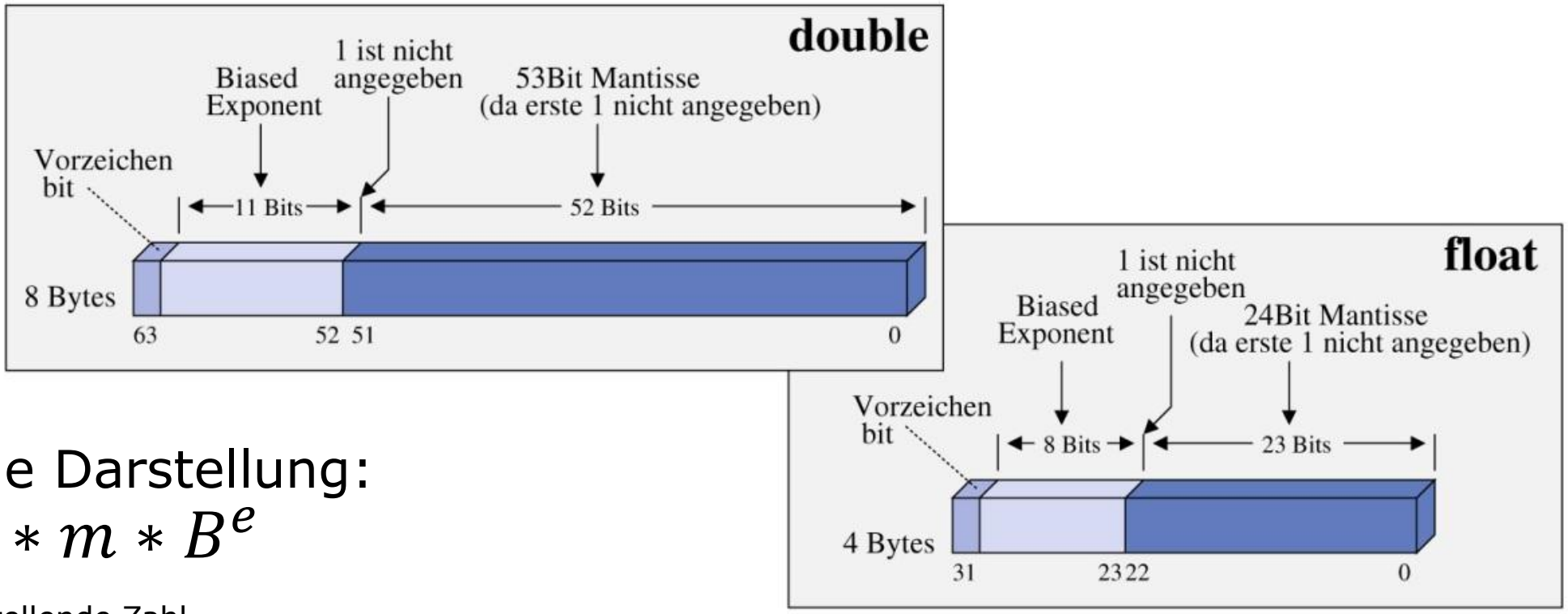
→ Wir brauchen die Möglichkeit große Zahlen mit einer großen Genauigkeit darstellen und verarbeiten zu können. Daher wird stattdessen die Gleitpunktdarstellung verwendet. Gleitkommazahlen können beides - großer Wertebereich verbunden mit einer großen Genauigkeit.

Die Lösung ist die Normalisierung und Exponentialdarstellung. Hier muss nur der Wert hinter dem Komma und der Exponent gespeichert und verarbeitet werden. Die Position des Kommas ergibt sich aus dem Exponenten.

Digitale Daten
Rechnen mit Gleitkommazahlen

Die Ergebnisse von arithmetischen Operationen sind durch Rundungsfehler eventuell nicht korrekt darzustellen. Diesen Rundungsfehler gilt es möglichst klein zu halten.

Bei Berechnungen, bei denen dividiert und multipliziert wird, kann es zu erheblichen Ungenauigkeiten kommen.



Formale Darstellung:

$$x = v * m * B^e$$

X= Darzustellende Zahl
V= Vorzeichen
B= Basis (meistens 2)
e = Exponent
m = Mantisse

Quelle: https://www.elektronik-kompendium.de/sites/dig/1807231.htm

Digitale Daten
Darstellung von Kommazahlen | Gleitpunktzahlen

Die Mantisse bestimmt die Genauigkeit. Je größer die Mantisse, desto genauer ist die Zahl hinter dem Komma. Der Exponent bestimmt die Größe des Wertebereichs der Zahl.

Wertebereich von Gleitkommazahlen:

Größe	Mantisse	Exponent	minimaler Exponent	maximaler Exponent	signifikate Dezimalstellen	kleinste normalisierte Zahl	größte Zahl	Wertebereich	Seicherbedarf	Datentyp
Binärformate										
16 Bit	10 Bit	5 Bit	-14	15	3 bis 4	$+-6{,}104 \times 10^{-5}$	+-65504		16 Bit (2 Byte)	
32 Bit	23 Bit	8 Bit	-126	127	7 bis 8	$+-1{,}175 \times 10^{-38}$	$+-3{,}403 \times 10^{38}$	von - 3,40282347 x 10^{38} bis 3,40282347 x 10^{38}	32 Bit (4 Byte)	Float (Single Precision)
64 Bit	52 Bit	11 Bit	-1022	1023	15 bis 16	$+-2{,}225 \times 10^{-308}$	$+-1{,}798 \times 10^{308}$	von - 1.79769313486231570 x 10^{308} bis 1.79769313486231570 x 10^{308}	64 Bit (8 Byte)	Double (Double Precision)
128 Bit	112 Bit	15 Bit	-16382	16383	33 bis 34	$+-3{,}362 \times 10^{-4932}$	$+-1{,}190 \times 10^{4932}$		128 Bit (16 Byte)	

Bias = 127

Bias = 1023

Quelle: https://www.elektronik-kompendium.de/sites/dig/1807231.htm

Digitale Daten
Darstellung von Kommazahlen | Gleitpunktzahlen nach IEEE*

Prinzipielle Vorgehensweise:

1. Umwandlung der Dezimalzahl in eine binäre Festkommazahl ohne Vorzeichen
    a. Vorkommazahl in Dual (Division :2, von unten nach oben)
    b. Nachkommazahl in Dual (Multiplikation · 2, von oben nach unten)
2. Normalisieren und Bestimmen des Exponenten
3. Charakteristik = Exponent + Bias (127 bei float, 1023 bei double)
    a. Umwandlung in Dual
4. Vorzeichen-Bit bestimmen
    a. + = 0
    b. - = 1
5. Gleitkommazahl zusammensetzen
    a. Vorzeichen-Bit + Exponent + Mantisse

*Institute of Electrical and Electronics Engineers

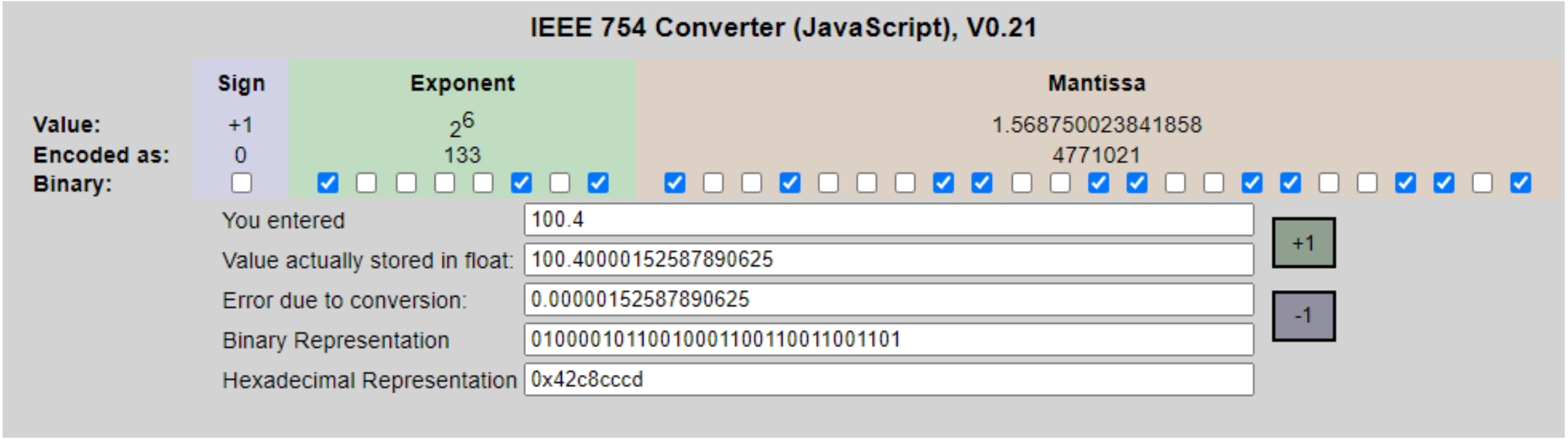
Digitale Daten
Darstellung von Kommazahlen | Gleitpunktzahlen nach IEEE*

Beispiel: 17,625

Bei der Normalisierung geht es darum, dass man nur die Nachkommastellen speichern möchte. Bei der binären Darstellung von Zahlen steht vorne immer eine Eins (1). Diese Vorkomma-Eins kann man beim Speichern bzw. bei der Darstellung weglassen (hidden bit), weil hier immer eine Eins steht. Dafür hat man hinten eine Stelle mehr für die Genauigkeit.

Digitale Daten
Darstellung von Kommazahlen | Gleitpunktzahlen nach IEEE*

Digitale Daten
Darstellung von Kommazahlen | Gleitpunktzahlen nach IEEE*



https://www.h-schmidt.net/FloatConverter/IEEE754de.html

CODES



Digitale Daten
Codierung

Unter Codierung ist die Umwandlung einer Nachricht von einer Form in eine andere zu verstehen.

- Ein Code bildet die Zeichen eines Zeichenvorrates auf die Zeichen eines zweiten Zeichen-vorrates ab.

Zeichenvorrat 1: 0…5…9 → Zeichenvorrat 2: 0…0101…1001

- Sinnvollerweise soll auch eine Decodierung möglich sein, bei der aus dem codierten Zeichen wieder das ursprüngliche gewonnen wird.

Zeichenvorrat 1: 0…5…9 ← Zeichenvorrat 2: 0…0101…1001

Für jede Anwendung gibt geeignete Codes. So ist für die Zahlenarithmetik in einem Rechner ein anderer Code sinnvoll als für die Übertragung von Zahlen über eine Nachrichtenverbindung.

Digitale Daten
Codierung

Übungsaufgabe:

Wo werden Codes in der Informatik und Elektrotechnik eingesetzt?

Abkürzung	Vollständiger Name
ECC	Error-Correcting Code
JPEG	Joint Photographic Experts Group
MPEG	Moving Picture Experts Group
CRC	Cyclic Redundancy Check
QR	Quick Response
SSI	Synchronous Serial Interface

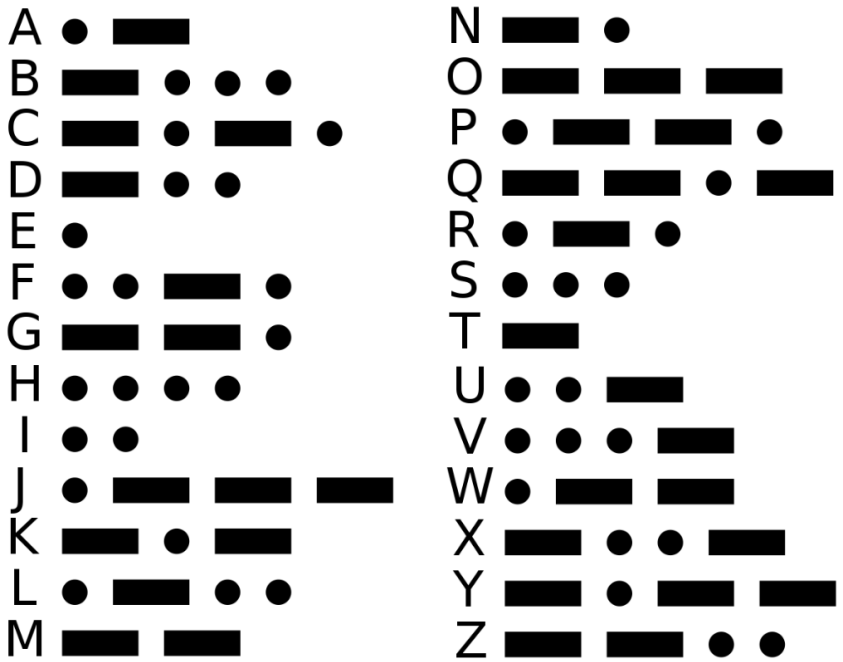
Bereich	Beispiel für Code	Wofür?
Zahlendarstellung	Binär-, BCD-, Gray-Code	Zahlen in digitalen Systemen darstellen
Zeichenkodierung	ASCII, Unicode/UTF-8	Buchstaben und Sonderzeichen als Bits darstellen
Fehlererkennung	Paritätsbit, CRC	Übertragungsfehler erkennen
Fehlerkorrektur	Hamming-Code, Reed-Solomon	Fehler erkennen und korrigieren
Datenübertragung	Manchester-Code, 8b/10b	Bits für die physikalische Übertragung codieren
Sensorik/Automatisierung	Gray-Code, SSI	Positionen bzw. Messwerte übertragen
Kommunikationsnetze	Ethernet-Codierungen, Modulation	Daten über Kabel/Funk übertragen
Speicher	ECC	Fehler in RAM/Datenspeichern erkennen/korrigieren
Prozessoren	Maschinencode/Instruktionscodes	Befehle für die CPU darstellen
Bild/Audio/Video	JPEG, MPEG, …	Daten komprimieren bzw. codieren
Identifikation	QR-Code, Barcode	Informationen maschinenlesbar speichern
Steuerungstechnik	Bitmuster, Status-/Fehlercodes	Zustände und Steuerinformationen übertragen

Digitale Daten
Anforderungen Codierung

- Übertragbarkeit
  - Fälschungssicherheit
  - Optimale Nutzung der technischen Randbedingungen des Übertragungskanals
- Speicherbarkeit
  - Minimierung des Speicherplatzes
  - Optimale Nutzung der technischen Randbedingungen und Eigenschaften des Speichermediums
- Verarbeitbarkeit
  - Einfache und schnelle Durchführung von Operationen
- Lesbarkeit
  - Gute Lesbarkeit für Menschen oder Maschinen

Digitale Daten
Codierung

Ein bekanntes Beispiel für einen Code ist der Morse-Code. Die Definition eines Codes geschieht durch eine Zuordnungstabelle. Dieser Code ist umkehrbar, da aus dem Buchstaben ein Morse-zeichen, und daraus wieder der Buchstabe eindeutig ermittelt werden kann.



Quelle: https://upload.wikimedia.org/wikipedia

Digitale Daten
ASCII Code

Der ASCII-Code (ASCII=American Standard for Coded Information Interchange) ist eine festgelegte Abbildungsvorschrift zur binären Codierung von Zeichen.

Die Kodierung erfolgt in einem Byte (8 Bits), so dass mit dem ASCII-Code 256 verschiedene Zeichen dargestellt werden können. Genutzt werden im Standard ASCII-Code jedoch nur 7 Bits, sodass nur 128 Zeichen (0-127) dargestellt werden können.

Test (z.B. in Word):

Alt +77 → M
Alt + 3 → ♥

dez							
0 NUL 0	16 DLE 20	32 SP 40	48 0 60	64 @ 100	80 P 120	96 ` 140	112 p 160
1 SOH 1	17 DC1 21	33 ! 41	49 1 61	65 A 101	81 Q 121	97 a 141	113 q 161
2 STX 2	18 DC2 22	34 " 42	50 2 62	66 B 102	82 R 122	98 b 142	114 r 162
3 ETX 3	19 DC3 23	35 # 43	51 3 63	67 C 103	83 S 123	99 c 143	115 s 163
4 EOT 4	20 DC4 24	36 $ 44	52 4 64	68 D 104	84 T 124	100 d 144	116 t 164
5 ENQ 5	21 NAK 25	37 % 45	53 5 65	69 E 105	85 U 125	101 e 145	117 u 165
6 ACK 6	22 SYN 26	38 & 46	54 6 66	70 F 106	86 V 126	102 f 146	118 v 166
7 BEL 7	23 ETB 27	39 ' 47	55 7 67	71 G 107	87 W 127	103 g 147	119 w 167
8 BS 10	24 CAN 30	40 (50	56 8 70	72 H 110	88 X 130	104 h 150	120 x 170
9 HT 11	25 EM 31	41) 51	57 9 71	73 I 111	89 Y 131	105 i 151	121 y 171
10 LF 12	26 SUB 32	42 * 52	58 : 72	74 J 112	90 Z 132	106 j 152	122 z 172
11 VT 13	27 ESC 33	43 + 53	59 ; 73	75 K 113	91 [133	107 k 153	123 { 173
12 FF 14	28 FS 34	44 , 54	60 < 74	76 L 114	92 \ 134	108 l 154	124 | 174
13 CR 15	29 GS 35	45 - 55	61 = 75	77 M 115	93] 135	109 m 155	125 } 175
14 SO 16	30 RS 36	46 . 56	62 > 76	78 N 116	94 ^ 136	110 n 156	126 ~ 176
15 SI 17	31 US 37	47 / 57	63 ? 77	79 O 117	95 _ 137	111 o 157	127 DEL 177

Legende:

dez
Zeichen
hex okt

Digitale Daten
ASCII Code

Der ASCII-Code wird auf 8 Bit (1 Byte) erweitert, sodass nun 256 Zeichen dargestellt werden können.

Beispiel:

- Westeuropäische und amerikanische Sprachen Latin 1, ISO-8859-1
- Slawische Sprachen Latin 2, ISO-8859-2
- Türkisch, Latin 5, ISO-8859-9
- …

Code	…0	…1	…2	…3	…4	…5	…6	…7	…8	…9	…A	…B	…C	…D	…E	…F
0…	*nicht belegt*															
1…																
2…	*SP*	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3…	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4…	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5…	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6…	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7…	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
8…	*nicht belegt*															
9…																
A…	*NBSP*	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	*SHY*	®	¯
B…	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C…	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D…	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E…	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F…	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Quelle: https://de.wikipedia.org/wiki/ISO_8859-1

Wir brauchen aber eine Möglichkeit mit dem wir sämtliche - („alle“) Zeichen darstellen können - einen weltweiten Standard.

Digitale Daten
Unicode

Der Unicode (https://home.unicode.org/) ist ein Zeichensatz mit 32 Bits, der als weltweiter Standard geplant ist.

"The Unicode Standard provides a unique number for every character, no matter what platform, device, application or language." (Quelle: https://www.unicode.org/standard/WhatIsUnicode.html)

Mit Hilfe der zur Verfügung stehenden 32 Bits sind 4.294.967.296 Zeichen darstellbar. Das gesamte System ist dabei in Zeichenbereiche aufgeteilt.

Die Speicherung und Übertragung von Unicode erfolgt in unterschiedlichen Formaten – Unicode Transformation Format (UTF). Am häufigsten wird der UTF-8 Standard (je Zeichen 1-4 Byte) eingesetzt, da er eine variable Länge sowie einen relativ niedrigen Speicherbedarf bietet.

Notation im Hexadezimalsystem U+(XX)XX XX XX

- Dabei geben die letzten 2 Byte eine Position in einer Plane (Ebene) an
- Die Angabe der Plane ist dabei optional (Standard = 00)

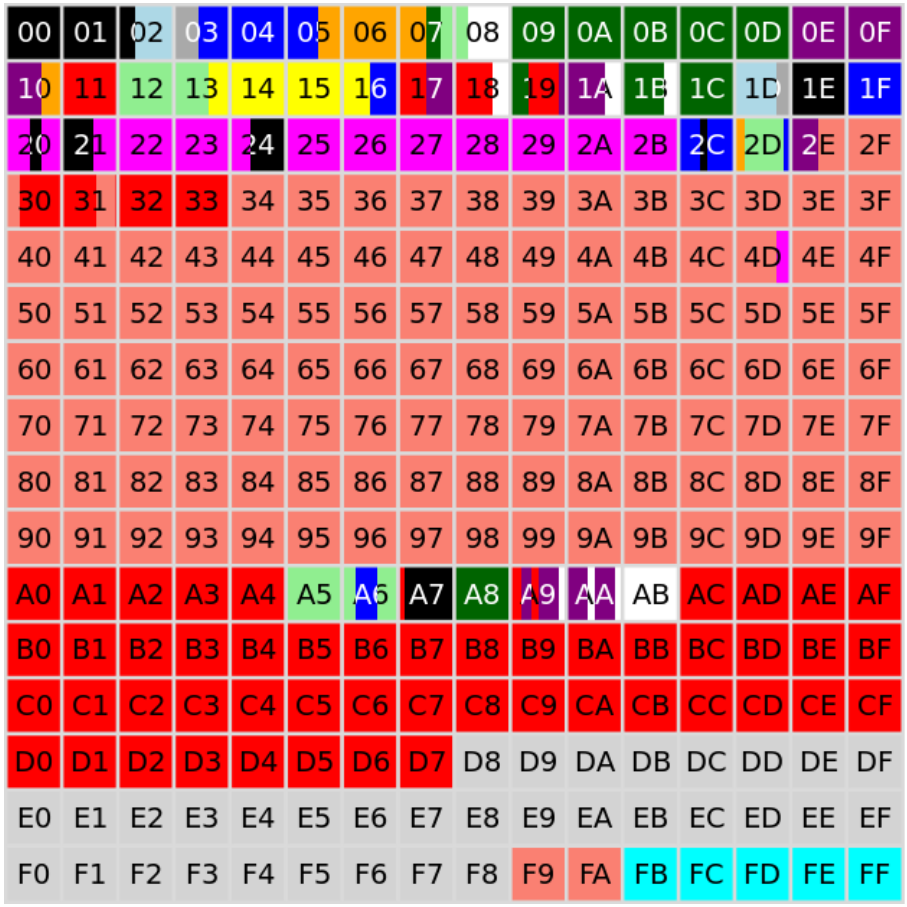
Digitale Daten
Unicode

UTF-8 ist dazu geeignet, den Speicherplatz eines Zeichens den Anforderungen anzupassen.

Handelt es sich bei einem Zeichen um ein ASCII-Zeichen wird ein Byte zur Codierung benötigt

- (0 – 127) → 0xxx xxxx
- Handelt es sich um ein anderes Zeichen werden 2 bis 4 Bytes zur Codierung benötigt
- Das erste Byte beginnt mit einer Anzahl von Einsen, die der Anzahl von zu verwendenden Bytes entspricht, gefolgt von einer 0
- Jedes folgende Byte beginnt mit 10

Digitale Daten
Unicode



Die *Basic Multilingual Plane* (BMP; Deutsch *Mehrsprachige Basis-Ebene*, auch als *Plane 00* bezeichnet) enthält hauptsächlich Schriftsysteme, die aktuell in Gebrauch sind.

Quelle: https://de.wikipedia.org/wiki/Unicode

Eine gute Übersicht gibt folgende Seite:

https://www.compart.com/de/unicode/block

Digitale Daten
Unicode

Das erste Byte beginnt mit einer Anzahl von Einsen, die der Anzahl von zu verwendenden Bytes entspricht, gefolgt von einer 0 - jedes folgende Byte beginnt mit 10.

Unicode-Bereich	UTF-8-Kodierung	Möglichkeiten	Platz für
0000 0000 – 0000 007F	0xxx xxxx	128	7 Bits
0000 0080 – 0000 07FF	110x xxxx 10xx xxxx	2048	11 Bits
0000 0800 – 0000 FFFF	1110 xxxx 10xx xxxx 10xx xxxx	65.536	16 Bits
0001 0000 – 0010 FFFF	1111 0xxx 10xx xxxx 10xx xxxx 10xx xxxx	2.097.152	21 Bits

Digitale Daten
Unicode | Beispiel

Es soll der Unicode U+265E (♞) in UTF-8 dargestellt werden:

U+265E benötigt selbst 16 Bits, also müssen zur UTF-8 Codierung 3 Bytes verwendet werden.

- Grundgerüst für 3 Byte UTF-8 Codierung: 1110 xxxx | 10xx xxxx | 10xx xxxx
- Binäre Darstellung von U+265E: (2) 0010 (6) 0110 (5) 0101 € 1110
- Einsetzen in das Grundgerüst ergibt: 1110 0010 | 1001 1001 | 1001 1110

Digitale Daten
Unicode | Übung

Ermitteln sie bitte für folgende Zeichen sowohl den Unicode als auch den UTF-8 Code in Hex.

Zeichen / Dezimalwert	Unicode	UTF-8
A / 65	?	
ä / 228	?	
Ў / 1038	?	

Digitale Daten
Grundlagen Numerische Codes

- Ein binärer Code ist einschrittig, wenn sein Zeichenvorrat ein Codealphabet bildet und zwei benachbarte Codeworte sich nur in genau einem Bit unterscheiden.

- Bei einem stetigen Code ist die Zahl der Bits, in denen sich benachbarte Codeworte unterscheiden, konstant.

- Das Gewicht eines binären Codewortes ist die Anzahl der Einsen in dem Codewort.
  - Bei gleichgewichteten binären Codes ist die Anzahl der Einsen in allen Codeworten identisch.
  - Bei einem balancierten binären Code ist in allen Codeworten die Anzahlen der Einsen gleich der Anzahl der Nullen.
  - Ein binärer numerischer Code ist bewertbar, wenn jeder Stelle eine (positive) Stellenwertigkeit zugeordnet ist. Diese Stellenwertigkeit gilt üblicherweise für den Fall, dass an der Stelle eine 1 steht.

Digitale Daten
Tetradische BCD (binary coded decimals) Codes

Codes, welche mit 4 Zeichen auskommen. Man hat insgesamt 16 Tetraden.

Davon verwirft man 6 Tetraden (Pseudotetraden). Die 10 übrigen nennt man Nutztetraden. Man nennt solche Codes redundant (weitschweifig).

Beispiel:
$237_{(10)}$

Dual: 11101101
BCD: 0010/0011/0111

	Dualer BCD (8-4-2-1)	Aiken (2-4-2-1)	Stibitz (Exzess-3)	Gray
0000	0	0	-	0
0001	1	1	-	1
0010	2	2	-	3
0011	3	3	0	2
0100	4	4	1	7
0101	5	-	2	6
0110	6	-	3	4
0111	7	-	4	5
1000	8	-	5	-
1001	9	-	6	-
1010	-	-	7	-
1011	-	5	8	-
1100	-	6	9	8
1101	-	7	-	9
1110	-	8	-	-
1111	-	9	-	-

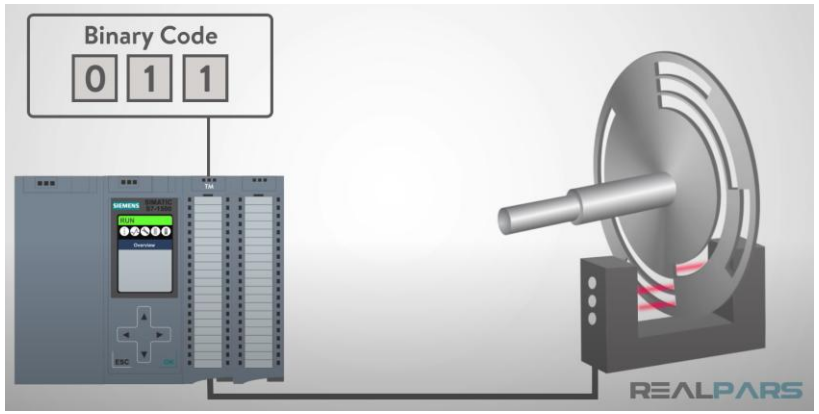
gewichtet | nicht gewichtet

Digitale Daten
3XS Gray Code

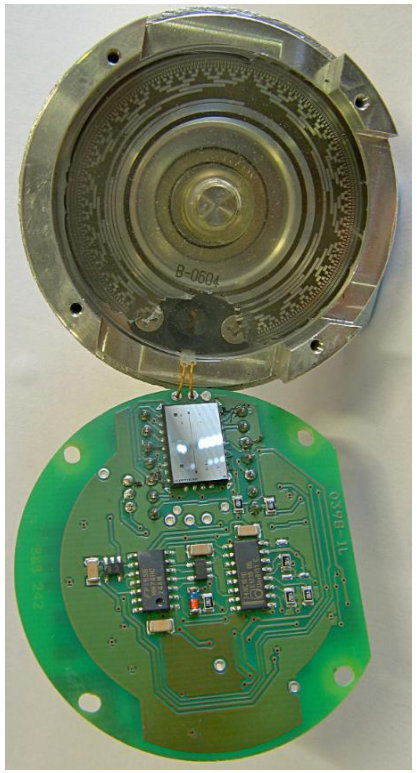
Dual	3Xs Gray
0000	-
0001	-
0010	0
0011	-
0100	4
0101	3
0110	1
0111	2
1000	-
1001	-
1010	9
1011	-
1100	5
1101	6
1110	8
1111	7

4 Bit Gray Code

Bei 3XS Gray Code handelt es sich um einen robusten 1-schrittigen Code, d.h. beim Übergang von einer Dezimalziffer zur nächsten ändert sich nur ein Bit. Das bietet Vorteile bei der Umwandlung analoger in digitale Signale. Er findet häufig Anwendung bei industriellen Winkel- und Längengebern.



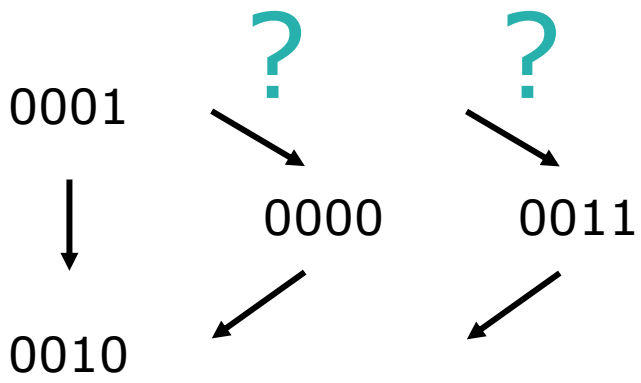
https://www.youtube.com/watch?v=-Qk--Sjgq78



Quelle: https://upload.wikimedia.org/wikipedia

Bei einer gleichzeitigen Umschaltung von 2 Ziffern könnten sich Fehlschaltungen ergeben.

Als Beispiel für einen derartigen Fehler soll die Umschaltung von $1_{(10)}$ auf $2_{(10)}$ im Dualcode betrachtet werden:

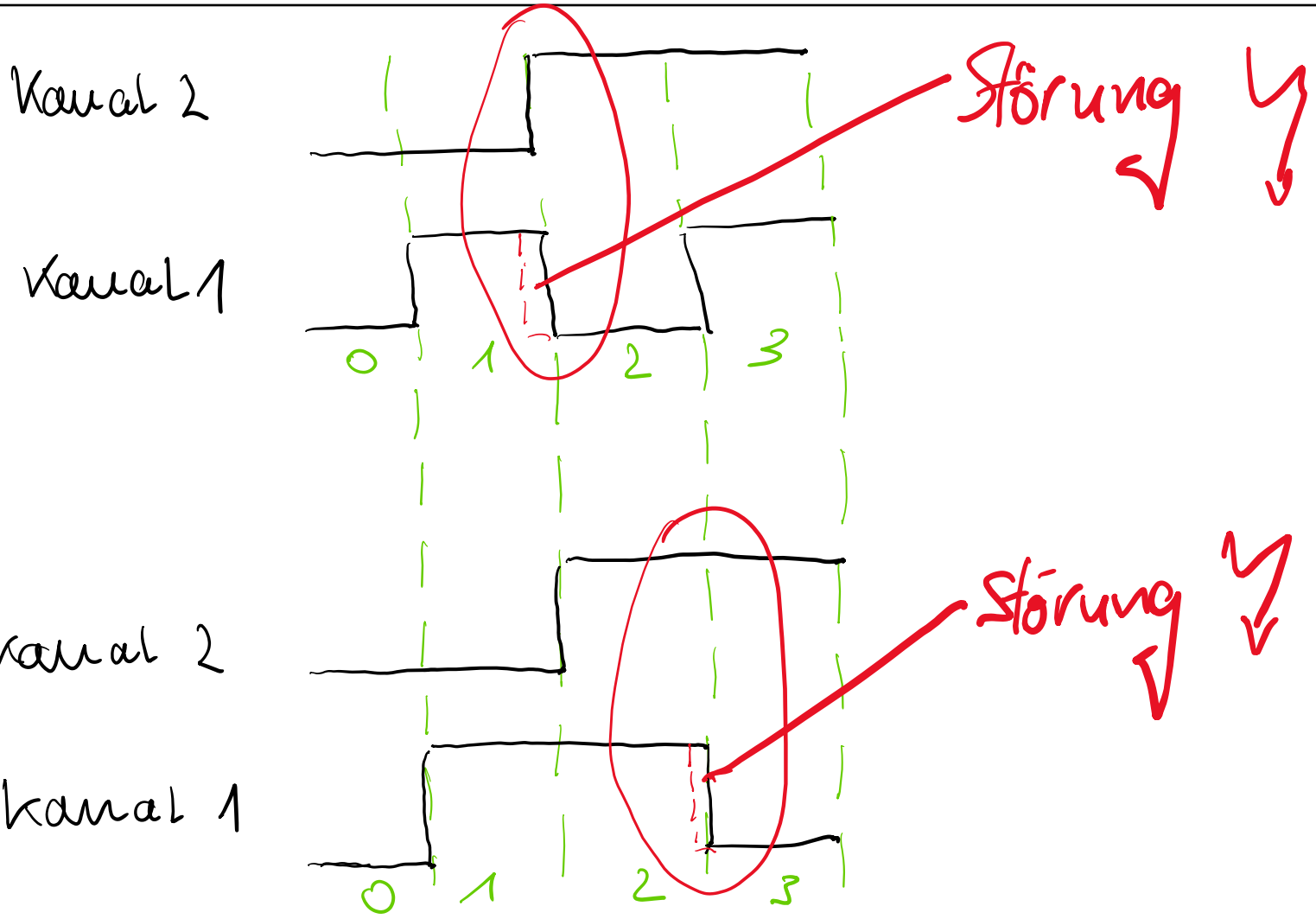


2 Bit Code

0	0	0	1
1	0	1	2
2	1	0	1
3	1	1	
Kanal	2	1	

2Bit-Gray Code

0	0	0	1
1	0	1	1
2	1	1	1
3	1	0	
Kanal	2	1	



Digitale Daten
Hamming Distanz

Die Hamming-Distanz ist ein Maß für die Unterschiedlichkeit von Zeichenketten. Die Hamming-Distanz zweier Blöcke mit gleicher Länge (sogenannter Codewörter) ist dabei die Anzahl der unterschiedlichen Stellen.

Die Hamming-Distanz wird zur Fehlererkennung und zur Fehlerkorrektur benutzt, indem Dateneinheiten, die über eine Übertragungsstrecke empfangen werden, mit gültigen Zeichen verglichen werden. Ob eine Fehlererkennung oder -korrektur stattfinden kann, hängt von der Hamming-Distanz (HD) ab.

0000	1001	1010
0001	0110	1111
HD=	HD=	HD=

HD_{min}=

Digitale Daten
Prüf- oder Paritätsbit

Das Paritätsbit einer Folge von Bits dient als Ergänzungsbit, um die Anzahl der mit 1 belegten Bits (inklusive Paritätsbit) der Folge als gerade oder ungerade zu ergänzen.

Die „Parität“ der Bitfolge heißt:

- gerade (englisch even), wenn die Anzahl der mit 1 belegten Bits in der Folge (inkl. Paritätsbit) gerade ist
- ungerade (englisch odd), wenn die Anzahl der mit 1 belegten Bits in der Folge (inkl. Paritätsbit) ungerade ist.

Das Paritätsbit kann zur Erkennung von Übertragungsfehlern kontrolliert werden.

8-4-2-1	Prüfbit
0000	1
0001	0
0010	0
0011	1
0100	0
0101	1
0110	1
0111	0
1000	0
1001	1

Beispiel:
Das Prüfbit ist so gewählt, dass jedes Zeichen eine ungerade Anzahl von Elementen im Zustand 1 enthält.

Digitale Daten
Technische Darstellung von Informationen

Positive Logik:
0 ≙ L (Low) näher an $-\infty$
1 ≙ H (High) näher an $+\infty$

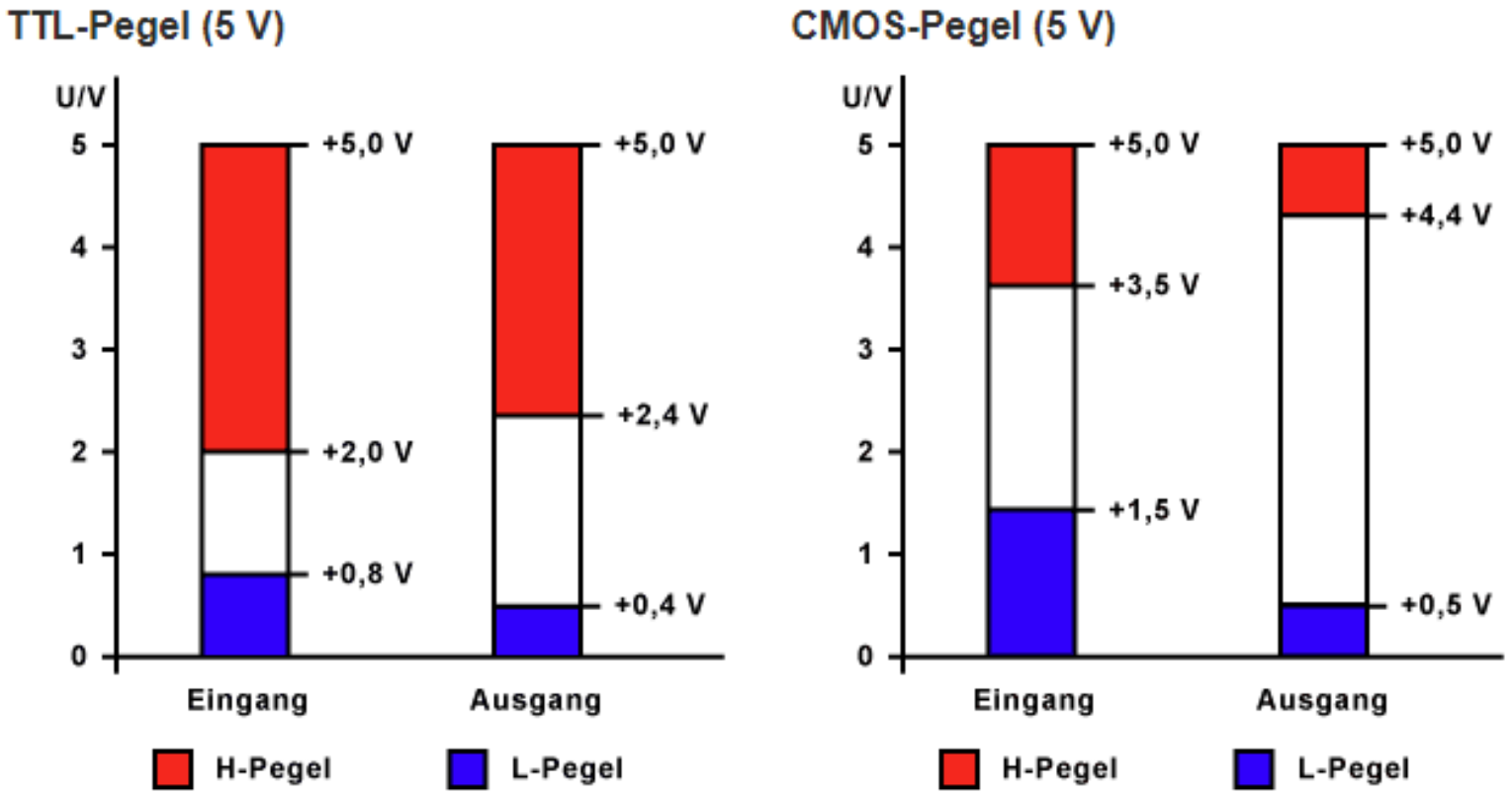
„physikalische Darstellung“
(Bsp. 0V, -10V, 4mA)
(Bsp. +5V, +10V, 20mA)



Negative Logik:
0 ≙ H (High)
1 ≙ L (Low)

Findet Anwendung bei der V. 24 oder RS 232 Schnittstelle.

Digitale Daten
Technische Darstellung von Informationen



Beispiel: Vierfach-NAND-Gatter

