Zahlensysteme

Teil I: Zahlensysteme

1. Einführung und Zahlensysteme
2. Zahlensysteme / Algorithmik
3. Zahlendarstellung im Rechner

1. Einführung und Zahlensysteme

- Modell $\rightarrow$ Wirklichkeit
- Die „einfachste“ Form der Darstellung einer Zahl
- Basis-Zahlendarstellung

Modell → Wirklichkeit

- Phänomen Anzahl

10 Äpfel	$\oplus$	5 Äpfel	=	?
13 Autos	$\oplus$	18 Autos	=	?
37 Häuser	$\oplus$	12 Häuser	=	?
10 Rinder	$\oplus$	8 Rinder	=	?

Operationen Summenbildung

Differenz

Produkt

Division

Potenzen

etc.

- Zahlen

1. Frage: Wie notiert man Zahlen?
   Wie aufwendig ist das Notieren?

2. Frage: Wie addiert man Zahlen?
   Wie aufwendig ist das addieren?

Die „einfachste“ Form der Darstellung einer Zahl

„7“ ≈ | | | | | | |

„15“ ≈ | | | | | | | | | | | | | | |

Vorteil: unmittelbar verständlich

Nachteil: zu großer Aufwand bei großen Zahlen,
außerdem nicht wirklich **benennbar**,
damit praktisch nicht kommunizierbar.

Addition

- Wie addiert man solche Zahlen?

$$\overbrace{|\,|\,|\,|\,|\,|\,|\,|\,|}^{9} \quad + \quad \overbrace{|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|}^{22}$$

$$= \quad |\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|$$

Vorteil: Algorithmus trivial:
Man hängt die einen Striche an die anderen Striche.
(Algorithmus z.B. durch Wegstreichen)

Nachteil: hoher Aufwand,
Ergebnis schwer nutzbar.

Basis-Zahlendarstellung

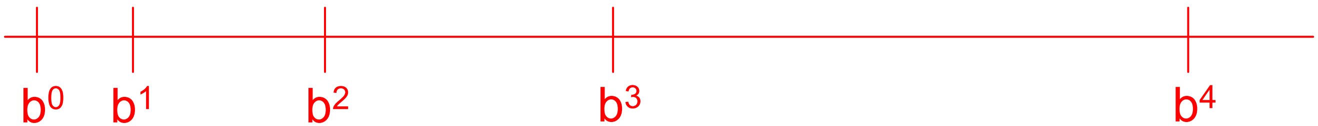
Man nehme eine Zahl b, z.B. b = 2, 3, 4, …
Betrachtet man die Zahl b^n für n = 0, 1, 2, … dann ergibt sich

$b^0 = 1$

$b^1 = b$

$b^2 = b^2$

$b^3 = b^3$



wächst exponentiell schnell
b^x wächst schneller als jedes Polynom

2^{300} ist schon größer als die Anzahl aller Atome im Universum

Mit 8-Tupeln aus 0 und 1 (0, 1, 1, 0, 0, 0, 1, 1) lassen sich schon 256 verschiedene Dinge unterscheiden bzw. beschreiben.

Basis-Zahlendarstellung (1)

$$zahl = \sum_{i=0}^{n} a_i \cdot b^i$$

- die Basis b ≥ 2 ist aus den natürlichen Zahlen
- die Ziffer a_i ist aus den natürlichen Zahlen $0 \leq a_i \leq b-1$
- die Darstellung ist eindeutig
- Schreibweise:

  $zahl = (a_n \ldots a_0)_b$

  Beispiel: $(1024)_{10} = 1 \cdot 10^3 + 0 \cdot 10^2 + 2 \cdot 10^1 + 4 \cdot 10^0$

- gebräuchliche Zahlenbasen:
  - 2 (Binär-System)
  - 8 (Oktal-System)
  - 10 (Dezimal-System)
  - 16 (Hexadezimal-System)

Beispiel

100011101

435

285

11D (Ziffern: 0, ..., 9, A, B, C, D, E, F)

Konvertierung zwischen zwei Basen

$$zahl = \sum_{i=0}^{n} a_i \cdot b^i = a_n b^n + a_{n-1} b^{n-1} + \ldots + a_1 b + a_0$$

- Basis b $\rightarrow$ Basis 10
  - Eingabe: b, Ziffernfolge a_0, a_1, ... , a_n
  - Ausgabe: Dezimalzahl

Basis 10 besonders wichtig, weil wir diese gewöhnt sind

- Basis 10 $\rightarrow$ Basis b
  - Eingabe: Dezimalzahl, neue Basis b
  - Ausgabe: Ziffernfolge a_0, a_1, ... , a_n

Basis-Zahlendarstellung (2)

Vorteile: Mit Basis-Zahlendarstellungen (Stellenwertsystemen) gelingt eine **exponentielle Verkürzung** der Darstellung gegenüber dem Strichcode.
Sie basiert im wesentlichen auf der Größe b^i
daher gibt es auch Bezeichnungen

z.B. 38 Milliarden

aktuelles Programm der US Bundesregierung zur Stärkung US Finanzsystems:
US$ 700 Milliarden

Man braucht Namen im Zehnersystem:

0, 1, 2, … , 9		
10	zehn	
100	hundert	
10^6	Million	
10^{100}	Googol	(Namensgeber der Suchmaschine „google“)

Basis-Zahlendarstellung (3)

92 im Französischen:	quatre vingt douce
	$4 \cdot 20 + 12$

- Wesentlich:
  1. Potenzen spielen eine zentrale Rolle.
  2. Man braucht die Null („0“).
     (Kulturhistorische Revolution)
  3. In größeren Systemen (z.B. im 12-er System / im 16-er System) benötigt man außerdem neue Zeichen
     (im 12-er System für 10 und 11 / im 16-er System für 10 bis 15)

2. Zahlensysteme / Algorithmik

- Summenbildung
- Addition mit Übertrag
- Multiplikation

Summenbildung

- Kann man in der kurzen Codierung eine (bzgl. der kurzen Codierung) **schnelle**, d.h. polynomische Addition hinbekommen?

Summenbildung = zentraler Algorithmus
(Subtraktion, Multiplikation, Division, Potenz, Wurzel werden darauf zurückgeführt)

Im französischen heißt die Rechnung im Lokal „l'addition"

Zentraler Algorithmus für das 10-er System:

$$
\begin{array}{r}
3\,4\,9\,7\,1\,6\,5 \\
+\quad 5\,4\,8\,2\,1\,3 \\
\hline
1\;\,1\;\,1\;\,0\;\,0\;\,0\;\; \\
\hline\hline
4\,0\,4\,5\,3\,7\,8
\end{array}
$$

Übertrag ist immer 0 oder 1

Addieren von 2 Zahlen = 3-er Addition, von denen eine nur aus 0-en und 1-en besteht

Summenbildung (1)

- Weitere Beispiele:

$$\begin{array}{r} 15723 \\ +\quad 6098 \\ \hline {\scriptstyle 1\,0\,1\,1} \\ \hline\hline 21821 \end{array}$$

$$\begin{array}{r} 48313 \\ +\quad 75688 \\ \hline {\scriptstyle 1\,1\,1\,1\,1} \\ \hline\hline 124001 \end{array}$$

Summenbildung (2)

- Addition ist also realisierbar
  über eine **Sequenz** von sogenannten **Volladdierern**
  (beinhalten den Übertrag)

- bzw. über spezielle Algorithmen, die ohne eine solche Sequenz auskommen.

(wird später genauer behandelt)

Summenbildung (3)

➢ Formale stellenweise Addition (mit Übertrag)

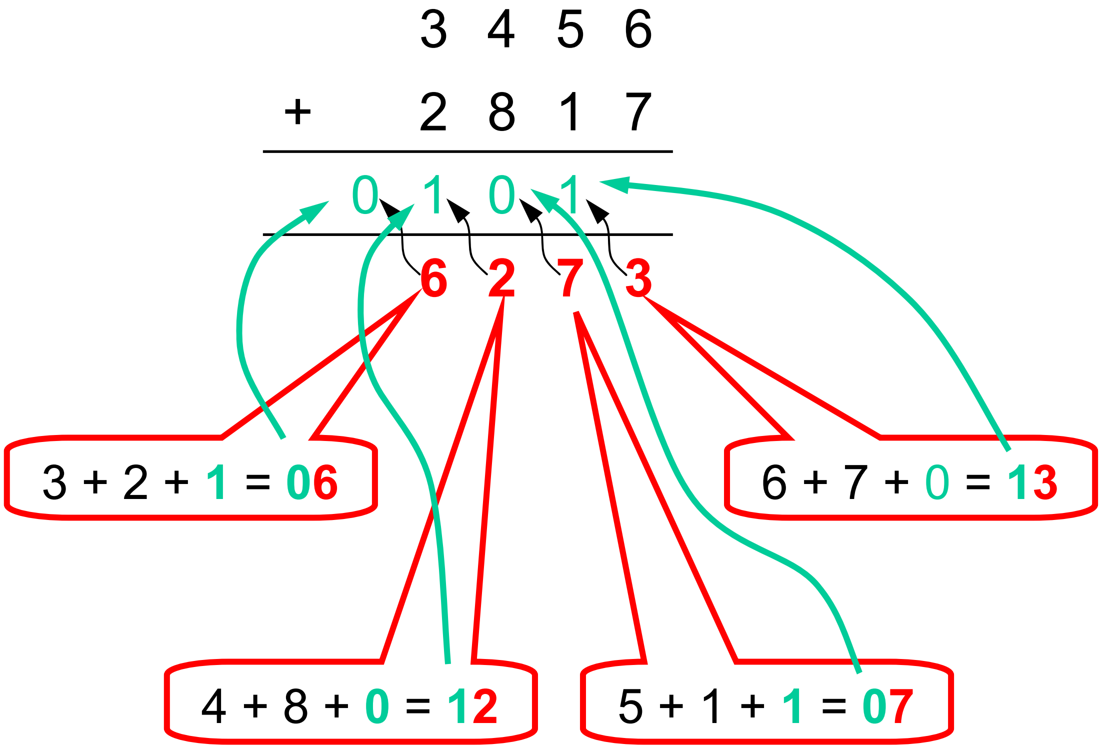
$$
\begin{array}{ccccccc|c|cc}
a_n b^n & + & a_{n-1} b^{n-1} & + & \ldots & + & & a_1 b^1 & + & a_0 b^0 \\
 & & & & & & & & & \\
a'_n b^n & + & a'_{n-1} b^{n-1} & + & \ldots & + & & a'_1 b^1 & + & a'_0 b^0 \\
\hline
 & & \ldots & & & & & \ddot{u}_0 & & \\
\hline
 & & \ldots & & & & & s_1 \cdot b^1 & + & s_0 \cdot b^0
\end{array}
$$

$$
\ddot{u}_0 = \begin{cases} 0, \text{ falls } a_0 + a'_0 < b \\ 1, \text{ falls } a_0 + a'_0 \geq b \end{cases}
$$

$$
s_0 = \begin{cases} a_0 + a'_0, \text{ falls } a_0 + a'_0 < b \\ (a_0 + a'_0) - b, \text{ falls } a_0 + a'_0 \geq b \end{cases}
$$

Summenbildung (4)

- Konkretes Beispiel:



Addition im 5-er System (1)

- Ziffern: 0, 1, 2, 3, 4

$$5_{10} = 10_5$$
$$6_{10} = 11_5$$
$$7_{10} = 12_5$$
$$8_{10} = 13_5$$
$$9_{10} = 14_5$$

größte Zahl bei der Addition zweier 5-er Zahlen, wenn noch ein Übertrag von 1 dazu kommt, da $4 + 4 (+1) = 9_{10} = 14_5$

- Additionstabelle

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	**10**
2	2	3	4	**10**	**11**
3	3	4	**10**	**11**	**12**
4	4	**10**	**11**	**12**	**13**

Die **grün** gekenn-zeichneten Fälle erzeugen einen Übertrag

Addition im 5-er System (2)

- mit Übertrag:

+ (+1)	0	1	2	3	4
0	1	2	3	4	**10**
1	2	3	4	**10**	**11**
2	3	4	**10**	**11**	**12**
3	4	**10**	**11**	**12**	**13**
4	**10**	**11**	**12**	**13**	**14**

> Die **grün** gekenn-zeichneten Fälle erzeugen einen Übertrag

5^0	=	1
5^1	=	5
5^2	=	25
5^3	=	125
	…	

Addition im 5-er System (3)

- einfache Beispiele zur Konvertierung:

$$(10)_{10} \;=\; 2\cdot \overset{5}{5^1} + 0\cdot \overset{1}{5^0} \;=\; (20)_5$$

$$(11)_{10} \;=\; 2\cdot 5^1 + 1\cdot 5^0 \;=\; (21)_5$$

$$(12)_{10} \;=\; 2\cdot 5^1 + 2\cdot 5^0 \;=\; (22)_5$$

$$(15)_{10} \;=\; 3\cdot 5^1 + 0\cdot 5^0 \;=\; (30)_5$$

$$(18)_{10} \;=\; 3\cdot 5^1 + 3\cdot 5^0 \;=\; (33)_5$$

$$\overbrace{(26)_{10}}^{\text{III} \ldots \text{III}} \;=\; 1\cdot \overset{25}{5^2} + 0\cdot \overset{5}{5^1} + 1\cdot \overset{1}{5^0} \;=\; (101)_5$$

Vorgehensweise:

25 (= 5^2) ist die größte 5-er Potenz, die gerade noch kleiner ist als die gegebene 26.

25 „passt“ genau 1 mal „in 26 hinein“.

(bisher) → $1\cdot 5^2$

Nun die $1\cdot 5^2$ (= 25) von den bisherigen 26 abziehen.
→ Es bleibt ein Rest von 1.

5 (= 5^1) „passt” genau 0 mal „in den Rest (= 1) hinein”.

(bsiher) → $1\cdot 5^2 + 0\cdot 5^1$

Nun die $0\cdot 5^1$ (= 0) vom bisherigen Rest (= 1) abziehen.
→ Es bleibt ein Rest von 1.

1 (= 50) „passt“ genau 1 mal „in den Rest (= 1) hinein”.

(bisher) → $1\cdot 5^2 + 0\cdot 5^1 + 1\cdot 5^0$

Nun die $1\cdot 5^0$ (= 1) vom bisherigen Rest (= 1) abziehen.
→ Es bleibt ein Rest von 0.

→ Wir sind fertig!

Addition im 5-er System (4)

- einfache Beispiele zur Addition:

$$(3)_5 + (4)_5 = (7)_{10}\cdot 5^0 = 1\cdot 5^1 + 2\cdot 5^0 = (12)_5$$

Umweg über das 10-er System
(eigentlich unschön – und hier
nur zur Veranschaulichung)

- direkter Weg:

$$(22)_5 + (30)_5 = ?$$

$$\begin{array}{rcccl} & & 2 & 2 & = (12)_{10} \\ + & & 3 & 0 & = (15)_{10} \\ \hline \text{Übertrag} & 1 & 0 & & \\ \hline & 1 & 0 & 2 & = (27)_{10} \end{array}$$

$$\left.\begin{array}{l}(12)_{10}\\(15)_{10}\end{array}\right\} = (27)_{10}$$

Addition im 5-er System (5)

➢ Beispiel

- Addiere $(13431)_5 + (42344)_5 = (111330)_5$

$$
\begin{array}{lr}
 & (1\ 3\ 4\ 3\ 1)_5 \\
+ & \underline{(4\ 2\ 3\ 4\ 4)_5} \\
 & 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ \ \ \\
\hline\hline
 & (1\ 1\ 1\ 3\ 3\ 0)_5
\end{array}
\rightarrow (3125)_{10} + (625)_{10} + (125)_{10} + 3\cdot(25)_{10} + 3\cdot(5)_{10} = (3965)_{10}
$$

➢ Rechnen im 5-er System

$5^0 = 1,\quad 5^1 = 5,\quad 5^2 = 25,\quad 5^3 = 125,\quad 5^4 = 625,\quad 5^5 = 3125$

- $(13431)_5 = 1\cdot(625)_{10} + 3\cdot(125)_{10} + 4\cdot(25)_{10} + 3\cdot(5)_{10} + 1\cdot(1)_{10}$
  $= 1\cdot(625)_{10} + (375)_{10} + (100)_{10} + (15)_{10} + (1)_{10} = (1116)_{10}$
- $(42344)_5 = 4\cdot(625)_{10} + 2\cdot(125)_{10} + 3\cdot(25)_{10} + 4\cdot(5)_{10} + 4\cdot(1)_{10}$
  $= (2500)_{10} + (250)_{10} + (75)_{10} + (20)_{10} + (4)_{10} = (2349)_{10}$

$$\sum = (3965)_{10}$$

Addition im 12-er System (1)

- Ziffern: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B

  $10_{10} = A_{12}$
  $11_{10} = B_{12}$
  $12_{10} = 10_{12}$
  …
  $23_{10} = 1B_{12}$

größte Zahl bei der Addition zweier 12-er Zahlen, wenn noch ein Übertrag von 1 dazu kommt, da
$B_{12} + B_{12}\ (+1) = 11_{10} + 11_{10}\ (+1)$
$1B_{12} = 23_{10}$

- Additionstabelle

+	0	1	2	3	4	5	6	7	8	9	A	B
0	0	1	2	3	4	5	6	7	8	9	A	B
1	1	2	3	4	5	6	7	8	9	A	B	**10**
2	2	3	4	5	6	7	8	9	A	B	**10**	**11**
3	3	4	5	6	7	8	9	A	B	**10**	**11**	**12**
4	4	5	6	7	8	9	A	B	**10**	**11**	**12**	**13**
5	5	6	7	8	9	A	B	**10**	**11**	**12**	**13**	**14**
6	6	7	8	9	A	B	**10**	**11**	**12**	**13**	**14**	**15**
7	7	8	9	A	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**
8	8	9	A	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
9	9	A	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
A	A	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
B	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**1A**

Die grün gekennzeichneten Fälle erzeugen einen Übertrag

Addition im 12-er System (2)

- mit Übertrag:

+ (+1)	0	1	2	3	4	5	6	7	8	9	A	B
0	1	2	3	4	5	6	7	8	9	A	B	**10**
1	2	3	4	5	6	7	8	9	A	B	**10**	**11**
2	3	4	5	6	7	8	9	A	B	**10**	**11**	**12**
3	4	5	6	7	8	9	A	B	**10**	**11**	**12**	**13**
4	5	6	7	8	9	A	B	**10**	**11**	**12**	**13**	**14**
5	6	7	8	9	A	B	**10**	**11**	**12**	**13**	**14**	**15**
6	7	8	9	A	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**
7	8	9	A	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
8	9	A	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
9	A	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
A	B	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**1A**
B	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**1A**	**1B**

Die **grün** gekenn-zeichneten Fälle erzeugen einen Übertrag

Addition im 12-er System (3)

12^0	=	1
12^1	=	12
12^2	=	144
12^3	=	1728
12^4	=	20736
12^5	=	248832
12^6	=	2985984
12^7	=	35831808
12^8	=	429981696

- Beispiel
  - Addiere $(63A5)_{12} + (5B43)_{12} = (10328)_{12}$

$$
\begin{array}{rr}
 & (6\ 3\ A\ 5)_{12} \\
+ & (5\ B\ 4\ 3)_{12} \\
\hline
 & 1\ \ 1\ \ 1\ \ 0\quad \\
\hline\hline
 & (1\ 0\ 3\ 2\ 8)_{12}
\end{array}
$$

$$
\begin{aligned}
\rightarrow\ & 1_{12}\cdot 12^4 + 0_{12}\cdot 12^3 + 3_{12}\cdot 12^2 + 2_{12}\cdot 12^1 + 8_{12}\cdot 12^0 = \\
& 1\cdot(20736)_{10} + 0\cdot(1728)_{10} + 3\cdot(144)_{10} + 2\cdot(12)_{10} + 8\cdot(1)_{10} = \\
& (20736)_{10} + (432)_{10} + (24)_{10} + (8)_{10} = \\
& (21200)_{10}
\end{aligned}
$$

- Rechnen im 12-er System
  - $(63A5)_{12} = 6_{12}\cdot 12^3 + 3_{12}\cdot 12^2 + A_{12}\cdot 12^1 + 5_{12}\cdot 12^0 =$
    $= 6\cdot(1728)_{10} + 3\cdot(144)_{10} + 10\cdot(12)_{10} + 5\cdot(1)_{10} =$
    $= (10368)_{10} + (432)_{10} + (120)_{10} + (5)_{10} = (10925)_{10}$
  - $(5B43)_{12} = 5_{12}\cdot 12^3 + B_{12}\cdot 12^2 + 4_{12}\cdot 12^1 + 3_{12}\cdot 12^0 =$
    $= 5\cdot(1728)_{10} + 11\cdot(144)_{10} + 4\cdot(12)_{10} + 3\cdot(1)_{10} =$
    $= (8640)_{10} + (1584)_{10} + (48)_{10} + (3)_{10} = (10275)_{10}$

$$\sum = (21200)_{10}$$

Multiplikation

➢ Gegeben zwei Zahlen:

„7“ $\approx$ $| \; | \; | \; | \; | \; | \; |$

„5“ $\approx$ $| \; | \; | \; | \; |$

➢ Wie erhält man das Produkt?

- mehrfaches Addieren der Zahl

$$7 \cdot 5 \quad = \quad \underbrace{5 + 5 + 5 + 5 + 5 + 5 + 5}_{\text{7 Summanden}} \quad = \quad \begin{array}{c} |\,|\,|\,|\,| \;\; |\,|\,|\,|\,| \\ |\,|\,|\,|\,| \;\; |\,|\,|\,|\,| \\ |\,|\,|\,|\,| \;\; |\,|\,|\,|\,| \\ |\,|\,|\,|\,| \end{array}$$

Multiplikation (1)

➢ Multiplikationstabelle (das kleine 1·1 im 10-er-System)

·	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	10	12	14	16	18
3	0	3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5	0	5	10	15	20	25	30	35	40	45
6	0	6	12	18	24	30	36	42	48	54
7	0	7	14	21	28	35	42	49	56	63
8	0	8	16	24	32	40	48	56	64	72
9	0	9	18	27	36	45	54	63	72	81

Multiplikation (2)

- Multiplikationstabelle (das kleine 1·1 im 5-er-System)

·	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	11	13
3	0	3	11	14	22
4	0	4	13	22	31

- Bemerkung: kleiner, einfacher

Multiplikation (3)

- Wie multipliziert man effizienter als durch mehrfache Addition?

- Beispiel: (im 10-er-System)

$$1\ 2\ 3 \quad \cdot \quad 4\ 5\ 6 \qquad = \qquad ?$$

Multiplikation (4)

$$\begin{aligned} 123 \cdot 456 &= 123 \cdot (400 + 50 + 6) \\ &= 123 \cdot 400 + 123 \cdot 50 + 123 \cdot 6 \\ &= 12300 \cdot 4 + 1230 \cdot 5 + 123 \cdot 6 \\ &= 49200 + 6150 + 738 \\ &= 56088 \end{aligned}$$

- D.h. Multiplikation „beliebiger“ Zahlen im 10-er System wird zurückgeführt auf
  - Multiplikation mit 1, 10, 100 durch Anhängen keiner, einer, zweier etc. Nullen
  - Multiplikation mit einstelligen Zahlen (← Multiplikationstabelle!)
  - Addition (← Additionstabelle!)

Multiplikation (5)

➢ Kurzdarstellung

1	2	3	·	4	5	6
		4				
			8			
			1	2		
		4	9	2	0	0
			5			
			1	0		
				1	5	
			6	1	5	0
				6		
				1	2	
					1	8
+				7	3	8
		1	1	0	0	
		5	6	0	8	8

3. Zahlendarstellung im Rechner

- Das 2-er System
- Konvertierung
- Negative Zahlen und Subtraktion
- 2-er Komplement

Das 2-er System

Vorteil: nur noch 0 und 1
2 gut identifizierbare und technisch realisierbare Zustände
(Strom fließt vs. Strom fließt nicht,
positive Spannung vs. negative Spannung)

Wie sieht die Zahl

$$\overbrace{|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|\,|}^{13}$$

im Zweier-System aus?

$$13 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 \rightarrow 1101$$

$$8 + 4 + 0 + 1$$

$$5 \rightarrow 101$$

Addition im 2-er System (1)

- Ziffern: 0, 1

$$
\begin{aligned}
0_{10} &= 000_2\\
1_{10} &= 001_2\\
2_{10} &= 010_2\\
3_{10} &= 011_2\\
4_{10} &= 100_2\\
5_{10} &= 101_2\\
6_{10} &= 110_2
\end{aligned}
$$

- Additionstabelle:

+	0	1
0	0	1
1	1	**10**

mit Übertrag:

+ (+1)	0	1
0	1	**10**
1	**10**	**11**

Die grün gekenn-zeichneten Fälle erzeugen einen Übertrag

Addition im 2-er System (2)

Summenbildung:

$$
\begin{array}{rcr}
1\,1\,0\,1 & \rightarrow & 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 \;=\; 13_{10} \\
+ \quad 1\,0\,1 & \rightarrow & 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 \;=\; 5_{10} \\
\hline
{\scriptstyle 1\;1\;0\;1\;\;} & & \\
\hline\hline
1\,0\,0\,1\,0 & \rightarrow & 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 \;=\; 18_{10}
\end{array}
$$

Konvertierung

Konvertierung vom 10-er System in 2-er System

$$(377)_{10} = (?)_2$$

377	:	2	=	188	Rest	1
188	:	2	=	94	Rest	0
94	:	2	=	47	Rest	0
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

Stop bei 0

Ergebnis: Rückwärtsanordnung der Reste, d.h.

$$(1\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 1)_2 = (377)_{10}$$

Probe, d.h. Konvertierung vom 2-er System in 10-er System

1	0	1	1	1	1	0	0	1	
256 +		64 +	32 +	16 +	8 +			1	= 377

Konvertierung (1)

Wieso werden Reste rückwärts angeordnet ?

$$(32)_{10} = (?)_2$$

32	:	2	=	16	Rest	0
16	:	2	=	8	Rest	0
8	:	2	=	4	Rest	0
4	:	2	=	2	Rest	0
2	:	2	=	1	Rest	0
1	:	2	=	0	Rest	1

$$(32)_{10} = (100000)_2$$

Endliche Zahlenbereiche (1)

Digitalrechner sind **endlich** !

- Es gibt nur endlich viele Stellen, endlich viele Plätze, … die zur Zahlendarstellung verwendet werden können. Häufig werden 32 bit, d.h. 32 Dualstellen bereitgestellt.

$$\underbrace{0/1}_{2^{31}}\ \underbrace{0/1}\ \ldots.\ \underbrace{0/1}_{2^1}\ \underbrace{0/1}_{2^0=1}$$

Kleinste Zahl $(0 \ldots 0)_2 = (0)_{10}$

Größte Zahl $(1 \ldots 1)_2 = (4\,294\,967\,295)_{10}$

Größere Zahlenbereiche als der 32 bit sind möglich

z.B. 48 bit oder 64 bit.

Endliche Zahlenbereiche (2)

Problem endlicher Zahlenbereiche:

- Summe, Produkt darstellbarer Zahlen ist u.U. nicht mehr darstellbar („Überlauf“, overflow)

  Hypothetisch kleines Beispiel: nur 4 Bits

$$\begin{array}{c|cccc}
 & 0 & 1 & 1 & 1 \\
+ & 1 & 0 & 0 & 1 \\
\hline
1 & 1 & 1 & 1 & \\
\hline
1 & 0 & 0 & 0 & 0
\end{array}$$

?

$\Rightarrow$ Summenbildung nicht ausführbar!

Negative Zahlen und Subtraktion

- Ziel: Sowohl Darstellung negativer Zahlen als auch Arithmetik damit

- Grundsätzlich 2 Wege denkbar:

  - Extra Substraktionsalgorithmus
  - Rückführung auf Addition (← bevorzugt)

Negative Zahlen und Subtraktion (1)

1. Versuch: (signed integer)

- Ein Bit wird als Vorzeichen reserviert.
- Dann sind nun weniger Bits zur eigentlichen Zahlendarstellung (Betragsdarstellung) verfügbar.
- Typischerweise wird das Bit ganz links als Vorzeichen verwendet.

  - Vorzeichenbit = 0 $\leftrightarrow$ +
  - Vorzeichenbit = 1 $\leftrightarrow$ –

- Beispiel:

$$\left.\begin{array}{l} (\,\mathbf{0}\ 1\ 1\ 1\ 0\ 0\ 1\ 0\,)_2 = 114 \\ \quad \mathbf{+}\ 64 + 32 + 16 + 8 + \quad 2 \\[2em] (\,\mathbf{1}\ 1\ 1\ 1\ 0\ 0\ 1\ 0\,)_2 = -114 \\ \quad \mathbf{-}\ 64 + 32 + 16 + 8 + \quad 2 \end{array}\right\}$$

Darstellbarkeit negativer Zahlen erscheint in Ordnung

Negative Zahlen und Subtraktion (2)

- kleines Problem: Die „0“ hat zwei Darstellungen.
  Alle anderen darstellbaren Zahlen haben eindeutige Darstellungen.

$$\underbrace{(\,0 \ldots 0\,)_2}_{\text{„+ 0“}} \quad = \quad 0 \quad = \quad \underbrace{(\,1\;0 \ldots 0\,)_2}_{\text{„– 0“}}$$

  (gewöhnungsbedürftig, aber beherrschbar)

- großes Problem: Addition einer negativen Zahl
  = Subtraktion dieser Zahl?

Negative Zahlen und Subtraktion (3)

- Beispiel: 9 – 4

	0	0001001	(= 9)
+	1	0000100	(= – 4)
	1	0001101	(= – 13)

→ so nicht

Negative Zahlen und Subtraktion (4)

- Vorzeichenbit = 1 $\leftrightarrow$ +
- Vorzeichenbit = 0 $\leftrightarrow$ –

➢ 9 – 4

$$
\begin{array}{rl}
 & 1\,0001001 \quad (= 9) \\
+ & 0\,0000100 \quad (= -4) \\
\hline
 & 1\,0001101 \quad (= +13)
\end{array}
$$

→ so auch nicht

➢ Aber: Für eine **reine Darstellung** gemischt auftretender positiver und negativer Zahlen ist die Vorzeichen-konvention brauchbar!

2-er Komplement

2. Versuch: (2-er Komplement)

- Finde zu irgendeiner Binärzahl, z.B. 00001001_2 eine Binärzahl, so dass deren Summe 0 ist. (Beschränkung auf 8 Bit) z.B. 9 – 9 = 0

Löse das Problem rückwärts!

	0	0	0	0	1	0	0	1
+				?				
	0	0	0	0	0	0	0	0

2-er Komplement

2. Versuch: (2-er Komplement)

- Finde zu irgendeiner Binärzahl, z.B. 00001001_2 eine Binärzahl, so dass deren Summe 0 ist. (Beschränkung auf 8 Bit) z.B. 9 – 9 = 0

Löse das Problem rückwärts!

	0	0	0	0	1	0	0	1
+	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**
1	1	1	1	1	1	1	1	
	0	0	0	0	0	0	0	0

1. Beobachtung:

Es entsteht eine neue Stelle.
→ ignorieren

2. Beobachtung:

Die zu addierende Zahl ist das Komplement der ursprünglichen Zahl – bis auf die letzte Stelle.

Komplement: 0 → 1
1 → 0

2-er Komplement (1)

- Das 2-er Komplement wird in zwei Schritten gebildet

  1. Bildung des reinen Komplements
  2. Addition von 1,
     wobei eine vorne ggf. entstehende neue Stelle ignoriert wird.

- Beispiele für 2-er Komplement:

		00001111
Komplementbildung		--------------------------------
		11110000
Addition von 1	+	00000001
		11110001

		00001110
Komplementbildung		--------------------------------
		11110001
Addition von 1	+	00000001
		11110010

2-er Komplement (2)

- Komplement (Komplement (Zahl))   =   Zahl

  aber auch
- 2-er Komplement (2-er Komplement (Zahl))   =   Zahl

- Beispiel:
  - 2-er Komplement von 00001110 ist 11110010 (vorige Folie)
  - Davon das 2-er Komplement:

$$
\begin{array}{lr}
 & 11110010 \\
\text{Komplementbildung} & \text{- - - - - - - - - - - -} \\
 & 00001101 \\
\text{Addition von 1} & +\ 00000001 \\
\hline
 & 00001110
\end{array}
$$

Addition des 2-er Komplements (1)

- $(13 - 9)_{10}$

1. Schritt: **Bildung** des 2-er Komplements von 9

	0	0	0	0	1	0	0	1	
	1	1	1	1	0	1	1	0	
+	0	0	0	0	0	0	0	1	
	1	1	1	1	0	1	1	1	2-er Komplement von $(9)_{10}$

2. Schritt: **Addition** von 13 und dem 2-er Komplement von 9

		0	0	0	0	1	1	0	1	$= (13)_{10}$
+		1	1	1	1	0	1	1	1	= 2-er Komplement von $(9)_{10}$
	1	1	1	1	1	1	1	1		
		0	0	0	0	0	1	0	0	$= (4)_{10}$

ignorieren des Übertrags!

Addition des 2-er Komplements (2)

- $(9 - 13)_{10}$ (Die Darstellung negativer Zahlen ist im Zusammenhang mit dem 2-er Komplement im bisherigen Verlauf ungeklärt.)

1. Schritt: Bildung des 2-er Komplements von 13

	0	0	0	0	1	1	0	1
	1	1	1	1	0	0	1	0
+	0	0	0	0	0	0	0	1
	1	1	1	1	0	0	1	1

2-er Komplement von $(13)_{10}$

2. Schritt: Addition von 9 und 2-er Komplement von 13

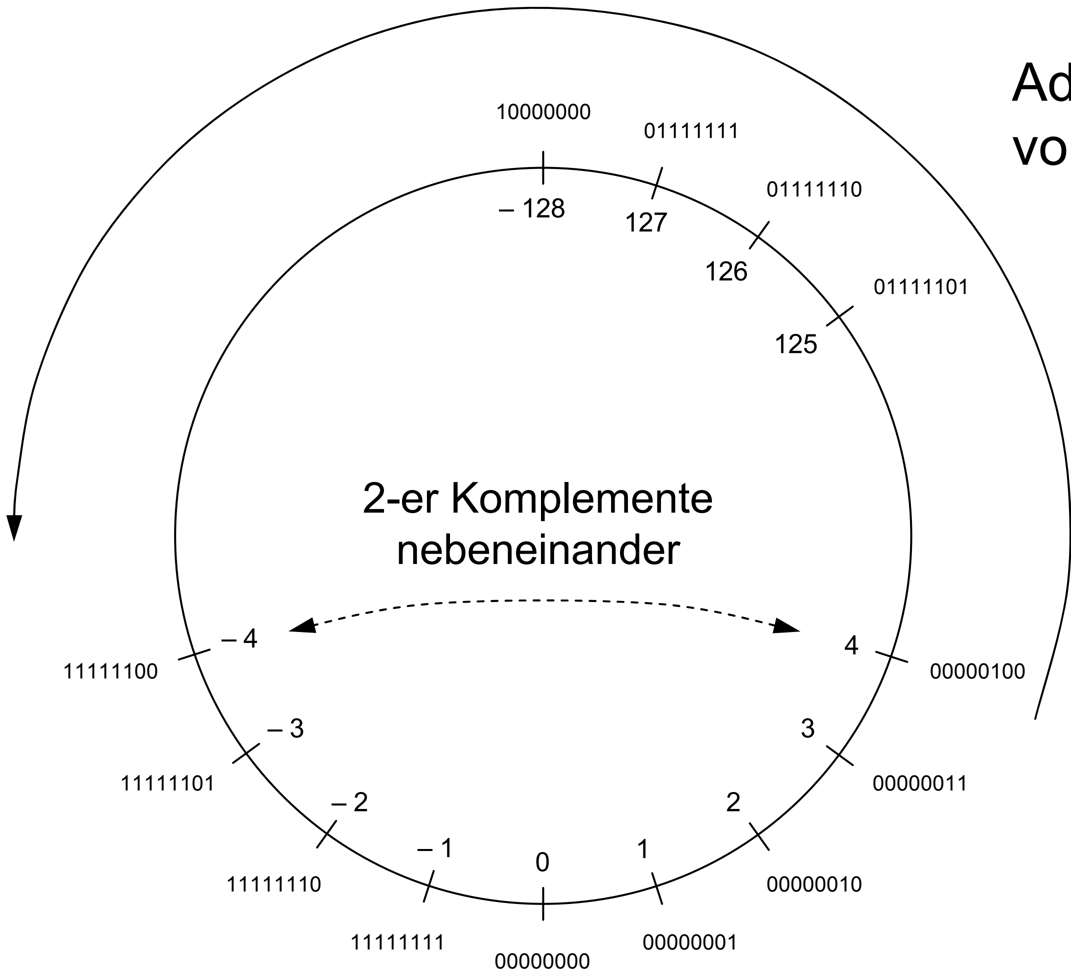
		0	0	0	0	1	0	0	1
+		1	1	1	1	0	0	1	1
	0	0	0	0	0	0	1	1	
		1	1	1	1	1	1	0	0

$= (-4)_{10}$?

kein Übertrag entstanden

Darstellung negativer Zahlen noch nicht festgelegt → möglicherweise sinnvoll

2-er Komplement von 8-Bit Zahlen (1)



Addition von 00000001
von Position zu Position

- Das erste Bit ist das Vorzeichen
  - 1 ↔ –
  - 0 ↔ +
- 2-er Komplemente liegen sich in diesem Zahlenkreis **nicht** gegenüber, sind aber betragsmäßig gleich.

2-er Komplement von 8-Bit Zahlen (2)

- Subtraktion wurde tatsächlich auf Addition mit 2-er Komplement zurückgeführt.
- Die Bildung des 2-er Komplements kann wiederum auf Addition zurückgeführt werden, denn die Komplementbildung kann auf Addition zurückgeführt werden – allerdings auf Addition **ohne** Überträge
- Beispiel:

```
    00001001
  + 11111111
  ----------
    11110110
```

Überträge ignorieren

Diese beiden Zahlen sind die reinen Komplemente voneinander.

Es wird stets diese Zahl zum Binärmuster addiert.

- Subtraktion vollständig auf Addition(en) zurückgeführt!
- Komplementbildung durch ziffernweise / bitweise Negation.

2-er Komplement von 8-Bit Zahlen (3)

- Zahlenbereichsüberschreitungen beim 2-er Komplement werden in dieser Vorlesung (Übungen, Tutorien, Klausur) **nicht** behandelt.

- Multiplikationen beim 2-er Komplement werden in dieser Vorlesung (Übungen, Tutorien, Klausur) **auch nicht** behandelt.