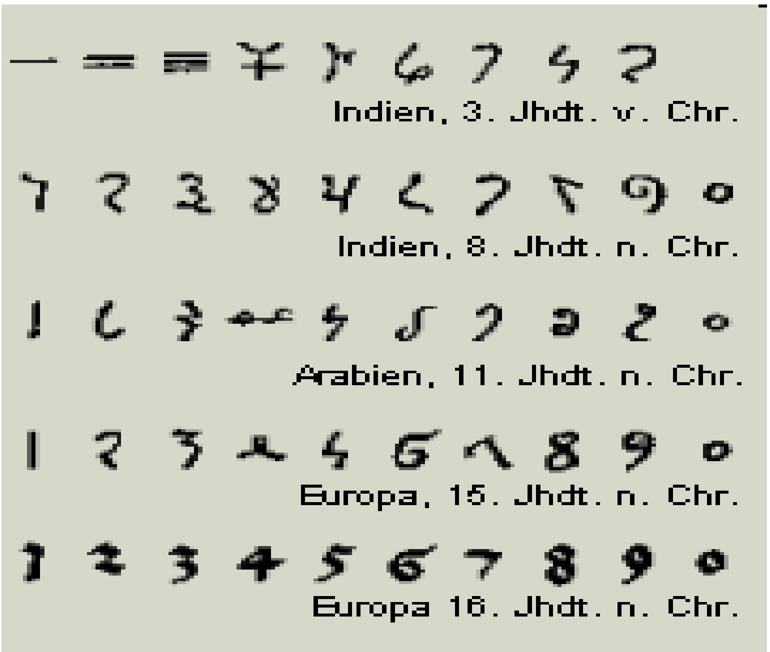
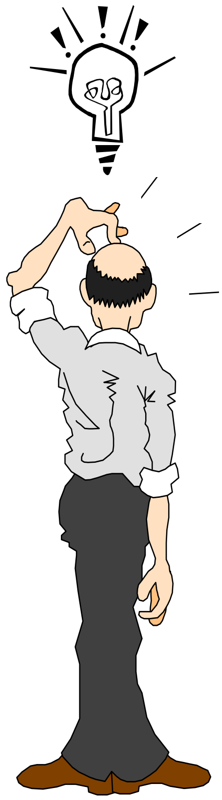
Zahlensysteme





1. Zahlensysteme

1.1. Daten

Daten stellen Information (= Inhalt und Bedeutung einer Nachricht) in einer maschinell verarbeitbaren Form dar (auf Grund bekannter oder unterstellter Abmachungen).

Verarbeitung von Daten
Damit ein (digitaler) Rechner Information verarbeiten kann, müssen entsprechende Daten in „berechenbarer" Form vorliegen, d.h., in Form von (binären) Zahlen, die nach bestimmten Rechenregeln be- und verarbeitet werden.

Rechnen
Um Rechnen zu können, muss man zuerst Zählen können und ein System zur Darstellung von Zahlen besitzen.

1.2. Überblick Zahlensysteme

Im Laufe der Menschheitsgeschichte wurden verschiedenste Rechenhilfsmittel und damit auch unterschiedliche Arten der Zahlendarstellung entwickelt:

- Kerbhölzer (Strichmethode)
- Knotenschnüre (Kipus)
- Abakus (Suan-Pan - Verfahren)
- Römische Zahlenschreibweise
- Hindu-Arabisches Zahlensystem

Je nach Art der Zusammenfassung und der Anordnung der Zeichen bezeichnet man diese Systeme als Additions-, bzw. als Stellenwertsysteme
(Positionssysteme, Positionalsysteme).

1.2.1. Additionssystem

Zahlen werden durch Aneinanderreihung von vordefinierten Zeichen dargestellt.

Das bekannteste ist das Römische Zahlensystem:

I	V	X	L	C	D	M
1	5	10	50	100	500	1000

Regeln für den Aufbau der Zahlendarstellung:

- Regel allgemein ... Zeichen für größeren Zahlenwert steht immer links vom kleineren
  Ausnahme: maximal ein kleineres Zahlzeichen darf links von
  einem größeren angeschrieben werden
- Regeln für I, X, C ... nicht mehr als drei gleiche Zeichen hintereinander
- Regel für die Zusammensetzung ungleicher Zeichen:
  -) Kleineres Zeichen steht rechts von größerem Zahlzeichen ---> Wert addieren
  -) Kleineres Zeichen steht links von größerem Zahlzeichen ---> Wert subtrahieren
- Regel für V, L, D ... kommen in einer Zahlendarstellung jeweils höchstens einmal vor
- Regel für M ... darf in einer Zahlendarstellung beliebig oft vorkommen

Beispiel:

Römische Darstellung	Dezimale Darstellung
XXX	30
XII	12
XXXVII	37
DXIX	519
MCMLXXXIX	1989
MMCCXXII	2222

Nachteil von Additionssystemen:
Rechenoperationen können nur sehr umständlich durchgeführt werden.

1.2.2. Stellenwertsystem (Positionalsystem)

Zahlen werden durch Zeichen (--> Ziffern) dargestellt, die in Abhängigkeit von ihrer Position (--> Stelle) eine andere Bedeutung (--> Wertigkeit) besitzen.

Das bekannteste ist das (Hindu-)Arabische Zahlensystem, das die Zeichen 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 zur Darstellung von Zahlenwerten verwendet.
[Die Inder haben auch vor ca. 1500 Jahren die Null (0) als Zeichen für nicht besetzte Stellen innerhalb einer Zahlendarstellung erfunden.]

Das Wesen dieses Systems besteht darin, daß jeweils zehn Grundeinheiten zu einer neuen Gruppe zusammengefaßt werden, z.B. die Einer zu einem Zehner, davon wieder zehn zu einem Hunderter, usw., bis der Zahlenwert erreicht ist. Für eine jeweils neue Gruppe wird jedoch kein neues Symbol eingeführt, sondern sie wird durch die Ziffernposition innerhalb dieser Zahl dargestellt.

Da jeweils Zehnergruppen gebildet werden, spricht man auch vom *Dekadischen Zahlen-system* [griech.: deka - zehn] oder vom *Dezimalen Zahlensystem* [lat.: decem - zehn].

Beispiel: Die Dezimalzahl 2222 verwendet viermal dasselbe Zeichen.
Innerhalb der Zahlendarstellung kommt die ‚Zwei' aber an verschiedenen Stellen vor und hat daher, im Unterschied zum römischen System, jeweils einen anderen Stellenwert.
Dabei steht der niedrigste Stellenwert - der Einer - am weitesten rechts.

2222 bedeutet 2*Tausend + 2*Hundert + 2*Zehn + 2*Eins

MMCCXXII bedeutet 1000 + 1000 + 100 + 100 + 10 + 10 + 1 + 1

Andere Stellenwertsysteme, deren Basis nicht Zehn ist:

- Zwölfersystem ... Dutzend, Gros
- Sexagesimalsystem [Sechzigersystem] ... Stunden, Minuten, Sekunden; Kreiseinteilung
- Hexadezimalsystem [Sedezimalsystem] ... Basis 16
- Oktalsystem ... Basis 8
- Dualsystem [Binärsystem] ... Basis 2

1.2.3. Verschiedene Stellenwertsysteme

Übersicht: Die wichtigsten Stellenwertsysteme

Name des Zahlensystems	Basis	Ziffernvorrat	Beispiel
Dual-, bzw. Binärsystem	2	0,1	1001
(Quartärsystem)	4	0,1,2,3	3312
Oktalsystem	8	0,1,2,3,4,5,6,7	173
Dezimalsystem	10	0,1,2,3,4,5,6,7,8,9	46798
Hexadezimalsystem	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F	CAFE

1.3. Aufbau eines Zahlensystems zur Basis B

1.3.1. Begriffe

Ziffer: Zeichen, aus denen die Zahl aufgebaut ist
Ziffernfolge: die aufeinanderfolgenden Ziffern einer Zahl
Signifikante Stellen: in einer Ziffernfolge werden, von links her, die Ziffern ab der ersten von Null verschiedenen Ziffer als ‚signifikante Stellen' bezeichnet
Basis: größer als 1, ganzzahlig; ein Zahlensystem mit der Basis B hat B verschiedene Ziffern, der Wert der größten Ziffer ist B-1
Stellenwert: jeder Stelle einer Zahl ist ein bestimmter Wert, der Stellenwert, zugeordnet. Diese Werte sind Potenzen der Basis B: Stellenwert = ‚Basis' hoch ‚Stellenzahl' --> --> Stellenwert der Stelle i ist B^i
Stellenzahl 'i': für ganzzahlige Stellen nimmt i den Wert 0, 1, 2, 3, …., n an; für gebrochenzahlige Stellen hat i den Wert -1, -2, -3, …., -m .

1.3.2. Zahlenwert ganzer Zahlen

Der Wert einer vorgegebenen Zahl in einem Zahlensystem zur Basis B wird mittels folgender Summe errechnet:

$$Z = z_n * B^n + z_{n-1} * B^{n-1} + \ldots\ldots + z_1 * B^1 + z_0 * B^0 \qquad B^0 = 1$$

Anders angeschrieben gilt für Z:

$$Z = \sum_{i=0}^{n} z_i * B^i \ ; \quad 0 \le z_i < B \ , \quad 0 \le n < \infty \ , \quad B > 1 \ ,$$ n … höchstwertigste Stelle der Zahl

Beispiel: Ganzzahlige Werte in verschiedenen Zahlensystemen

Binärsystem	Oktalsystem	Dezimalsystem	Hexadezimalsystem
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	A
1011	13	11	B
1100	14	12	C
1101	15	13	D
1110	16	14	E
1111	17	15	F
10000	20	16	10

1.3.3. Zahlenwert gebrochener Zahlen

Auch der gebrochenzahlige Wert ist die Summe der einzelnen Stellenwerte:

z.B. …….. $3{,}456 = 3*10^0 + 4*10^{-1} + 5*10^{-2} + 6*10^{-3}$

d.h., der gebrochene Anteil eines Zahlenwertes wird durch folgende Summenformel dargestellt:

$$Z_{gb} = z_{-1} * B^{-1} + z_{-2} * B^{-2} + \ldots + z_{-m+1} * B^{-m+1} + z_{-m} * B^{-m} \qquad B^0 = 1, \quad B^{-m} = \frac{1}{B^m}$$

Der gesamte Zahlenwert hat somit folgende Darstellung:

$$Z = z_n * B^n + z_{n-1} * B^{n-1} + \ldots + z_1 * B^1 + z_0 * B^0 + z_{-1} * B^{-1} + \ldots + z_{-m+1} * B^{-m+1} + z_{-m} * B^{-m}$$

Allgemein lässt sich diese Summe folgendermaßen anschreiben -
- gebrochenzahliger Anteil:

$$Z_{gb} = \sum_{i=-m}^{-1} z_i * B^i \; ; \quad 0 \le z_i \langle B, \; -\infty < m \le -1, \; B \rangle 1,$$ m … niederwertigste Stelle der Zahl

- ganzzahliger Anteil: Z_i … Ziffern des Zahlensystems ($0 \le z_i < B$)

$$Z_{gz} = \sum_{i=0}^{n} z_i * B^i \; ; \quad 0 \le z_i \langle B, \; 0 \le n \langle \infty, \; B \rangle 1,$$ n … höchstwertigste Stelle der Zahl

Zahlenwert $Z = Z_{gz} + Z_{gb} = \sum_{i=-m}^{n} z_i * B^i$,

dabei gilt: Z_i … Ziffern des Zahlensystems ($0 \le z_i < B$)
B … Basis des Zahlensystems ($B > 1$)
n … Ordnungszahl der ganzzahligen Stellen ($+\infty > n \ge 0$)
m …Ordnungszahl der gebrochenzahligen Stellen ($-1 \ge m > -\infty$)

Beispiel: Z = 931,5706
Ziffernfolge … 9315706
Basis … B=10
Ordnungszahl der signifikantesten Stelle … n=2
Ordnungszahl der letzten (von Null verschiedenen) Stelle … m=4
Ganzzahliger Anteil … $Z_{gz}=931$
Gebrochenzahliger Anteil … $Z_{gb}=0{,}5706$

$Z_{gz}= 9*10^2 + 3*10^1 + 1*10^0$
$Z_{gb}= 5*10^{-1} + 7*10^{-2} + 0*10^{-3} + 6*10^{-4}$

$$Z = Z_{gz} + Z_{gb} = \sum_{-4}^{2} z_i * B^i$$

1.3.4. Notation

Um klarzustellen, in welchem Zahlensystem die Darstellung einer Zahl erfolgt, wird im Allgemeinen die Basis des Zahlensystems am rechten Ende der Zahl in tiefgestellter Form angeschrieben:

Beispiel: $2137{,}1457_8$ … Oktalzahl
$AB63F{,}08_{16}$ … Hexadezimalzahl
$110101{,}001_2$ … Binärzahl
$4203{,}134_5$ … Zahl im Zahlensystem zur Basis 5

1.4. Umrechnen von Zahlen zur Basis B ins Dezimalsystem

1.4.1. Umrechnen von ganzen Zahlen zur Basis B ins Dezimalsystem

1.4.1.1. Potenzmethode

Das Umrechnen mittels der Potenzmethode basiert direkt auf der Zahlendarstellung eines Stellenwertsystems → *Die Zählung der Stellenzahl für die ganzzahligen Stellen beginnt bei Null und setzt mit 1, 2, 3, 4, …… fort.*

Beispiele:

$$\begin{aligned}4170_8 &= 4*8^3 + 1*8^2 + 7*8^1 + 0*8^0 = \\ &= 4*512 + 1*64 + 7*8 + 0*1 = \\ &= 2048 + 64 + 56 + 0 = \underline{\underline{2168}}_{10}\end{aligned}$$

$$\begin{aligned}CAFE_{16} &= 12*16^3 + 10*16^2 + 15*16^1 + 14*16^0 = \\ &= 12*4096 + 10*256 + 15*16 + 14*1 = \\ &= 49152 + 2560 + 240 + 14 = \underline{\underline{51966}}_{10}\end{aligned}$$

$$\begin{aligned}1001011_2 &= 1*2^6 + 0*2^5 + 0*2^4 + 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0 = \\ &= 1*64 + 0*32 + 0*16 + 1*8 + 0*4 + 1*2 + 1*1 = \\ &= 64 + 8 + 2 + 1 = \underline{\underline{75}}_{10}\end{aligned}$$

1.4.1.2. Hornermethode (Hornerschema)

Ausgehend von der Summenformel für ganze Zahlen kann folgendes Verfahren zur Berechnung des Zahlenwertes im Dezimalsystem angewandt werden:

$$Z_{gz} = z_n * B^n + z_{n-1} * B^{n-1} + ... + z_1 * B^1 + z_0 * B^0$$ ...... B herausheben

$$Z_{gz} = (z_n * B^{n-1} + z_{n-1} * B^{n-2} + ... + z_2 * B^1 + z_1) * B + z_0$$ ...... B herausheben

$$Z_{gz} = ((z_n * B^{n-2} + z_{n-1} * B^{n-3} + ... + z_2) * B + z_1) * B + z_0$$

bis B nicht mehr heraushebbar ist

$$Z_{gz} = ((...(z_n * B + z_{n-1}) * B + ... + z_2) * B + z_1) * B + z_0$$

Beispiel:

$1032_4 = 1*4^3 + 0*4^2 + 3*4^1 + 2*4^0 =$
$(1*4^2 + 0*4^1 + 3)*4 + 2 =$
$((1*4^1 + 0)*4 + 3)*4 + 2 =$
$(4*4 + 3)*4 + 2 =$
$(19*4) + 2 = 76 + 2 = \underline{\underline{78}}_{10}$

Rechenschema zum praktischen Einsatz der Hornermethode:

1.) Ziffern der umzurechnenden Zahl nebeneinander anschreiben
2.) Mit einer Leerzeile Abstand einen Strich darunter ziehen
3.) Beginnend bei der höchsten Stelle ziffernweise die Summe bilden
⇨ für die höchstwertigste Stelle ist der Summand gleich Null
4.) Die jeweils letzte Summe wird mit der Basis der Ausgangszahl multipliziert → dieses Produkt ist der Summand für die nächste Stellensumme

	1	0	3	2
+	0	4	16	76
	*4	*4	*4	
	1	4	19	$\underline{\underline{78}}_{10}$

Probe mit der Potenzmethode:

1.4.2. Umrechnen von gebrochenen Zahlen zur Basis B ins Dezimalsystem

1.4.2.1. Potenzmethode

Das Umrechnen mittels der Potenzmethode basiert direkt auf der Zahlendarstellung eines Stellenwertsystems → *Die Zählung der Stellenzahl für die gebrochenzahligen Stellen beginnt bei -1 und setzt mit -2, -3, -4, …… fort.*

Beispiele:

$0{,}35_8 = 3*8^{-1} + 5*8^{-2} =$

$= 3*{}^{1}/_{8} + 5*{}^{1}/_{64} =$

$= {}^{24}/_{64} + {}^{5}/_{64} = \underline{\underline{{}^{29}/_{64}}}_{10}$

1.4.2.2. Hornermethode (Hornerschema)

Ausgehend von der Summenformel für gebrochene Zahlen kann folgendes Verfahren zur Berechnung des Zahlenwertes im Dezimalsystem angewandt werden:

$$Z_{gb} = z_{-1} * B^{-1} + z_{-2} * B^{-2} + ... + z_{-m+1} * B^{-m+1} + z_{-m} * B^{-m}$$

$$Z_{gb} = \frac{z_{-1}}{B} + \frac{z_{-2}}{B^2} + ... + \frac{z_{-m+1}}{B^{m-1}} + \frac{z_{-m}}{B^m}$$

Terme als Bruchzahlen anschreiben….
….und auf gleichen Nenner bringen

$$Z_{gb} = \frac{z_{-1} * B^{m-1} + z_{-2} * B^{m-2} + ... + z_{-m+1} * B + z_{-m}}{B^m}$$

Der Zähler dieses Bruchs hat die gleiche Struktur wie der ganzzahlige Anteil eines Zahlenwertes ⇨

⇨ <u>*Algorithmus zur Berechnung des gebrochenzahligen Anteils eines Zahlenwertes mittels Hornermethode:*</u>

Der Nachkommaanteil einer Zahl im Zahlensystem zur Basis B wird mit dem Hornerverfahren in eine Dezimalzahl umgerechnet und dieses Ergebnis mit dem Faktor „Basis hoch Anzahl der Nachkommastellen" dividiert → der Quotient ist die dezimale Darstellung des gebrochenzahligen Anteils der Ausgangszahl.

D.h., wir haben die gebrochene Zahl in eine ganze Zahl umgewandelt, für die das Hornerschema angewendet werden kann. Danach muss die erhaltene Zahl durch B^m dividiert werden.

Beispiel:

$0{,}35_8$ ---> 3 5
\+ 24
*8
3 29 --> ${}^{29}/_{8^2} = \underline{\underline{{}^{29}/_{64}}}_{10} = 0{,}35_8$

1.5. Umrechnen von Dezimalzahlen in ein Zahlensystem zur Basis B

1.5.1. Umrechnen von ganzen Dezimalzahlen ins Zielsystem

1.5.1.1. Probiermethode (Potenzmethode)

Durch Betrachtung der Zahlendarstellung $Z_{gz} = z_n * B^n + \underbrace{z_{n-1} * B^{n-1} + \ldots + z_1 * B^1 + z_0 * B^0}$

erkennt man, dass bei Division durch B^n der Term $z_{n-1} * B^{n-1} + \ldots + z_1 * B^1 + z_0 * B^0$ als Rest stehen bleibt; z_n tritt bei dieser Division als eine Ziffer der Zahlendarstellung im gesuchten Zahlensystem zur Basis B auf.
Durch wiederholte Division durch B^n können damit alle Ziffern der gesuchten Zahlendarstellung ermittelt werden.

Vorgangsweise:

1.) Die höchste Potenz n von der Basis B des Zielsystems bestimmen, die in der gegebenen Dezimalzahl noch enthalten ist; die gesuchte Zahl hat dann n+1 Stellen.
2.) Ermitteln, wie oft der Zahlenwert mit der höchsten Potenz in Z_{gz} enthalten ist - das ganzzahlige Ergebnis dieser Division ist die erste Ziffer der gesuchten Zahl.
3.) Den (ganzzahligen) Rest der Division bestimmen und mit diesem Rest so wie mit der ursprünglichen Dezimalzahl verfahren, wobei der Wert von n jeweils um 1 verringert wird.

Beispiel:

89_{10} --> 4er-System?

I) .... $4^2=16$ --> ist in 89 enthalten
$4^3=64$ --> ist in 89 enthalten

II).......... $4^4=256$ --> ist in 89 nicht mehr enthalten; daher ist die Stellenanzahl der gesuchten Zahlendarstellung 3+1= <u>4 Stellen</u> .

III) .... z_3 - [die signifikanteste Stelle der Zielzahl] - ist daher 1 ---> $^{89}/_{64}=1$, Rest=25

z_2 ist daher 1 --> $^{25}/_{16}=1$, Rest=9
z_1 ist daher 2 --> $^{9}/_{4}=2$, Rest=1
z_0 ist daher 1 --> $^{1}/_{1}=1$, Rest=0

$89_{10} = \underline{\underline{1121}}_4$ .... Probe: $1*4^3 + 1*4^2 + 2*4^1 + 1*4^0 = 64 + 16 + 8 + 1 = 89_{10}$

1.5.1.2. Restmethode (Hornerschema)

Die Hornermethode ergibt für ganzzahlige Werte folgende Darstellungsweise (siehe 1.4.1.2.):

$$Z_{gz} = ((...(z_n * B + z_{n-1}) * B + ... + z_2) * B + z_1) * B + z_0$$

Dabei erkennt man, dass bei ganzzahliger Division der Zahl Z_{gz} durch die Basis B der gesuchten Zahlendarstellung der (ganzzahlige) Rest dieser Division die Ziffer z_0 der gesuchten Zahl ist.
Der (neue) Quotient wird wieder durch B dividiert, wodurch man die Ziffer z_1 als den ganzzahligen Rest erhält; usw. usf.

Vorgangsweise:
Die gegebene Dezimalzahl wird solange durch die Basis B der gesuchten Zahlendarstellung dividiert, bis der **ganzzahlige** Quotient Null ist. Die dabei anfallenden ganzzahligen Divisionsreste sind die Ziffern der gesuchten Zahl, allerdings in umgekehrter Reihenfolge – d.h., der erste Divisionsrest ist die niederwertigste Stelle der gesuchten Zahl.

→ Nicht vergessen, einen größeren Rest als 9 durch die entsprechende Ziffer des Zielsystems zu ersetzen !!!

Beispiel: 89_{10} --> 4er-System?

$$\left.\begin{array}{l} {}^{89}/_4 = 22,\ \text{Rest } 1 \\ {}^{22}/_4 = 5,\ \text{Rest } 2 \\ {}^{5}/_4 = 1,\ \text{Rest } 1 \\ {}^{1}/_4 = 0,\ \text{Rest } 1 \end{array}\right\} 89_{10} = 1121_4$$

höchstwertigste Stelle (MSB) der Zielzahl

niederwertigste Stelle (LSB) der Zielzahl

347_{10} --> Hexadezimalsystem?

$$\left.\begin{array}{l} {}^{347}/_{16} = 21,\ \text{Rest } 11 \Rightarrow B \\ {}^{21}/_{16} = 1,\ \text{Rest } 5 \\ {}^{1}/_{16} = 0,\ \text{Rest } 1 \end{array}\right\} 347_{10} = 15B_{16}$$

1.5.2. Umrechnen von gebrochenen Dezimalzahlen ins Zielsystem

Ausgehend von der Zahlendarstellung $Z_{gb} = z_{-1} * B^{-1} + z_{-2} * B^{-2} + ... + z_{-m+1} * B^{-m+1} + z_{-m} * B^{-m}$ erkennt man, dass der ganzzahlige Anteil von Z_{gb}*B die erste Ziffer der Zahlendarstellung zur Basis B ergibt: $Z_{gb} * B = z_{-1} + z_{-2} * B^{-1} + ... + z_{-m+1} * B^{-m+2} + z_{-m} * B^{-m+1}$;
d.h., durch wiederholte Anwendung dieses Verfahrens erhält man alle Ziffern der Zahl im Zahlensystem zur Basis B (⇨ Multiplikationsmethode).

Vorgangsweise:
Man multipliziert den Nachkommateil der Dezimalzahl mit der Basis des Zielsystems. Der ganz-zahlige Teil dieses Produkts ist die erste Ziffer der gesuchten Zahl; der (neue) Nachkommateil wird wiederum mit der Basis des Zielsystems multipliziert. Der ganzzahlige Teil dieses Produkts ist die zweite Ziffer der gesuchten Zahl; usw. usf.
Das Verfahren endet, wenn entweder der Nachkommateil des letzten Produkts Null ist oder eine Periode auftritt oder wenn die geforderte Genauigkeit der gesuchten Zahlendarstellung erreicht ist.

Beispiel: $0{,}123_{10}$ --> Oktalsystem? …….. $0{,}123_{10} \approx 0{,}07676_8$

$0{,}123 * 8 = 0{,}984$ .... 0 höchstwertigste Stelle der Zielzahl (MSB)
$0{,}984 * 8 = 7{,}872$ 7
$0{,}872 * 8 = 6{,}976$ 6
$0{,}976 * 8 = 7{,}808$ 7
$0{,}808 * 8 = 6{,}464$ 6
$0{,}464$

1.6. Umwandlungen zwischen Zahlensystemen mit der Basis 2^n

Teilweise ist es möglich, Zahlenkonvertierungen einfacher durchzuführen. Aus der Tabelle auf Seite 5 ersieht man, dass jede Hexadezimalziffer durch eine Kombination von vier Binärziffern (*inklusive führender Nullen*) dargestellt werden kann; jede Oktalziffer kann durch drei Binärziffern gleichwertig dargestellt werden.

Z. B.:

$2^1 = 2 \Rightarrow n = 1$ *Dual*	$2^3 = 8 \Rightarrow n = 3$ *Oktal*	$2^4 = 16 \Rightarrow n = 4$ *Hexadezimal*
001011	13	B
00010011	23	13

Es ist unschwer zu erkennen, dass diese direkten Umformungen immer dann möglich sind, wenn die Basen der Zahlensysteme der umzurechnenden Zahlen sich als 2^n darstellen lassen.

→ <u>Gebräuchliche Zahlensysteme mit Basis 2^n:</u>

n=1 ⇨ B=2 …… Dualsystem / Binärsystem
n=3 ⇨ B=8 …… Oktalsystem
n=4 ⇨ B=16 ….. Hexadezimalsystem

1.6.1. Umwandeln vom Dualsystem ins Zahlensystem mit $B=2^n$

Erklärung am Beispiel $B=2^4$: ausgehend vom Dezimalpunkt wird die Dualzahl in Vierergruppen von Binärziffern eingeteilt und diese Gruppen als hexadezimale Ziffern dargestellt.
Beispiel:

1.6.2. Umwandeln vom Zahlensystem mit $B=2^n$ ins Dualsystem

Erklärung am Beispiel $B=2^3$: jede Oktalziffer wird mit ihrem binären Äquivalent angeschrieben.
Beispiel:

1.6.3. Umwandeln einer Zahl mit Basis 2^m in eine Zahl mit Basis 2^n

Diese Art der Konvertierung erfolgt über den Zwischenschritt der Binärdarstellung durch Kombina-tion der beiden vorher genannten Methoden.
Erklärung am Beispiel der Umwandlung von $B_m=2^4$ (Hexadezimalsystem) in $B_n=2^3$ (Oktalsystem).

Beispiel: Umwandlung der Zahl $EAB5,7C4D_{16}$ ins Oktalsystem →

- → Ausgehend vom Dezimalpunkt wird jede Hexadezimalziffer durch die entsprechende Vierergruppe von Binärziffern angeschrieben:

Nun werden die Binärziffern, ausgehend jeweils vom Komma, zu Dreiergruppen zusammen-gefasst; dabei darf nicht vergessen werden, am linken und vor allem am rechten Rand die letzte Gruppe zu vervollständigen und mit Nullen zu ergänzen. Danach werden die Dreiergruppen der Binärziffern durch die entsprechende Oktalziffer ersetzt:

1.7. Rechnen in Zahlensystemen zur Basis B

1.7.1. Addition

Im Dezimalsystem:

$$\begin{array}{lrrr} & 3 & 2 & 6\\ & & 4 & 8\\ \text{Übertrag} & & 1 & \\ \hline & 3 & 7 & 4 \\ \hline\hline \end{array}$$

Beim Addieren in einem Zahlensystem zur Basis B verwendet man die gleiche Methode wie im Dezimalsystem:
Man addiert stellenweise von rechts nach links, wobei jeweils der Übertrag der vorhergehenden Summe berücksichtigt werden muss.
Um eine Stelle der Zahlen mit Basis B zu addieren, müssen

1. die einzelnen Ziffern ins Dezimalsystem umgerechnet werden
2. diese Werte addiert werden (im Dezimalsystem)
3. das Ergebnis wieder ins Zahlensystem zur Basis B umgerechnet werden.
   Überträge werden wie im Dezimalsystem behandelt.

Zum Beispiel ⇒

⇒ Oktalsystem:

$$\begin{array}{rrr} 3 & 2 & 6\\ & 4 & 7\\ \hline 3 & 7 & 5 \\ \hline\hline \end{array}$$

Nebenrechnung: $6_8 + 7_8 = 15_8$ (Restmethode: $6_{10}+7_{10}=13_{10} :8= 1_8$ Rest 5_8)

1.7.2. Subtraktion

Im Dezimalsystem:

$$\begin{array}{lrrr} & 1 & 2 & 3\\ - & & 8 & 2\\ \text{Ausborgung} & 1 & & \\ \hline & & 4 & 1 \\ \hline\hline \end{array}$$

Bei der Subtraktion am Index 1 wird der Wert 10 von der Stelle mit Index 2 ausgeborgt und bei der Subtraktion am Index 2 wieder zurückgegeben.

Allgemein formuliert wird beim Ausborgen von der nächsthöheren Stelle die Basis B ausgeborgt, die dann im nächsten Schritt als 1 an diese verborgende Stelle wieder zurückgegeben werden muss.

Zum Beispiel ⇒

⇒ Oktalsystem:

$$\begin{array}{rrrrrr} & 1 & 0 & 1 & 6 & 4\\ & & {\scriptstyle +8} & {\scriptstyle +8} & & \\ - & & 1 & 7 & 2 & 1\\ \hline & & 6 & 2 & 4 & 3 \\ \hline\hline \end{array}$$

1.7.3. Multiplikation

Im Dezimalsystem: HZE - Stelle

$$\begin{array}{r} \underline{4567 * 345} \\ 13701 \\ 18268 \\ \underline{22835} \\ 1575615 \end{array}$$

D. h., die Multiplikation ergibt sich aus mehreren Additionen.
Zuerst wird die Hunderterstelle multipliziert. Da nicht mit 3, sondern eigentlich mit 300 multipli-ziert wird, spart man sich das Anschreiben der beiden letzten Nullen. Sowohl bei der Zehnerziffer als auch bei der Einerziffer wird jeweils um eine Stelle nach rechts gerückt:

(4567*300) + (4567*40) * (4567*5)

In anderen Zahlensystemen ist der Vorgang analog.

Zum Beispiel ⇒

⇒ Oktalsystem:

$$\begin{array}{r} \underline{356_8 * 27_8} \\ 734 \\ \underline{3202} \\ 12542_8 \end{array}$$

Nebenrechnungen:

$2*6 = 12_{10} = 14_8$
$(2*5)+1 = 11_{10} = 13_8$
$(2*3)+1= 7_8$ Übertrag
$7*6 = 42_{10} = 52_8$ $(\underline{5}*8+\underline{2})$ Stelle
$(7*5)+5 = 40_{10} = 50_8$ $(\underline{5}*8+\underline{0})$
$(7*3)+5 = 26_{10} = 32_8$ $(\underline{3}*8+\underline{2})$

1.7.4. Division

Im Dezimalsystem:

$$\begin{array}{l} 6497 : 89 = 73 \\ 267 \\ 0 \end{array}$$

Zuerst wurde abgeschätzt, wie oft 89 in 649 enthalten ist [Vorgangsweise: Wie oft ist 9(!) in 64 enthalten → 7 mal]; dann wird 89 mit 7 multipliziert und in einem Arbeitsgang von 649 abgezogen, wobei 26 als Rest bleibt.
Ist der verbleibende Rest größer oder gleich dem Divisor, war die Abschätzung zu klein,
ist der Rest negativ, war die Abschätzung zu groß.
Zum Rest 26 wird die nächste Stelle des Dividenden (der zu teilenden Zahl) angeschrieben und es wird gefragt, wie oft 89 in 267 enthalten ist; sodann wird wie im ersten Schritt verfahren.
Da in diesem Fall kein Rest mehr bleibt, erhält man das exakte Ergebnis 73.
D. h., eine Division wird in aufeinanderfolgende Multiplikations- und Subtraktionsschritte zerlegt, wobei diesen Rechenvorgängen ein Abschätzen vorangeht.

Haben Dividend oder Divisor Nachkommastellen, so kann durch ‚Erweitern' erreicht werden, dass beide Zahlen ganzzahlig sind; z.B.: um die Division 267,35 : 4,567 auszuführen, wird die Division 267350 : 4567 durchgeführt (→ Verschieben des Dezimalpunktes; gilt für alle Stellenwertsysteme).

Zum Beispiel ⇒

⇒ Oktalsystem

$$\begin{array}{l} 3646 : 26 = 131_8 \\ 104 \\ 26 \\ 0 \end{array}$$

Nebenrechnung(en):

$$\begin{array}{r} \underline{26_8 * 3_8} \\ 102_8 \end{array}$$

3. Zahlendarstellungen im digitalen Rechner

3.1. Klassifizierung der Daten für die Darstellung von Zahlenwerten als Maschinenzahl

Die folgende Datenklassifizierung erfolgt unter dem Gesichtspunkt der Darstellung und Verarbeitung von Zahlenwerten in einem digitalen Computer:

3.2. Verarbeitung numerischer Daten - binäre Zahlendarstellung

3.2.1. Festkommadarstellung

Unter ‚Festkommadarstellung' versteht man eine Zahlendarstellung, bei der nur ganzzahlige binäre Werte abgespeichert werden.
Eventuell auftretende Nachkommastellen müssen durch einen entsprechenden Faktor multipliziert werden, um eine ganze Zahl zu erhalten;
d.h., dass jede dargestellte Zahl in ein einheitliches Format mit **einer fixen Anzahl von n Stellen** umgewandelt wird, bei dem der *gedachte Dezimalpunkt* an einer Stelle *fix* gehalten wird.

Z. B.: B(asis)=10; 3 Vorkomma- und 2 gedachte Nachkommastellen [n=5, gedachter Dezimalpunkt zwischen der 2. und 3. Stelle von rechts]

Computer verarbeiten Werte mit fester Länge (d.h., mit fixer Stellenanzahl n); diese Wortbreite ist zumeist 16, 32, 48 oder 64 Bits.

Die zur Verfügung stehenden n Stellen zur Darstellung einer ganzen Zahl können genutzt werden, um vorzeichenlose (VZ-los) oder vorzeichenbehaftete (VZ-behaftet) Zahlen abzuspeichern.

3.2.1.1. VZ-lose Zahlen

Alle n Bits (Stellen) können zur Darstellung der Zahl genutzt werden. Daraus ergibt sich im Binärsystem ein *Wertebereich* von *0 bis 2^n-1* (bzw. B^n-1) für die Speicherung von ganzen Zahlen →

→ z.B.: B=2, n=4 …… kleinste Zahl: 0 , größte Zahl: 2^4 -1=15

⇒ B=2:

Stellen-anzahl n	größter darstell-barer Zahlenwert	Anzahl der darstell-baren Zahlenwerte	
1	1	2 (2^1)	0, 1
2	3	4 (2^2)	00, 01, 10, 11
3	7	8 (2^3)	000, 001, 010, 011, 100, 101, 110, 111
4	15	16 (2^4)	0000, 0001, 0010, 0011, 0100, 0101, 0110, …

→ z.B.: B=2, n=16 …… kleinste Zahl: 0 , größte Zahl: 2^{16} -1=65535_{10}
→ z.B.: B=2, n=32 …… kleinste Zahl: 0 , größte Zahl: 2^{32} -1=4294967295_{10}

→ Bestimmung der Zahlendarstellung

$\left.\begin{array}{l}\text{d........Anzahl der Nachkommastellen (→ gedachter Dezimalpunkt)}\\ \text{g........Anzahl der Vorkommastellen (→ möglicher ganzzahliger Anteil)}\end{array}\right\}$ **n = g + d**

d.h., dass immer ganze Zahlen gespeichert werden, die bei Division durch B^d die ursprüngliche Zahl ergeben.

Vorgangsweise:

1. Abzubildenden Zahlenwert in das Zahlensystem mit der Zielbasis umrechnen.
2. Mit B^d multiplizieren.
3. Wenn notwendig, Nachkommastellen abschneiden oder runden.
4. Bei Bedarf links mit Nullen auffüllen.

[5. Wenn g < ganzzahlige Stellenanzahl, die dargestellt / gespeichert werden soll, ist diese Zahl nicht verarbeitbar.]

3.2.1.2. VZ-behaftete Zahlen

Bei vorzeichenbehafteten Zahlen wird ein Bit (oder allgemeiner: eine Stelle) für das Vorzeichen reserviert. Das Vorzeichen wird in der Regel durch das höchstwertigste Bit (Stelle) abgebildet, wobei der Wert ‚0' eine positive Zahl und Bitwert ‚1' eine negative Zahl kennzeichnet; $\Rightarrow$ $n = g + d + 1$

z.B.: n=8, d=3; B=10; z=-123,45 $\Rightarrow$ -12345 → 10123450 [g=8-3-1=4]

z.B.: n=8, d=2; B=2; z=$-15{,}75_{10}$ $\Rightarrow$ $-1111{,}11_2$ → 10111111_2 [g=8-2-1=5]

→ Im Binärsystem werden negative Zahlenwerte in der Regel jedoch im *Zweierkomplement* (⇨ siehe 3.2.2.) gespeichert.

3.2.2. Subtraktion durch Komplementaddition

Die Differenzenbildung von Zahlen mit n Stellen kann auch durch Komplementaddition erfolgen (‚Komplement' ⇒ ‚Ergänzung').

3.2.2.1. (B-1) – Komplement

(B-1) – Komplement einer Zahl = Ergänzung der Zahl auf den Wert
([nächsthöherer Stellenwert] – 1) ⇒

⇒ das bedeutet, dass die Zahl addiert zu ihrem Kompliment den höchsten Zahlenwert ergeben muss, der bei n Stellen möglich ist;
z. B. im Dezimalsystem wird die Ergänzung auf 10^n-1 durchgeführt, also zur höchsten mit n Stellen darstellbaren Zahl (d.h., alle n Stellen mit 9ern besetzt).

Beispiele für die Berechnung des (B-1) - Komplements:

Basis	*Zahl*	*ergänzt wird auf*	*(B-1) – Komplement*	*Erläuterung*
10	38	99	61	$38_{10} + 61_{10} = 99_{10} = 10^2-1$
8	127	777	650	$127_8 + 650_8 = 777_8 = 8^3-1$
2	101	111	010	$101_2 + 010_2 = 111_2 = 2^3-1$

> Im <u>Binärsystem</u> versteht man unter dem
> **Einerkomplement**
> jene Binärzahl, die durch Invertierung aller Bits der Ausgangszahl x entsteht; d.h., aus jeder 0 wird eine 1 und aus jeder 1 wird eine 0 .

Stehen n Bitstellen zur Verfügung, wird durch Bildung des Einerkomplements daraus die (positive) Zahl 2^n-1-x →

→ Die größte darstellbare Zahl in einem binären Zahlensystem ist 2^n-1, d.h. alle Bitstellen sind mit Einsen besetzt. Werden jetzt diejenigen Bitstellen, die bei der Ursprungszahl den Wert 1 tragen, auf Null gesetzt, dann entspricht das einer Subtraktion um den Wert x !

Beispiel(e) zum Einerkomplement:

3.2.2.2. B – Komplement [2er-Komplement]

Um die <u>Subtraktion</u> von Zahlen mit n Stellen auf die <u>Addition mittels Komplementbildung</u> zurückführen zu können, muss folgende Forderung erfüllt sein:

$$x + (-x) = 0$$

Dazu wird das *B-Komplement* einer Zahl benötigt: **B-Komplement = (B-1) – Komplement + 1**
Dieses B-Komplement einer Zahl y mit n Stellen wird im folgenden mit $K_{B,n}(y)$ bezeichnet.

Die *Subtraktion* x – y mit je n Stellen und x > y lässt sich nun *auf* die *Addition* von x + $K_{B,n}(y)$ *zurückführen*, wobei beim Ergebnis die entstehende Stelle n+1 nicht weiter betrachtet wird.

Beispiel(e):

Im Binärsystem versteht man unter dem
Zweierkomplement (2er-Komplement)
jene Binärzahl, die durch Bildung des Einerkomplements und nachfolgender Addition von 1 entsteht.

Beispiel: 010011001 → 101100110 + 1 = 101100111 ……..

…….. negative Zahlen werden also als $2^n - x$ dargestellt →

→ d.h., dass im Binärsystem *negative Zahlenwerte durch* die Berechnung des *2er-Komplements* ermittelt und somit *abgebildet* werden.

Beispiel(e) zum Zweierkomplement:

Um im Binärsystem die Subtraktion zweier Zahlen $z_1 - z_2$ durchzuführen, genügt es somit, zu z_1 das Zweierkomplement von z_2 zu addieren, wobei das Vorzeichen schon richtig gesetzt wird !!

Darstellbarer Zahlenbereich für positive und negative Werte am Beispiel vierstelliger Binärzahlen ⇒

$0 \div +7_{10}$	0000	**0**		
	0001	**1**		
	0010	**2**		
	0011	**3**		
	0100	**4**		
	0101	**5**		
	0110	**6**		
	0111	**7**		
$-1 \div -8_{10}$	1000	8	2^4-8 = 16-8 = 8 → **-8**	$K_{2,4}(1000)=1000=8 \Rightarrow -8$
	1001	9	2^4-9 = 16-9 = 7 → **-7**	$K_{2,4}(1001)=0111=7 \Rightarrow -7$
	1010	10	= 16-10 = 6 → **-6**	$K_{2,4}(1010)=0110=6 \Rightarrow -6$
	1011	11	**-5**	$K_{2,4}(1011)=0101=5 \Rightarrow -5$
	1100	12	**-4**	$K_{2,4}(1100)=0100=4 \Rightarrow -4$
	1101	13	**-3**	$K_{2,4}(1101)=0011=3 \Rightarrow -3$
	1110	14	**-2**	$K_{2,4}(1110)=0010=2 \Rightarrow -2$
	1111	15	**-1**	$K_{2,4}(1111)=0001=1 \Rightarrow -1$

⇒ mit vierstelligen Binärzahlen können die dezimalen Werte von 0 bis +7 und von -1 bis -8 dargestellt werden

Beispiele für das Subtrahieren durch Addition des Zweierkomplements (vierstellige Binärzahlen):

6 – 6 = 6 + (-6) = 0110 + 1010 = [1]0000 = 0
3 – 3 = 3 + (-3) = 0011 + 1101 = [1]0000 = 0
5 – 1 = 5 + (-1) = 0101 + 1111 = [1]0100 = 4
1 – 5 = 1 + (-5) = 0001 + 1011 = 1100 = -4

Wertebereich vorzeichenbehafteter Zahlen

wegen Abbildung des Vorzeichens
wegen Abbildung der Zahl Null

z.B. n = 16 Bits: a) Größte positive Zahl → 0111 1111 1111 1111 …… $2^{15}-1 = 32767 \Rightarrow 2^{(n-1)}-1$

b) (Betragsmäßig) größte negative Zahl (d.h. kleinster Zahlenwert) →

→ 1000 0000 0000 0000

Einerkomplement: 0111 1111 1111 1111

Zweierkomplement: 1000 0000 0000 0000 …… 32768 → -32768 ⇒ $-2^{(n-1)}$

c) Gesamter Wertebereich: $-2^{(n-1)}$ …. $2^{(n-1)}-1$, d.h. -32768 …. +32767

(der mögliche Zahlenbereich von 65536 wird aufgeteilt auf 32768+1+32767 Werte) .

Allgemein formuliert liegt der abbildbare Wertebereich daher im Intervall:

$$\mathbf{[-2^{(n-1)}, 2^{(n-1)}-1]}$$

3.2.2.3. Rechenoperationen im Computer

Eine Subtraktion wird durch Komplementaddition ersetzt; die Multiplikation durch mehrere Additionen; eine Divison durch mehrere Komplementadditionen. [Binäre Multiplikation, bzw. Division, wird mittels Verschiebeoperationen ausgeführt.] ⇒

⇒ Ein Computer muss daher als Basisoperationen nur Addieren und Komplementbildung sowie das Verschieben (von Bits) beherrschen.

3.2.3. Gleitkommadarstellung

Um sehr große und sehr kleine Zahlenwerte auf gleiche Weise darstellen zu können, wird bei der Gleitkommadarstellung (engl.: floating point representation) gemeinsam mit der Zahl ein variabler Skalierungsfaktor abgespeichert.
Eine Zahl läßt sich bei dieser Darstellungsform auf eine feste Anzahl signifikanter Stellen genau darstellen; der Zahlenwert wird durch den Skalierungsfaktor festgelegt.

Z. B.: $12{,}34 = \underbrace{1234}_{\text{Ziffernfolge} \rightarrow \text{MANTISSE } \mathbf{m}} * 10^{\boxed{-2}}$ EXPONENT **e** (des Skalierungsfaktors) → Exponent → lat.: ‚Herausgehobener' ..... hier: die Hochzahl

Ziffernfolge → MANTISSE **m** → Mantisse → lat.: Anhängsel ..... hier: die Ziffern hinter dem Komma

Die Ziffernfolge der Zahl wird als **Mantisse** bezeichnet; der Skalierungsfaktor, der den Zahlenwert bestimmt, wird als **Exponent** bezüglich des verwendeten Zahlensystems angegeben.

⇒ Zahlen werden hier somit als $Z = m * B^e$ dargestellt [B^e ..... Skalierungsfaktor]
{**Z**ahlenwert = **M**antisse x **B**asis hoch **E**xponent}

Gespeichert wird die Mantisse m als ganze Zahl (‚1234') und der Exponent e als ganze Zahl (‚-2'); die Basis wird nicht gespeichert.
Sowohl die Mantisse wie auch der Exponent können negativ sein, d.h., es müssen zwei Vorzeichen berücksichtigt und in geeigneter Form mitgespeichert werden.

z.B.: $-0{,}000123 = -123 * 10^{-6}$

Sowohl für die Mantisse als auch für den Exponenten stehen jeweils nur eine beschränkte Anzahl von Speicherstellen (d.h. von Bits) zur Verfügung;
die Mantisse legt also die Genauigkeit fest und der Exponent die Größe des Wertebereiches.

Im Folgenden werden sowohl der Wert der Mantisse als auch deren Länge mit m bezeichnet, gleiches gilt für den Exponenten e .

Ein und dieselbe Zahl kann auf verschiedene Weisen dargestellt werden:

$$12{,}34 = 1234 * 10^{-2} = 0{,}1234 * 10^{2}$$

Um eine eindeutige Darstellung zu erhalten, wird die **<u>normalisierte Gleitkommadarstellung</u>** eingeführt: ⇒ *Der Skalierungsfaktor wird so gewählt, daß die betrachtete Zahl als rein gebrochenzahliger Wert dargestellt wird;*
d.h., der ganzzahlige Anteil am Zahlenwert ist Null, die <u>erste Stelle nach dem Dezimalpunkt ist eine von Null verschiedene Ziffer</u> (außer bei der Zahl 0 selbst).

Für die Mantisse gilt hier somit: $^{1}/_{B} \leq m < 1$ [Beim Zahlenwert Null ist die Mantisse jedoch gleich Null.]

..... im Dezimalsystem daher: $0{,}1 \leq m < 1$, bzw. $-1 < m \leq -0{,}1$
..... und im Binärsystem: $½_{10} \leq m < 1$, bzw. $-1 < m \leq -½_{10}$

z.B.: $123{,}47 = 0{,}12347 * 10^{3}$
$11010{,}11_2 = 0{,}1101011_2 * 2^{5}$

In einem digitalen Rechner brauchen daher nur Mantisse und Exponent sowie die Vorzeichendarstellung, aber kein Komma (Dezimalpunkt) abgespeichert werden.

Folgende Speicherdarstellung wird oft gewählt:

- Es wird die normalisierte Gleitkommadarstellung verwendet.
- Die Mantisse wird mit getrenntem Vorzeichenbit gespeichert → $n=m+e+VZ_m$
- Der Exponent wird zur Darstellung negativer Werte im Zweierkomplement gespeichert.

An der höchstwertigsten Stelle steht das Vorzeichenbit der Mantisse, dann folgt der Exponent und danach die Mantisse.
Dadurch kann ein Vergleich zweier Gleitkommazahlen bitweise von links nach rechts erfolgen.

Z. B.:

VZ_m	e	e	e	e	e	m	m	m	m	m	m	m	m	m	m

VZ_m .... Vorzeichenbit (-stelle) der Mantisse
e .... Exponentenbit (-stelle) [Exponentendarstellung im Zweier-Komplement]
m .... Mantissenbit (-stelle) [Mantissendarstellung in normalisierter Form]

Vorgehensweise beim Umwandeln einer Zahl in eine normalisierte Gleitkommadarstellung:

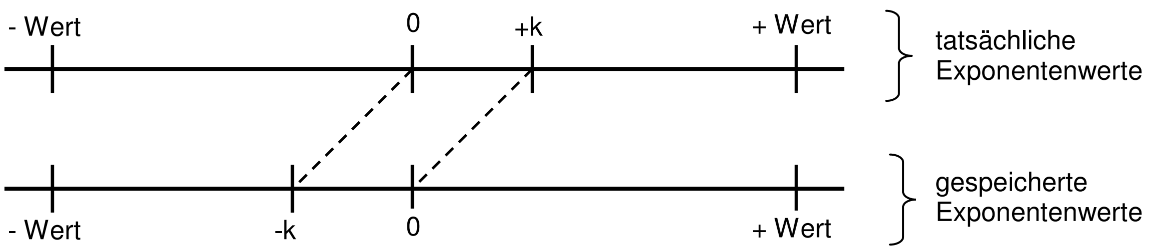
1. Positiven Wert der Zahl in eine Binärzahl umwandeln.

2. Diese Binärzahl in eine normalisierte Darstellung überführen (durch entsprechende Wahl des Exponenten):

- **a)** falls Exponent zu groß ist, kann die Zahl nicht dargestellt werden → siehe S. 36
- **b)** Exponent ggf. mit (führenden) Nullen ergänzen
- **c)** Bei negativem Exponent den positiven Wert des Exponenten in das Zweierkomplement umwandeln
- **d)** Bei negativem Zahlenwert das Vorzeichenbit der Mantisse auf ‚1' setzen
- **e)** Mantisse ggf. abschneiden oder mit (nachlaufenden) Nullen ergänzen

Charakteristik

Oft werden betragsmäßig große Zahlen und somit große positive Exponenten häufiger benötigt als kleine Zahlen {die größer als Null und kleiner als Eins sind, also mit negativen Exponenten}.
Um dafür den darstellbaren Zahlenbereich zu vergrößern, wird der Exponent als vorzeichenlose Zahl aufgefasst und eine Konstante k – die Charakteristik – zum tatsächlichen Exponenten hinzuaddiert (und dieser Wert gespeichert).
Daraus ergibt sich, dass der intern gespeicherte Wert Null dem Wert von –k entspricht und der Exponent Null als Wert k abgespeichert wird.



Die Darstellung einer Zahl mit Charakteristik k wird oft auch als k-Exzess, Exzess-k oder einfach als Exzessdarstellung bezeichnet.

Einsparung eines Mantissenbits

Bei der normalisierten Gleitkommadarstellung von Binärzahlen ist das erste Bit der Mantisse vereinbarungsgemäß immer ‚1'. Dieses Bit kann beim Speichern eingespart werden, wodurch sich die Genauigkeit erhöht.
Dabei ist zu beachten, dass die Zahl Null dann nicht mehr dargestellt werden kann (alle Bits wären Null). Um dieses Problem zu lösen, wird die Bitkombination für die kleinste darstellbare Zahl als Null aufgefasst.

IEEE-Format

Dieses vom „Institute of Electrical and Electronic Engineers" international genormte Format liefert eine Darstellung auf 32 Bit mit e=8 Stellen, k=127 und die Mantisse normalisiert auf 1.xxx mit 23 Nachkommastellen bei Einsparung der führenden Vorkommastelle.
[Eine analoge Normierung für doppelt lange Gleitkommazahlen existiert mit 64 Bit und e=11.]

3.2.4. BCD-(Binary Coded Decimal-)Darstellung

Bei der dezimalen Darstellung wird jede dezimale Ziffer einzeln in ihrem binären Äquivalent gespeichert.
Diese Darstellung wird gewählt, wenn Rundungsfehler bei der Abspeicherung vermieden werden sollen oder eine unbegrenzte Genauigkeit benötigt wird.

z. B.:

Problem dabei: Wie erkennt man die einzelnen dezimalen Ziffern, wenn die Bits hintereinander abgespeichert werden?
Lösung: Immer eine bestimmte Anzahl von Bits pro Dezimalziffer verwenden, z. B. immer 4 Bits als eine Tetrade.

z. B.:

Problem weiters: Wie erkennt man das Ende der Ziffernfolge?
Lösung: Die Länge der Zahl (d.h., die Anzahl der Ziffern) und auch das Vorzeichen extra abspeichern – also zusätzlich zu den Ziffern der Zahl.

3.2.4.1. Ungepackte (gezonte) Darstellung

Jede Dezimalziffer wird in einem Byte dargestellt.
Vorzeichen und Länge der Zahl wird in je einem zusätzlichen Byte abgelegt.

Darstellung:	1. Byte	2. Byte	3. – n-tes Byte
	Anzahl Ziffern	Vorzeichen (VZ)	Ziffernfolge

z. B.:

Vorteile:
*) einfache, nämlich direkte Ein- und Ausgabe der Zahlen möglich
*) beliebige Genauigkeit erreichbar (Zahlenbereich ist nicht eingeschränkt)

Nachteile:
-) großer Speicherbedarf
-) Vorzeichen und Länge einer Zahl müssen extra abgespeichert werden
-) arithmetische Operationen sind aufwändiger, da die Zahlen vorher umgewandelt werden müssen

3.2.4.2. Gepackte Darstellung

Prinzipiell wird die gleiche Darstellung verwendet wie bei der ungepackten Darstellung, hier wird jedoch eine Ziffer in genau einer Tetrade gespeichert (d.h., zwei Ziffern je Byte).

z. B.:

3.2.4.3. Gebrochene Zahlen

Prinzipiell sind wieder die beiden Varianten – gepackte oder ungepackte Darstellung – möglich; zusätzlich muss noch die Anzahl der Nachkommastellen gespeichert werden.

Darstellung:	1. Byte	2. Byte	3. Byte	4. – n-tes Byte
	Anzahl Ziffern	Anz. Nachkomma-	Vorzeichen (VZ)	Ziffernfolge