Binärdarstellung von Zahlen

Theorie

Inhaltsverzeichnis

1 Zahlensysteme

1.1 Das Dezimalsystem

Das Dezimalsystem ist ein Stellenwertsystem (Positionssystem) zur Basis 10. Das bedeutet, dass eine Ziffer neben ihrem eigenen Wert noch einen Wert erhält, der durch ihre Position innerhalb der Zahl gegeben ist.

Beispiel 1.1

Wie bildet man eine Zahl aus ihren Ziffern?

Dezimalziffer	6	9	3
Stellenwert	10^2	10^1	10^0
	100	10	1

$693 = 3 \cdot 10^0 + 9 \cdot 10^1 + 6 \cdot 10^2$

Beispiel 1.2

Wie gewinnt man die Ziffern aus einer Zahl?

$$\begin{array}{rcrcrcr} 693 & : & 10 & = & 69 & \text{Rest} & 3 \\ 69 & : & 10 & = & 6 & \text{Rest} & 9 \\ 6 & : & 10 & = & 0 & \text{Rest} & 6 \end{array}$$

1.2 Das Binärsystem

Das Binärsystem ist ein Stellenwertsystem zur Basis 2.

Bekanntlich werden Zahlen im Dezimalsystem aus den zehn Ziffern 0 bis 9 gebildet. Entsprechend werden Zahlen im Binärsystem aus den zwei Ziffern 0 und 1 gebildet.

Wenn nicht aus dem Kontext hervorgeht, welche Basis einer Zahl zu grunde liegt, kennzeichnet man sie mit einem entsprechenden Index.

Beispiele: 101_{10}, 101_2 oder 235_6.

1.2.1 Umrechnung vom Binär- ins Dezimalsystem

Beispiel 1.3

Analog zum Beispiel 1.1 erhalten wir:

$1011_2 = 1 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 1 \cdot 2^3 = 1 + 2 + 8 = 11_{10}$

Beispiel 1.4

Rechne die Binärzahl 10000101_2 ins Dezimalsystem um.

$10000101_2 = 1 \cdot 1^0 + 1 \cdot 2^2 + 1 \cdot 2^7 = 1 + 4 + 128 = 133$

1.2.2 Umrechnung vom Dezimal- ins Binärsystem

Beispiel 1.5

Die Umrechnung erfolgt analog zum Beispiel 1.2.

293	:	2	=	146	Rest 1
146	:	2	=	73	Rest 0
73	:	2	=	36	Rest 1
36	:	2	=	18	Rest 0
18	:	2	=	9	Rest 0
9	:	2	=	4	Rest 1
4	:	2	=	2	Rest 0
2	:	2	=	1	Rest 0
1	:	2	=	0	Rest 1
0	:	2	=	0	Rest 0

Reste von unten nach oben gelesen: $293_{10} = 100100101_2$

Beispiel 1.6

Welche Darstellung hat die Dezimalzahl 47 im Binärsystem?

47	:	2	=	23	Rest 1
23	:	2	=	11	Rest 1
11	:	2	=	5	Rest 1
5	:	2	=	2	Rest 1
2	:	2	=	1	Rest 0
1	:	2	=	0	Rest 1

$47 = 101111_2$

Beispiel 1.7

Welche Darstellung hat die Dezimalzahl 148 im Binärsystem?

$$\begin{array}{rcrcrcr} 148 & : & 2 & = & 74 & \text{Rest} & 0 \\ 74 & : & 2 & = & 37 & \text{Rest} & 0 \\ 37 & : & 2 & = & 18 & \text{Rest} & 1 \\ 18 & : & 2 & = & 9 & \text{Rest} & 0 \\ 9 & : & 2 & = & 4 & \text{Rest} & 1 \\ 4 & : & 2 & = & 2 & \text{Rest} & 0 \\ 2 & : & 2 & = & 1 & \text{Rest} & 0 \\ 1 & : & 2 & = & 0 & \text{Rest} & 1 \end{array}$$

$148 = 10010100_2$

1.3 Das Hexadezimalsystem

Für das (Sechzehnersystem) benötigen wir sechzehn Ziffern. Da wir im Dezimalsystem aber nur zehn Ziffern zur Verfügung haben, verwenden wir für die fehlenden Ziffern die ersten sechs Buchstaben unseres Alphabets.

10er-System	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16er-System	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Gross- und Kleinschreibung wird nicht unterschieden.

In der Informatik werden Hexadezimalzahlen zur Kennzeichnung das Präfix `0x` oder dass Suffix h beigefügt. ($\texttt{1A53}_{16} = \texttt{0x1A53} = \texttt{1A53h}$)

1.3.1 Umrechnung vom Hexadezimal- ins Dezimalsystem

Beispiel 1.8

Hexadezimalzahl	1	5	E
Stellenwert	$16^2 = 256$	$16^1 = 16$	$16^0 = 1$

$15\text{E}_{16} = 14 \cdot 1 + 5 \cdot 16 + 1 \cdot 256 = 350_{10}$

1.3.2 Umrechnung vom Dezimal- ins Hexadezimalsystem

Beispiel 1.9

$$\begin{array}{rcrcrcr} 1610 & : & 16 & = & 100 & \text{Rest} & \text{A} \\ 100 & : & 16 & = & 6 & \text{Rest} & 4 \\ 6 & : & 16 & = & 0 & \text{Rest} & 6 \end{array}$$

$1610_{10} = 64\text{A}_{16}$

1.3.3 Umrechnungen vom Hexadezimal- ins Binärsystem

Beispiel 1.10

Da auch das Hexadezimalsystem die Basis 2 hat, ist das Umrechnen zwischen diesen Systemen einfach. Da jede Hexadezimalzahl durch genau vier Binärziffern dargestellt werden kann, teilen wir jede Binärzahl von rechts nach links in Vierergruppen ein. Fehlende Stellen links werden durch Nullen aufgefüllt. Dann wandelt man jede Vierergruppe in die entsprechende Hexadezimalzahl um.

Beispiel 1.11

Wandle die Binärzahl 1111001001_2 in eine Hexadezimalzahl um:

0	0	1	1	1	1	0	0	1	0	0	1
3				C				9			

Beispiel 1.12

Stelle die Hexadezimalzahl F4E7 als Binärzahl dar:

F				4				E				7			
1	1	1	1	0	1	0	0	1	1	1	0	0	1	1	1

1.4 Das Oktalsystem

Das Oktalsystem ist ist das Zahlensystem zur Basis 8, das aus den Ziffern 0, 1, . . . , 6, 7 besteht.

Oktalzahlen werden in der Informatik manchmal durch eine vorangestellte Null oder ein nachgestelltes kleines „o“ gekennzeichnet, was jedoch leicht zu Verwechslungen führen kann.

Beispiel: $371_8 = 0371 = 371o$

1.4.1 Umrechnung vom Oktal- ins Dezimalsystem

Beispiel 1.13

Oktalziffer	1	0	2	7
Stellenwert	8^3	8^2	8^1	8^0
	512	64	8	1

$1027_8 = 7 \cdot 1 + 2 \cdot 8 + 0 \cdot 64 + 1 \cdot 512 = 535$

1.4.2 Umrechnung vom Dezimal- ins Oktalsystem

Beispiel 1.14

$$\begin{array}{rcrcrcl} 843 & : & 8 & = & 105 & \text{Rest} & 3 \\ 105 & : & 8 & = & 13 & \text{Rest} & 1 \\ 13 & : & 8 & = & 1 & \text{Rest} & 5 \\ 1 & : & 8 & = & 0 & \text{Rest} & 1 \end{array}$$

$843_{10} = 1513_8$

Beispiel 1.15

Rechne 473_8 ins Dezimalsystem um:

$473_8 = 3 \cdot 1 + 7 \cdot 8 + 4 \cdot 64 = 315$

Beispiel 1.16

Rechne 1217_{10} ins Oktalsystem um:

$$\begin{array}{rcrcrcl} 1217 & : & 8 & = & 152 & \text{Rest} & 1 \\ 152 & : & 8 & = & 19 & \text{Rest} & 0 \\ 19 & : & 8 & = & 2 & \text{Rest} & 3 \\ 2 & : & 8 & = & 0 & \text{Rest} & 2 \end{array}$$

$1217_{10} = 2301_8$

1.4.3 Umrechnungen zwischen Oktal- und Binärsystem

Da die Basis 8 des Oktalsystems eine Potenz der Basis 2 des Binärsystems ist, sind die beiden Systeme in einer gewissen Weise miteinander verwandt. Das erleichtert das Umrechnen zwischen diesen Systemen.

Wollen wir eine Binärzahl *direkt* ins Oktalsystem umwandeln teilen wir ihre Ziffern von rechts nach links in Dreiergruppen ein. Fehlende Stellen links füllen wir mit Nullen auf. Da $001_2 = 1_8$, $010_2 = 2_8$, …, $111_2 = 7_8$, kann man jede Dreiergruppe aus Binärzählen in die entsprechende Oktalzahl umwandeln.

Beispiel 1.17

0	1	0	1	1	0	0	1	1
2			6			3		

Beispiel 1.18

Die Umrechnung in die andere Richtung ist ebenso einfach. Rechne als Beispiel die Oktalzahl 3756_8 in eine Binärzahl um.

3			7			5			6		
0	1	1	1	1	1	1	0	1	1	1	0

Beispiel 1.19

Stelle 11110101010_2 im Oktalsystem dar:

0	1	1	1	1	0	1	0	1	0	1	0
3			6			5			2		

$011110101010_2 = 3652_8$

Beispiel 1.20

Stelle 1037_8 im Binärsystem dar:

$001|000|011|111_2$

2 Ganze Zahlen in Binärdarstellung

2.1 Bitwertigkeit

Um Missverständnisse bei der Darstellung binärer Zahlen zu vermeiden, vereinbaren wir folgende Begriffe:

- Das Bit, das die kleinste Zweierpotenz repräsentiert, wird *least significant bit* (LSB) genannt.
- Das Bit, das die grösste Zweierpotenz repräsentiert, wird *most significant bit* (MSB) genannt.

Da in unserer Zahlendarstellung die Stellenwerte von rechts nach links aufsteigen, vereinbaren wir, dass das LSB jeweils ganz rechts steht. Dies wird schematisch so dargestellt:

7							0
0	1	0	0	1	0	1	1

2.2 Addition

Allgemein gilt:

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 0$ Übertrag 1

Der Ablauf ist wie im Dezimalsystem: von rechts nach links, mit Übertrag. Führende Leerstellen werden wie Nullen behandelt.

Beispiel 2.1

	0	0	1	1	1	=	7
+	0	1	1	0	1	=	13
=	1	0	1	0	0	=	20

2.3 Negative ganze Zahlen

Ansatz: Stelle der Binärdarstellung ein zusätzliches Bit voran, wobei beispielsweise `1` eine negative und `0` eine positive Zahl bedeutet.

$0011_2 = 3_{10}$
$1011_2 = -3_{10}$

Wie wir aus der 7. Klasse wissen, muss man beim Rechnen mit Vorzeichen „mühsame“ Fallunterscheidungen beachten.

Deshalb verwendet man eine andere Darstellung für negative Zahlen, mit der Computer einfacher und schneller rechnen können.

Diese Darstellungsweise soll nun vorgestellt werden.

Stelle die ganzen Zahlen von 0 bis 7 im Binärsystem mit führenden Nullen dar. Setze anschliessend die Tabelle in „natürlicher“ Weise in den Bereich der negativen Zahlen fort.

7	0	1	1	1
6	0	1	1	0
5	0	1	0	1
4	0	1	0	0
3	0	0	1	1
2	0	0	1	0
1	0	0	0	1
0	0	0	0	0

-1	1	1	1	1
-2	1	1	1	0
-3	1	1	0	1
-4	1	1	0	0
-5	1	0	1	1
-6	1	0	1	0
-7	1	0	0	1
-8	1	0	0	0

Eigenschaften

- Das Bit ganz links kann als Vorzeichenbit interpretiert werden.
- Die Bitsumme der Zahlen k und $-(k+1)$ ergibt immer ein Bitmuster aus lauter Einsen.

  Beispiel:
  $$\begin{array}{rrcr} & 0100 & = & 4 \\ + & 1011 & = & -5 \\ \hline & 1111 & = & -1 \end{array}$$

- Die Bitsumme der Zahlen k und $-k$ ergibt immer ein Bitmuster aus Nullen und einer einer Eins (vom Übertrag).

  Beispiel:
  $$\begin{array}{rrcr} & 0100 & = & 4 \\ & 1100 & = & -4 \\ \hline & 10000 & = & 0 \end{array}$$

Das Einerkomplement

Das *Einerkomplement* $\overline{A}$ einer Binärzahl A erhält man durch „Flippen“ jedes Bits von A.

Beispiel 2.2

$$\begin{array}{rcccccccc} A & = & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ \overline{A} & = & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ \hline A+\overline{A} & = & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$$

Allgemein gilt für eine n-stelligen Binärzahl A und ihr Einerkomplement $\overline{A}$:

$$A + \overline{A} = 2^n - 1$$

Das Zweierkomplement

Das *Zweierkomplement* einer Binärzahl A ist der um 1 vergrösserte Wert des Einerkomplements $\overline{A}$.

Beispiel 2.3

$$\begin{array}{rcccccccccl} A & = & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & \\ \overline{A} & = & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & \text{(Einerkomplement)} \\ \overline{A}+1 & = & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & \text{(Zweierkomplement)} \end{array}$$

Allgemein gilt für eine n-stelligen Binärzahl A und ihr Zweierkomplement $\overline{A}+1$:

$$A + \left(\overline{A}+1\right) = 2^n$$

Zusammenfassung

- Im n-stelligen Binärsystem der ganzen Zahlen stellt das Zweierkomplement der Zahl k die Gegenzahl $-k$ dar.
- Die Summe von k und $-k$ ergibt 2^n.
- Die Summe von k und $-k$ ergibt 0, wenn wir den Übertrag ignorieren.

Beispiel 2.4

Bestimme die Gegenzahl von 3 wenn vier Bit für die Zahlendarstellung zur Verfügung stehen:

$$\begin{array}{rcccccc} A & 0 & 0 & 1 & 1 & = & +3 \\ \overline{A} & 1 & 1 & 0 & 0 & = & -4 \\ \overline{A}+1 & 1 & 1 & 0 & 1 & = & -3 \end{array}$$

Beispiel 2.5

Bestimme die Gegenzahl von -5 wenn vier Bit für die Zahlendarstellung zur Verfügung stehen:

$$\begin{array}{rcccccc} A & 1 & 0 & 1 & 1 & = & -5 \\ \overline{A} & 0 & 1 & 0 & 0 & = & 4 \\ \overline{A}+1 & 0 & 1 & 0 & 1 & = & 5 \end{array}$$

Die Berechnung der Gegenzahl funktioniert offenbar auch, wenn man von einer negativen ganzen Zahl ausgeht.

2.4 Subtraktion

Da wir jetzt einem Methode kennen, mit der man aus der Binärdarstellung der Zahl k ihre (binäre) Gegenzahl $-k$ bestimmen kann, können wir die Subtraktion einer Zahl k als Addition ihrer Gegenzahl $-k$ darstellen. Formal:

$$m - k = m + (-k)$$

Subtraktion mit positivem Ergebnis

Berechne $6 - 4 = 6 + (-4)$:

	0	1	1	0	=	6
+	1	1	0	0	=	−4
=	0	0	1	0	=	2

So lange das Resultat innerhalb des Darstellungsbereiches (hier von -8 bis $+7$) liegt, darf eine allfällige Übertrags-Eins ganz links ignoriert werden.

Subtraktion mit negativem Ergebnis

Berechne $4 - 7 = 4 + (-7)$:

	0	1	0	0	=	4
+	1	0	0	1	=	−7
=	1	1	0	1	=	−3

Addition von zwei negativen Zahlen

Berechne $-2 + (-3)$:

	1	1	1	0	=	−2
+	1	1	0	1	=	−3
=	1	0	1	1	=	−5

2.5 Multiplikation

Allgemein gilt:

$$\begin{aligned} 0 \cdot 0 &= 0 \\ 0 \cdot 1 &= 0 \\ 1 \cdot 0 &= 0 \\ 1 \cdot 1 &= 1 \end{aligned}$$

Beispiel 2.6

Die Multiplikation zweier Binärzahlen besteht aus einer fortgesetzten Addition unter Stellenverschiebung; hier gezeigt an der Rechnung $13 \cdot 10$.

1	1	0	1	·	1	0	1	0
					1	0	1	0
				0	0	0	0	–
			1	0	1	0	–	–
		1	0	1	0	–	–	–
	1	0	0	0	0	0	1	0

Beispiel 2.7

1	0	·	1	0	1	1	$(2 \cdot 11)$
			0	0	0	0	
		1	0	1	1	–	
		1	0	1	1	0	(22)

Moral: Im binären Zahlensystem bedeutet eine Multiplikation mit dem dezimalen Wert 2 das Anhängen einer Null rechts vom LSB.

2.6 Division

Die Division binärer Zahlen wird auf eine fortgesetzte Subtraktion unter Stellenverschiebung zurückgeführt.

Beispiel 2.8

	1	0	0	0	0	0	1	:	1	1	0	1	=	1	0	1
–		1	1	0	1	↓	↓									
				1	1	0	1									
			–	1	1	0	1									
							0									

Beispiel 2.9

Binäre Division von 22 : 2:

$$
\begin{array}{rccccccccccc}
 & 1 & 0 & 1 & 1 & 0 & : & 1 & 0 & = & 1 & 0 & 1 & 1 \\
- & 1 & 0 & \downarrow & \downarrow & & & & & & & & & \\
\hline
 & & 0 & 1 & 1 & & & & & & & & & \\
 & & - & 1 & 0 & \downarrow & & & & & & & & \\
\hline
 & & & & 1 & 0 & & & & & & & & \\
 & & & - & 1 & 0 & & & & & & & & \\
\hline
 & & & & & 0 & & & & & & & &
\end{array}
$$

Moral: Im Binärsystem bedeutet eine Division durch 2_{10} das Verschieben aller Binärstellen um eine Stelle nach rechts und löschen des LSB.

Problem

Bei der ganzzahligen Division einer ungeraden Zahl durch 2 entsteht ein Rest.

3 Binärdarstellung von Dezimalzahlen

3.1 Binärdarstellung von Zahlen zwischen 0 und 1

Beispiel 3.1

Bestimme die Binärdarstellung von 0.8125 durch Probieren:

n	2^{-n}	Bit	kumuliert
1	0.5	1	0.5
2	0.25	1	0.75
3	0.125	0	0.75
4	0.0625	1	0.8125

$0.8125 = 0.1101_2$ (von oben nach unten gelesen!)

algorithmisch:

$$
\begin{array}{lcccl}
2 & \cdot & 0.8125 & = & 1 \quad \mathrm{R} \quad 0.625 \\
2 & \cdot & 0.625 & = & 1 \quad \mathrm{R} \quad 0.25 \\
2 & \cdot & 0.25 & = & 0 \quad \mathrm{R} \quad 0.5 \\
2 & \cdot & 0.5 & = & 1 \quad \mathrm{R} \quad 0 \\
2 & \cdot & 0 & = & 0 \quad \mathrm{R} \quad 0 \\
2 & \cdot & \ldots & = & \ldots \quad \mathrm{R} \quad \ldots
\end{array}
$$

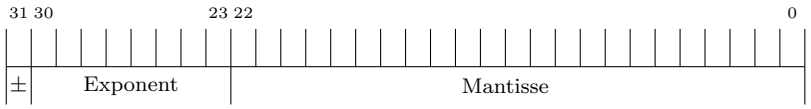
Beispiel 3.2

Bestimme die Binärdarstellung von 0.1:

$$
\begin{array}{lcccl}
2 & \cdot & 0.1 & = & 0 \quad \mathrm{R} \quad 0.2 \\
2 & \cdot & 0.2 & = & 0 \quad \mathrm{R} \quad 0.4 \\
2 & \cdot & 0.4 & = & 0 \quad \mathrm{R} \quad 0.8 \\
2 & \cdot & 0.8 & = & 1 \quad \mathrm{R} \quad 0.6 \\
2 & \cdot & 0.6 & = & 1 \quad \mathrm{R} \quad 0.2 \\
2 & \cdot & 0.2 & = & 0 \quad \mathrm{R} \quad 0.4 \\
2 & \cdot & 0.4 & = & 0 \quad \mathrm{R} \quad 0.8 \\
2 & \cdot & \ldots & = & \ldots \quad \mathrm{R} \quad \ldots
\end{array}
$$

$0.1 = 0.000110011001100\ldots_2 = 0.000\overline{1100}_2$

3.2 Gleitkommazahlen im IEEE 754-Standard



Vorzeichen (Bit 31)

S (*sign*) ist das Vorzeichenbit.

$S = 0$ markiert eine positive Zahl; $S = 1$ eine negative Zahl.

Für die Null erlaubt der Standard sowohl ein positives als auch ein negatives Vorzeichen.

Exponent (Bits 23–30)

Mit 8 Bits lassen sich $2^8 = 256$ Exponenten darstellen. Jedoch sind 0 und 255 für spezielle Zahlen reserviert und können nicht als Exponenten verwendet werden.

Negative Exponenten werden durch Addition von $B = 127$ (*bias*) positiv gemacht und dann binär dargestellt.

Mantisse (Bits 0–22)

Die Mantisse ist die Ziffernfolge einer Zahl. Die Dezimalzahlen 0.002357 und 235.7 haben beispielsweise dieselbe Mantisse 2357.

Die Binärzahl wird so lange mit einer Zweierpotenz multipliziert oder dividiert, bis bis die führende 1 vor dem Dezimalpunkt steht. Diesen Vorgang nennt man *Normalisieren*.

Binärzahl	Normalform	Mantisse
1101.01	$1.10101 \cdot 2^3$	(1)10101
0.00101	$1.01 \cdot 2^{-3}$	(1)01

Indem man die durch das Normalisieren zwangsläufig entstehende führende 1 weglässt, kann man ein Bit Speicher sparen.

3.3 Umrechnung Dezimalzahl ins IEEE 754-Format

Beispiel 3.3

Welche IEEE 754-Darstellung hat die Zahl 5.75?

Vorzeichenbit

Wegen $5.75 > 0$ ist $S = 0$

Mantisse

Den ganzzahligen und den gebrochenen Anteil binär darstellen:

ganzzahliger Anteil:

$$\begin{array}{rcrcrcl} 5 & : & 2 & = 2 & \text{R} & 1 \\ 2 & : & 2 & = 1 & \text{R} & 0 \\ 1 & : & 2 & = 0 & \text{R} & 1 \end{array}$$

$5 = 101_2$ (von unten nach oben gelesen)

gebrochener Anteil:

$$\begin{array}{rcrcrcl} 2 & \cdot & 0.75 & = & 1 & \text{R} & 0.5 \\ 2 & \cdot & 0.5 & = & 1 & \text{R} & 0 \end{array}$$

$0.75 = 0.11_2$ (von oben nach unten gelesen)

Normalisierung: $5.75{=}101.11_2{=}1.0111 \cdot 2^2 \Rightarrow M = 0111$

Exponent

Addiere den Bias ($B = 127$) zum Exponenten und stelle ihn anschliessend binär dar.

$E = 2 + 127 = 129 = 10000001_2$

Resultat

$5.75 = 0|10000001|01110000000000000000000_2$

3.4 Umrechnung einer IEEE 754-Zahl in eine Dezimalzahl

Beispiel 3.4

Welcher Gleitkommazahl entspricht die Binärzahl

1|10000100|01010000000000000000000

im IEEE 754-Format?

Vorzeichen

$S = 1 \quad \Rightarrow \quad s = (-1)^1 = -1$ (negative Zahl)

Exponent

$E = 10000100_2 = 132$

$e = E - B = 132 - 127 = 5$

Mantisse

$m = 1.M = 1.0101_2$ (die weggelassene 1 voranstellen)

Für die manuelle Umrechnung ist es einfacher, wenn man die Mantisse (noch) nicht ins Zehnersystem umwandelt.

Resultat

$(-1) \cdot 1.0101_2 \cdot 2^5 = (-1) \cdot 101010_2 = (-1) \cdot (2 + 8 + 32) = -42$

3.5 Spezielle Zahlen

Die betragsmässig grösste normalisierte Zahl

Der maximale Exponent beträgt $E = 254 - 127 = 127$, da $11111111_2 = 255$ für andere Zwecke reserviert ist.

Die grösste Mantisse beträgt $M = (1)11111111111111111111111$ wenn wir wieder die Eins an die höchstwertige Stelle setzen.

Damit erhalten wir

$$\begin{aligned} & 1.11111111111111111111111 \cdot 2^{127} \\ &= 11111111111111111111111 \cdot 2^{104} \\ &\approx 3.403 \cdot 10^{38} \end{aligned}$$

als betragsmässig grösste normalisierte Zahl

Die betragsmässig kleinste normalisierte Zahl

Der minimale Exponent beträgt $1 - 127 = -126$, da $00000000_2 = 0$ für andere Zwecke reserviert ist.

Die kleinste Mantisse beträgt $M = (1)00000000000000000000000$ wenn wir wieder die Eins an die höchstwertige Stelle setzen.

Damit erhalten wir

$$1 \cdot 2^{-126} \approx 1.175 \cdot 10^{-38}$$

als betragsmässig kleinste normalisierte Zahl

Die Null

Auf der einen Seite gewinnen wir durch die Normalisierung immer eine Binärstelle mehr an Genauigkeit. Andererseits zwingt uns dies mit $m = 1.M$ immer mindestens eine 1 in der Mantisse auf, so dass die Darstellung der 0 auf diese Weise unmöglich wird.

Um die Normalisierung zu „verhindern“ wird der Exponent mit dem Wert $E = 0$ codiert und die Mantisse wird in der Form $m = 0.M$ interpretiert. Dies führt dazu, dass die Null auch als Gleitkommazahl aus lauter Nullen besteht – naja nur fast, denn es gibt auch noch die „negative“ Null, welche der Standard nicht verbietet. Somit hat die Null die folgenden Darstellung(en):

$$\begin{aligned} +0 &= 0|00000000|00000000000000000000000 \\ -0 &= 1|00000000|00000000000000000000000 \end{aligned}$$

Subnormale (denormalisierte) Zahlen

Mit $E = 00000000$ und $M = 00000000000000000000000$ wird die Null codiert. Was sollen wir nun aber mit den Mantissen

$$\begin{aligned} & 10000000000000000000000 \\ & \dots \\ & 11111111111111111111111 \end{aligned}$$

machen, wenn der Exponent $E = 00000000$ ist? Diese Werte lassen sich dazu verwenden, um Zahlen darzustellen, die zwischen Null und der kleinsten normalisierte Zahl sind. Deshalb der Ausdruck *sub*normale (oder: denormalisierte) Zahlen.

Unendlich

Nachdem wir mit dem Exponenten $E = 0$ die Zahl Null und die subnormalen Zahlen gewonnen haben, klären wir noch, was es mit dem maximalen Exponenten $E = 255$ auf sich hat.

Die kleinste mit diesem Exponenten darstellbare Mantisse $M = 0$ wird für Unendlich (`Infinity`) verwendet. Wieder gibt es zwei Formen:

`0|11111111|00000000000000000000000` = `+Infinity`

`1|11111111|00000000000000000000000` = `-Infinity`

Der Wert `+Inf` bzw. `-Inf` repräsentiert Zahlen, deren Betrag zu gross ist, um dargestellt zu werden.

Zahlen, die gar keine sind

Auch hier können wir uns fragen, was wir mit den übrigen Mantissen M zum Exponenten $E = 255$ anfangen sollen. Die Informatiker haben hier eine besondere Lösung gefunden. Mit den Mantissen $M \neq 0$ werden Ereignisse angezeigt, die es bei korrektem Rechnen nicht geben darf.

- Division durch Null
- Quadratwurzel aus einer negativen Zahl
- Logarithmus einer negativen Zahl
- Arcussinus oder Arcuscosinus einer Zahl x mit $|x| > 1$

Diese mit $E = 255$ und $M \neq 0$ codierten Objekte werden als `NaN` (*Not a Number*) bezeichnet. Der IEEE 754-Standard ignoriert dabei das Vorzeichenbit.