technische universität dortmund

fakultät für informatik

Rechnerstrukturen, Teil 1

Vorlesung 4 SWS WS 17/18

Prof. Dr. Jian-Jia Chen
Fakultät für Informatik – Technische Universität Dortmund
jian-jia.chen@cs.uni-dortmund.de
http://ls12-www.cs.tu-dortmund.de

Übersicht

1. Organisatorisches ✓
2. Einleitung ✓
3. **Repräsentation von Daten**
4. Boolesche Funktionen und Schaltnetze
5. Rechnerarithmetik
6. Optimierung von Schaltnetzen
7. Programmierbare Bausteine
8. Synchrone Schaltwerke

3. Repräsentation von Daten

3.2 Repräsentation von Texten

Beobachtung

Alles, was wir wissen, können wir aufschreiben.

Folgerung

Repräsentation von Texten ist zentral.

Naiver Ansatz

- Codierung A → 0, B → 1, . . .
- und dann Binärcodierung

Probleme

- Welche Zeichen sollen codiert werden?
- Wie kann man Daten mit anderen austauschen?

Lösung: **Definition von Standards**

3.2 Repräsentation von Texten

Texte: Zeichenfolgen aus Buchstaben und Satzzeichen

- Darstellung mittels Bitfolgen
- Codierung jedes Buchstabens / Zeichens durch Bitfolge

ASCII = American Standard Code for Information Interchange

- 7 Bit (= max. 128 Zeichen), Tabelle mit Nummerierung aller Zeichen
- z.B. „a“ hat Nummer 97, „A“ hat Nummer 65, „?“ hat Nummer 63
- Klein- und Großbuchstaben nach Alphabet durchnummeriert
- übliche Erweiterung auf PCs: 8bit, weitere Sonderzeichen, z.B. Umlaute
- Erweiterung in Europa: Latin-1, (nach Norm ISO 8859-1)

Unicode, z.B. von Java verwendet

- 16 Bit (= max. 65536 Zeichen)
- siehe http://www.unicode.org
- als Obermenge weltweit geläufiger Zeichensätze

3.2 Repräsentation von Texten

ASCII-Tabelle (7 Bit)

	000	001	010	011	100	101	110	111
0000...	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL
0001...	BS	HT	LF	VT	FF	CR	SO	SI
0010...	DLE	DC1	XON	DC3	XOF	NAK	SYN	ETB
0011...	CAN	EM	SUB	ESC	FS	GS	RS	US
0100...		!	"	#	$	%	&	'
0101...	(	)	*	+	,	-	.	/
0110...	0	1	2	3	4	5	6	7
0111...	8	9	:	;	<	=	>	?
1000...	@	A	B	C	D	E	F	G
1001...	H	I	J	K	L	M	N	O
1010...	P	Q	R	S	T	U	V	W
1011...	X	Y	Z	[	\	]	^	_
1100...	`	a	b	c	d	e	f	g
1101...	h	i	j	k	l	m	n	o
1110...	p	q	r	s	t	u	v	w
1111...	x	y	z	{	$\vert$	}	~	DEL

3.2 Repräsentation von Texten

ISO 8859-1 (8 Bit)

0020	0 0030	@ 0040	P 0050	` 0060	p 0070	00A0	° 00B0	À 00C0	Ð 00D0	à 00E0	ð 00F0
! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071	¡ 00A1	± 00B1	Á 00C1	Ñ 00D1	á 00E1	ñ 00F1
" 0022	2 0032	B 0042	R 0052	b 0062	r 0072	¢ 00A2	² 00B2	Â 00C2	Ò 00D2	â 00E2	ò 00F2
# 0023	3 0033	C 0043	S 0053	c 0063	s 0073	£ 00A3	³ 00B3	Ã 00C3	Ó 00D3	ã 00E3	ó 00F3
$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074	¤ 00A4	´ 00B4	Ä 00C4	Ô 00D4	ä 00E4	ô 00F4
% 0025	5 0035	E 0045	U 0055	e 0065	u 0075	¥ 00A5	µ 00B5	Å 00C5	Õ 00D5	å 00E5	õ 00F5
& 0026	6 0036	F 0046	V 0056	f 0066	v 0076	¦ 00A6	¶ 00B6	Æ 00C6	Ö 00D6	æ 00E6	ö 00F6
' 0027	7 0037	G 0047	W 0057	g 0067	w 0077	§ 00A7	· 00B7	Ç 00C7	× 00D7	ç 00E7	÷ 00F7
(0028	8 0038	H 0048	X 0058	h 0068	x 0078	¨ 00A8	¸ 00B8	È 00C8	Ø 00D8	è 00E8	ø 00F8
) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079	© 00A9	¹ 00B9	É 00C9	Ù 00D9	é 00E9	ù 00F9
* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A	ª 00AA	º 00BA	Ê 00CA	Ú 00DA	ê 00EA	ú 00FA
+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B	« 00AB	» 00BB	Ë 00CB	Û 00DB	ë 00EB	û 00FB
, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C	¬ 00AC	¼ 00BC	Ì 00CC	Ü 00DC	ì 00EC	ü 00FC
- 002D	= 003D	M 004D	] 005D	m 006D	} 007D	- 00AD	½ 00BD	Í 00CD	Ý 00DD	í 00ED	ý 00FD
. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E	® 00AE	¾ 00BE	Î 00CE	Þ 00DE	î 00EE	þ 00FE
/ 002F	? 003F	O 004F	_ 005F	o 006F	007F	¯ 00AF	¿ 00BF	Ï 00CF	ß 00DF	ï 00EF	ÿ 00FF

3.2 Repräsentation von Texten

ISO 8859-1 (8 Bit)

- International Organization for Standardization (gegründet 1947)
- ISO Latin 1
  - 8 Bit Code
  - enthält viele Sonderzeichen für westeuropäische Sprachen (z. B. Umlaute (¨ a, ¨ o, ¨ u, . . .))
  - enthält nicht alle gewünschten Zeichen (z. B. fehlen japanische und andere asiatische Schriftzeichen, . . .)

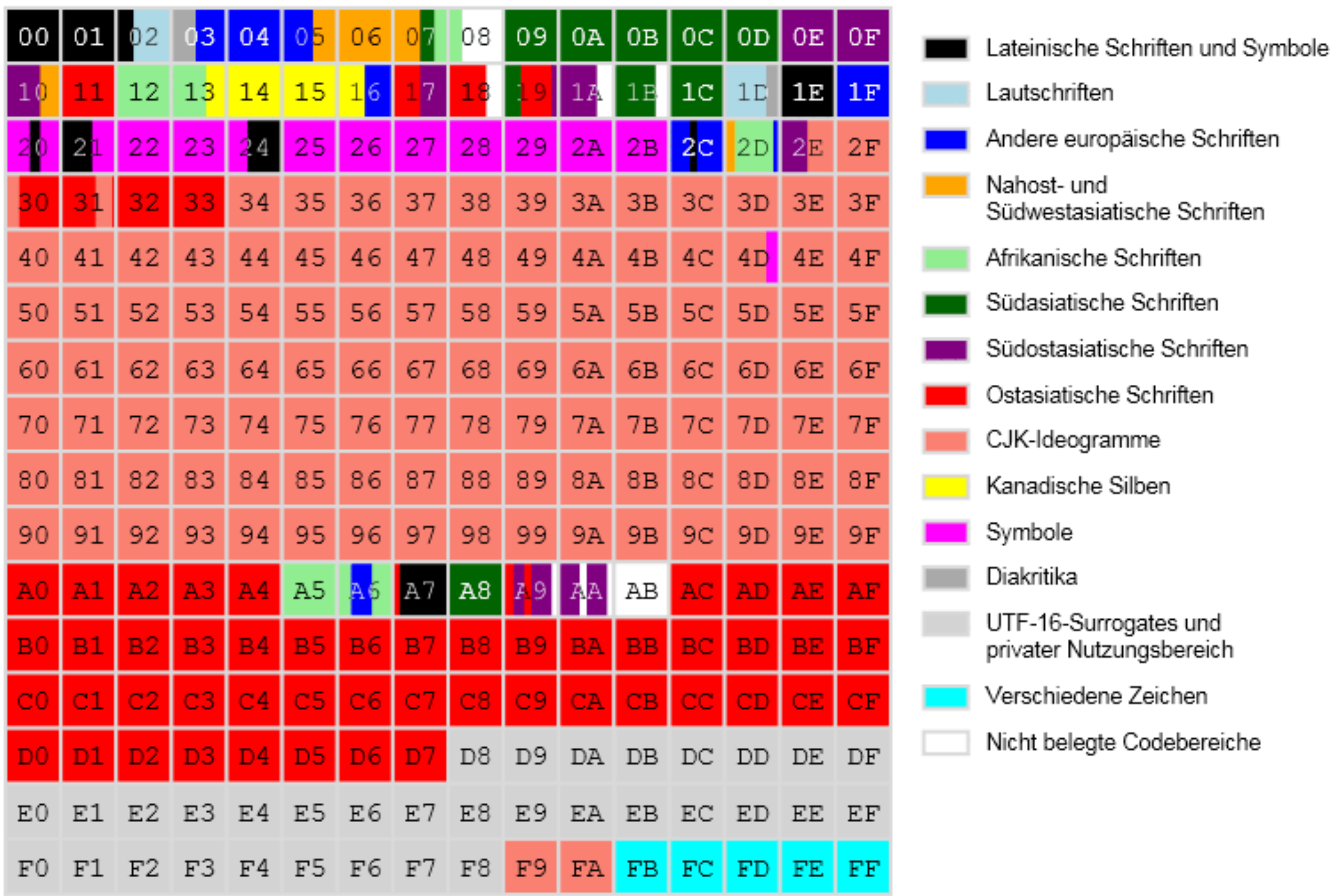
3.2 Repräsentation von Texten

Unicode

- aktueller Standard Unicode 9.0 (Stand: 7. Oktober 2016)
- verwaltet vom Unicode-Konsortium (http://www.unicode.org)
- unterstützt verschiedene Codierungsformate (Unicode Transformation Format):
  - UTF-8,
  - UTF-16,
  - UTF-32
- längere Formate erweitern kürzere Formate
- vereinbart auch weitere Informationen
  - Schreibrichtung,
  - Kombination von Zeichen
  - …

3.2 Repräsentation von Texten

Unicode



3.2 Repräsentation von Texten

Unicode – Beispiel: Kanadische Silben

ᐃᓄᒃᑎᑐᑦ
Inuktitut Syllabarium

ᖃᓂᐅᔮᖅᐸᐃᑦ

Anlaute	Silben				Auslaute
	ai	i	u	a	
ø	ᐁ	ᐃ	ᐅ	ᐊ	
p	ᐯ	ᐱ	ᐳ	ᐸ	ᐊᑉ
t	ᑌ	ᑎ	ᑐ	ᑕ	ᐊᑦ
k	ᑫ	ᑭ	ᑯ	ᑲ	ᐊᒃ
g	ᒉ	ᒋ	ᒍ	ᒐ	ᐊᒡ
m	ᒣ	ᒥ	ᒧ	ᒪ	ᐊᒻ
n	ᓀ	ᓂ	ᓄ	ᓇ	ᐊᓐ
s	ᓭ	ᓯ	ᓱ	ᓴ	ᐊᔅ
l	ᓓ	ᓕ	ᓗ	ᓚ	ᐊᓪ
j	ᔦ	ᔨ	ᔪ	ᔭ	ᐊᔾ
v	ᕓ	ᕕ	ᕗ	ᕙ	ᐊᕝ
r	ᕃ	ᕆ	ᕈ	ᕋ	ᐊᕐ
q	ᙯ	ᕿ	ᖁ	ᖃ	ᐊᖅ
ng	ᙰ	ᖏ	ᖑ	ᖓ	ᐊᖕ
nng	□	ᙱ	ᙳ	ᙵ	ᐊᖖ
□		ᖠ	ᖢ	ᖤ	ᐊᖦ

3.2 Repräsentation von Texten

Zur Bedeutung der Repräsentation von Texten

Programme

- Ein Programm wird zunächst meist als Text (**Quelltext**) erzeugt und wie normaler Text repräsentiert.
- Übersetzungsprogramme (**Compiler**) erzeugen daraus Programmcode in Maschinen-sprache.
- In jeder Form müssen alle Anteile eines Programms durch **Bitfolgen** codiert werden.

```java
public class HelloWorld {
    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Hallo Welt!");
    }
}
```

```
// Bytecode stream: 03 3b 84 00 01 1a 05 68 3b a7 ff f9
// Disassembly:
iconst_0      // 03
istore_0      // 3b
iinc 0, 1     // 84 00 01
iload_0       // 1a
iconst_2      // 05
imul          // 68
istore_0      // 3b
goto -7       // a7 ff f9
```

```
:10000000758 12F120199120252120258 90004D125E
:100010000 27B90005B12028575020 0744D12022D66
:10002000120069120327120 0E304F9D8FED9FC7408
:10003000 0E12022DE59012037A20B3040502800 20D
:1000400015028502A030B2D37502FF80CE4449501C
:1000500038205031286865782 93A00414455776541
```

3. Repräsentation von Daten

1. Repräsentation von natürlichen Zahlen ✓
2. Repräsentation von Texten ✓
3. **Repräsentation ganzer Zahlen**
4. Repräsentation rationaler Zahlen
5. Repräsentation anderer Daten

3.3 Repräsentation ganzer Zahlen

Wunsch

ganze Zahlen z $\in Z$ repräsentieren zu können

Vereinbarung

feste Repräsentationslänge ℓ

Wie viele verschiedene Zahlen kann man so höchstens repräsentieren?

- ℓ Positionen, jeweils 2 Möglichkeiten pro Position
- → 2^ℓ verschiedene Bitmuster der Länge ℓ

Fragen

- Wie können wir Zahlenbereich verwenden?
- Wie können wir positive und negative Zahlen unterscheiden?

3.3 Repräsentation ganzer Zahlen

Vorzeichenbetragsmethode

Vereinbarungen

- erstes Bit Vorzeichen
- restliche Bits Betragszahl in Binärdarstellung
- Vorzeichenbit $s = 1 \Leftrightarrow$ Vorzeichen negativ,
  da Zahl = $(-1)^s \cdot$Betragszahl gilt

Es gilt

- kleinste darstellbare Zahl $-(2^{\ell-1} - 1)$
- größte darstellbare Zahl $2^{\ell-1} - 1$

Eigenschaften

- + symmetrisch
- + Vorzeichenwechsel einfach
- − 0 nicht eindeutig
- − Vergleich von Zahlen schwierig
- − Verschwendung eines Codes

3.3 Repräsentation ganzer Zahlen

Darstellung mit Bias (Exzessdarstellung)

Vereinbarungen

- feste Verschiebung b (**Bias**).
- z wird als z + b als Binärzahl dargestellt
- übliche Werte für Bias $b = 2^{\ell-1}$ oder $b = 2^{\ell-1}-1$
- Beispiel $\ell=5 \rightarrow b=16$ oder $b=15$

Es gilt

- kleinste darstellbare Zahl $-b$
- größte darstellbare Zahl $2^{\ell} - 1 - b$

Eigenschaften

- + 0 eindeutig & alle Codes ausgenutzt
- + Vergleich von Zahlen einfach
- + bei üblichem Bias erstes Bit vorzeichenbitanalog
- − nicht symmetrisch
- − Vorzeichenwechsel schwierig

3.3 Repräsentation ganzer Zahlen

Einerkomplementdarstellung

Vereinbarungen

- nicht-negative Zahlen: **Binärdarstellung**
- negative Zahlen **Komplement der Binärdarstellung**

Es gilt

- kleinste darstellbare Zahl $-(2^{\ell-1}-1)$
- größte darstellbare Zahl $2^{\ell-1}-1$

Eigenschaften

- + symmetrisch
- + erstes Bit wie Vorzeichenbit
- – 0 nicht eindeutig
- – Verschwendung eines Codes
- – Vergleich von Zahlen schwierig

3.3 Repräsentation ganzer Zahlen

Zweierkomplementdarstellung

Vereinbarungen

- nicht-negative Zahlen: **Binärdarstellung**
- negative Zahlen **Komplement der Binärdarstellung + 1**

Es gilt

- kleinste darstellbare Zahl $-(2^{\ell-1})$
- größte darstellbare Zahl $2^{\ell-1} - 1$

Eigenschaften

- 0 eindeutig repräsentiert & alle Codes ausgenutzt
- \+ erstes Bit wie Vorzeichenbit
- \- Vergleich von Zahlen schwierig

Zweierkomplementdarstellung ist der Standard

Geschlossene Form zur Berechnung des Zweierkomplements

Theorem

Ein Bitvektor a = (a_{n-1}, ..., a_0) repräsentiert bei einer Kodierung im Zweierkomplement die Zahl

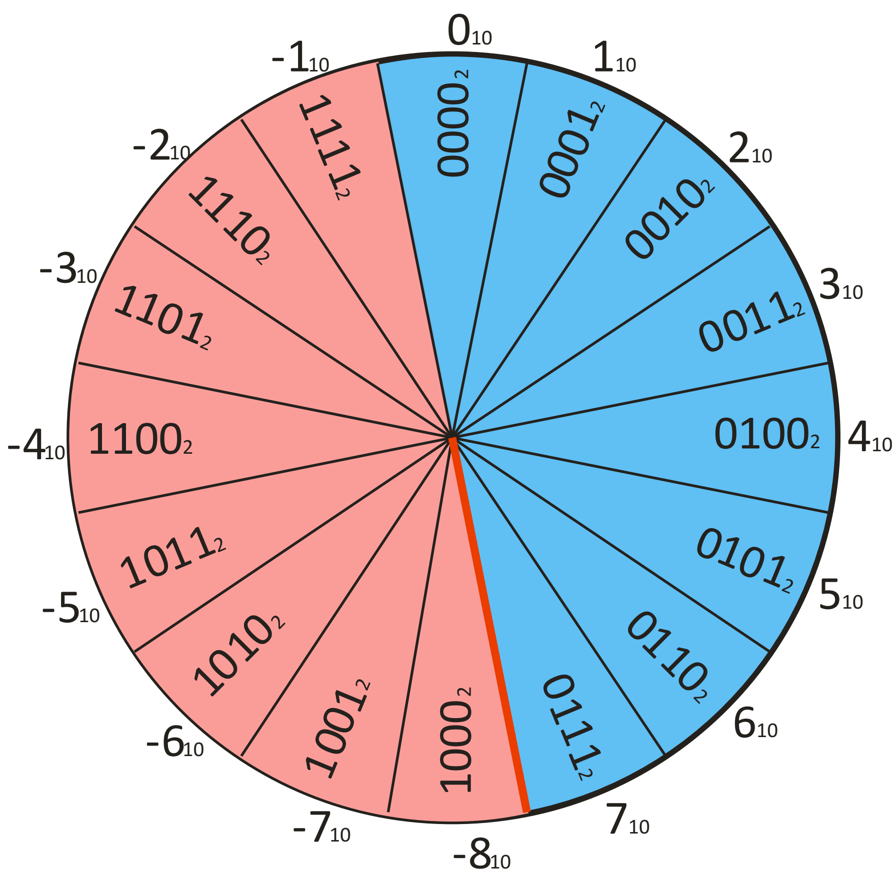
$$int(a) = -a_{n-1} \cdot 2^{n-1} + \sum_{i=0}^{n-2} a_i \cdot 2^i$$

Beispiel

$\text{int}(1101_{2K}) = -1 \cdot 2^3 + (1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0) = -8_{10} + 5_{10} = -3_{10}$

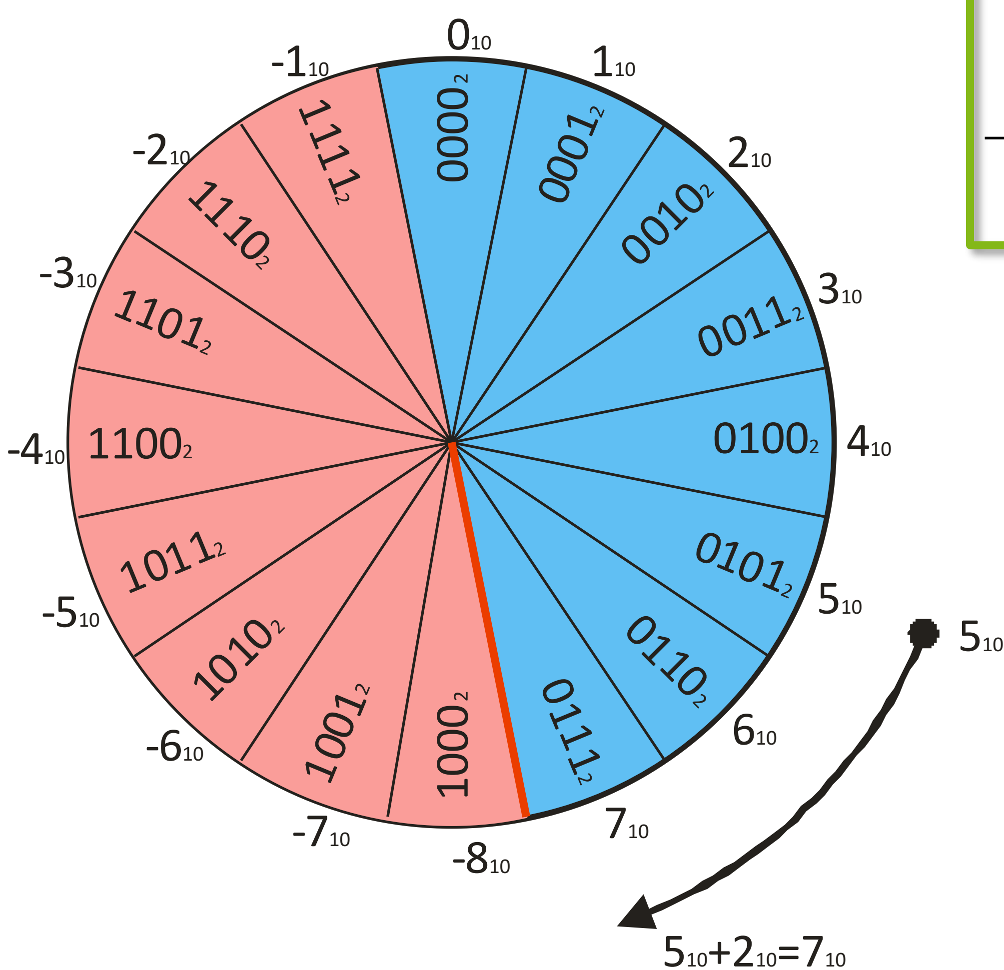
3.3 Repräsentation ganzer Zahlen

Für die Darstellung im Zweierkomplement ergibt sich folgendes Zahlenrad



3.3 Repräsentation ganzer Zahlen

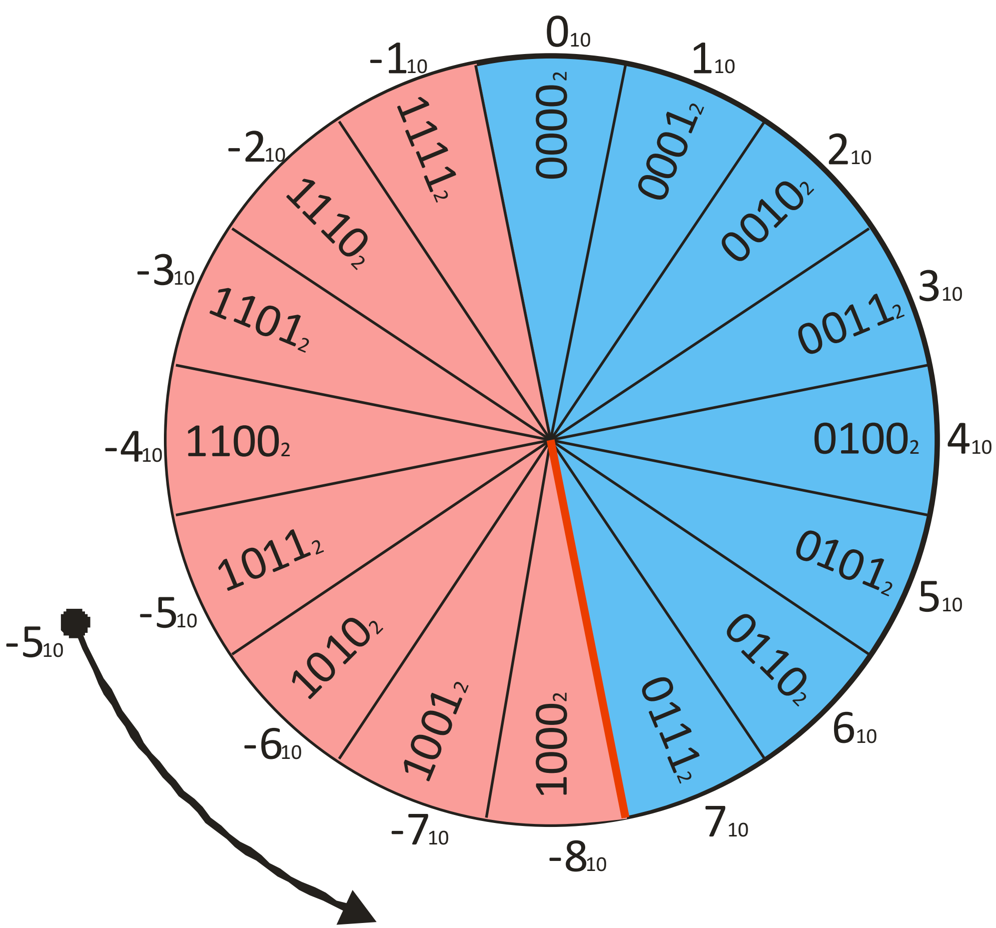
Keine Fallunterscheidung bei Addition und Subtraktion



5	**0**	**1**	**0**	**1**
+ 2	**0**	**0**	**1**	**0**
7	**0**	**1**	**1**	**1**

3.3 Repräsentation ganzer Zahlen

Keine Fallunterscheidung bei Addition und Subtraktion



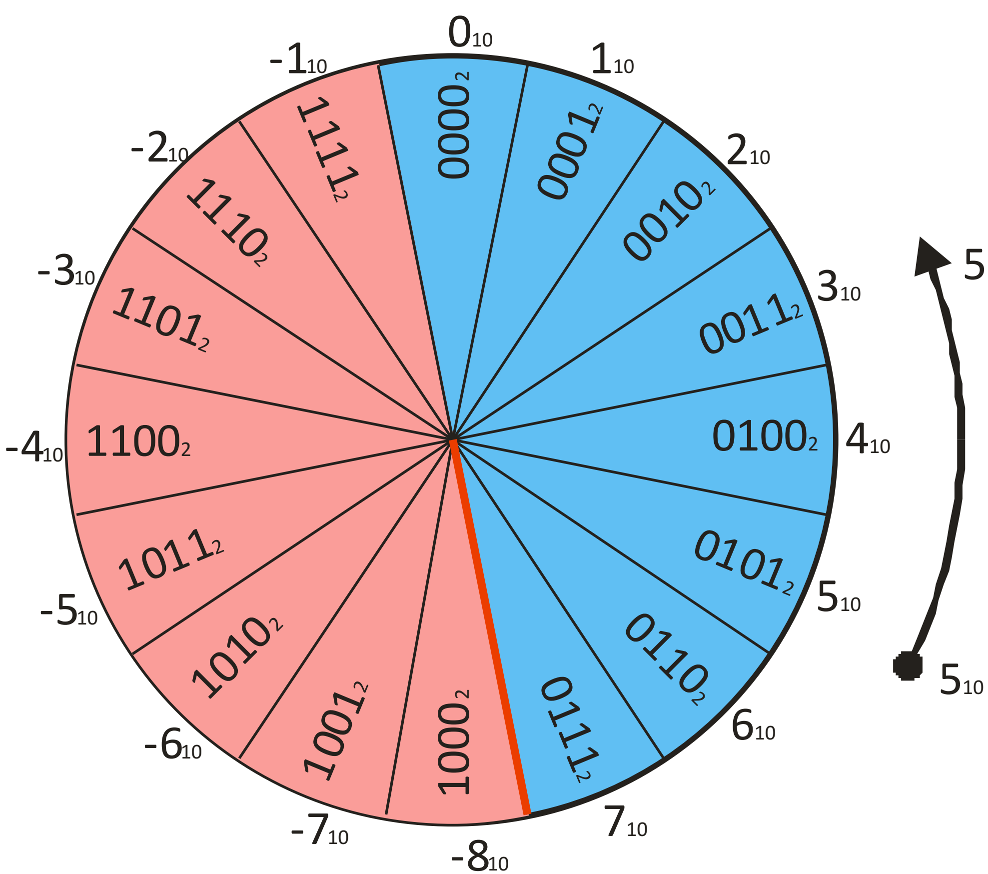
$-(5_{10}) + -(2_{10}) = -7_{10}$

-5		**1**	**0**	**1**	**1**
+ -2		**1**	**1**	**1**	**0**
-7	**1**	**1**	**0**	**0**	**1**

Übertragbit ignorieren

3.3 Repräsentation ganzer Zahlen

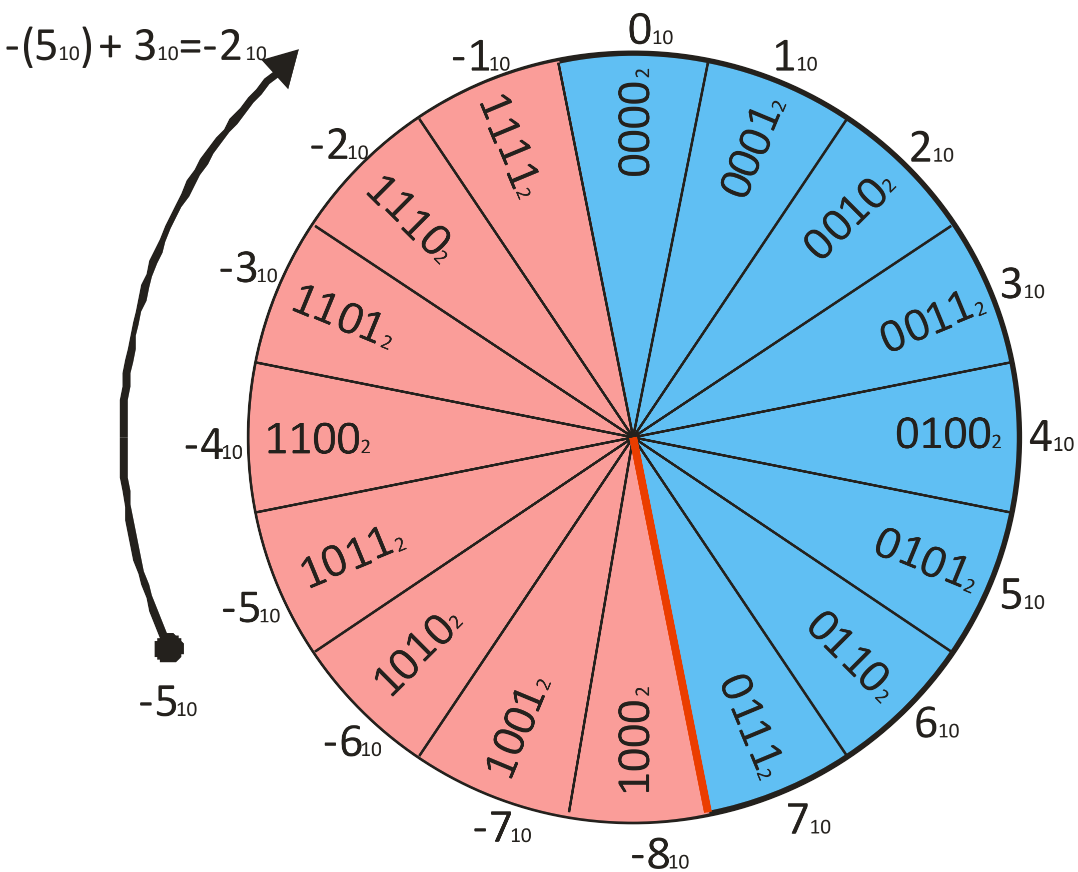
Keine Fallunterscheidung bei Addition und Subtraktion



5		**0**	**1**	**0**	**1**
+ -2		**1**	**1**	**1**	**0**
3	**1**	**0**	**0**	**1**	**1**

Übertragbit ignorieren

Keine Fallunterscheidung bei Addition und Subtraktion

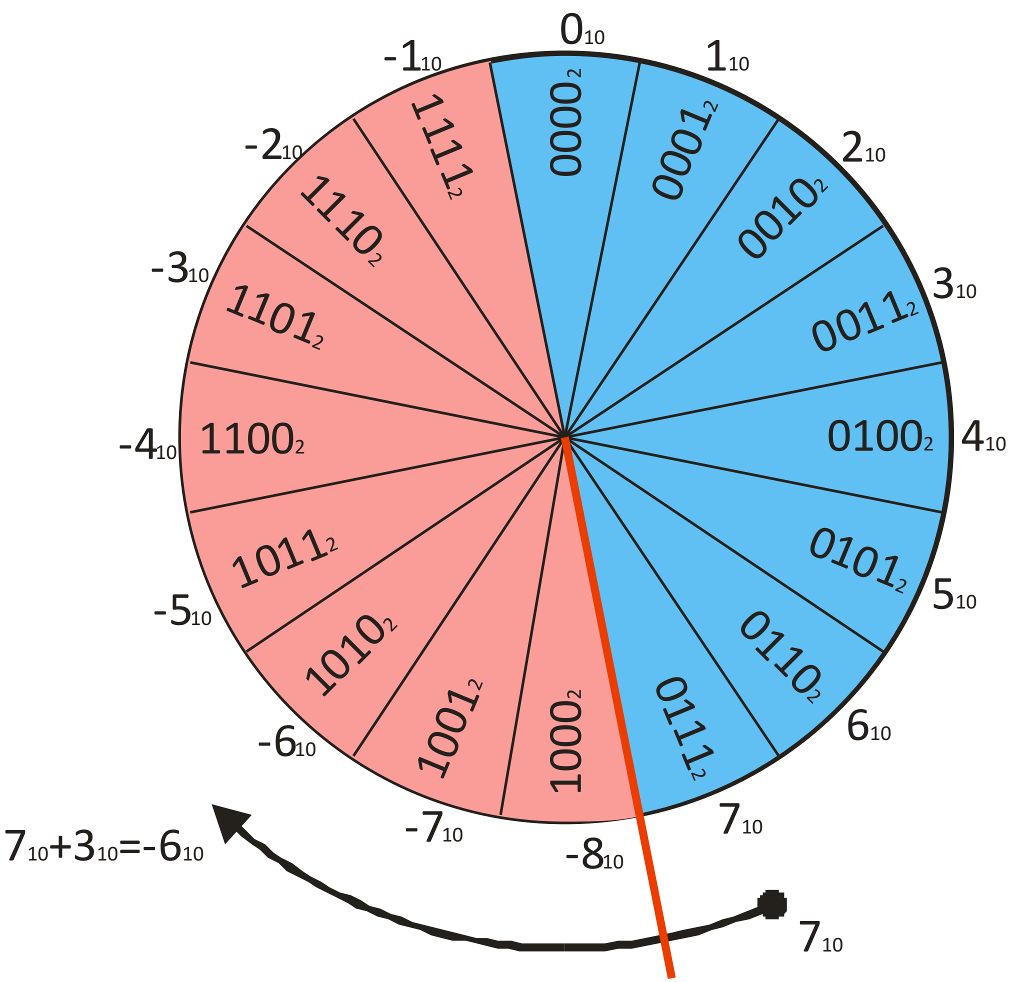


-5		**1**	**0**	**1**	**1**
+ 3		**0**	**0**	**1**	**1**
-2	**0**	**1**	**1**	**1**	**0**

Übertragbit ignorieren

3.3 Repräsentation ganzer Zahlen

Aber: Vorsicht bei Bereichsüberschreitung



7		0	1	1	1
+ 3		0	0	1	1
-6	0	1	0	1	0

Übertragbit ignorieren

3.3 Repräsentation ganzer Zahlen

Alles verstanden?

Wir haben

- $\ell = 5$
- Bias $b = 2^{\ell-1} = 2^4 = 16$
- $z = 10010$

Welche Zahl wird denn nun repräsentiert?

Vorzeichenbetragsdarstellung

1	0010_2
Vorzeichenbit	Binärdarstellung
-1^1	$0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 2$

→ $10010_2 = -1 \cdot 2 = -2$

3.3 Repräsentation ganzer Zahlen

Alles verstanden?

Wir haben

- $\ell = 5$
- Bias $b = 2^{\ell-1} = 2^4 = 16$
- $z = 10010$

Welche Zahl wird denn nun repräsentiert?

Darstellung mit Bias b = 16 (Exzessdarstellung)

$$10010_2 = 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 - b$$

$$= 16 + 2 - b$$

$$= 18 - 16$$

$$= 2$$

→ $10010_2 = 2$

3.3 Repräsentation ganzer Zahlen

Alles verstanden?

Wir haben

- $\ell = 5$
- Bias $b = 2^{\ell-1} = 2^4 = 16$
- $z = 10010$

Welche Zahl wird denn nun repräsentiert?

Einerkomplementdarstellung

- Führendes Bit 1 → negative Zahl
- Komplement von $10010_2 = 01101_2$
- $01101_2 = 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$

  $= 8 + 4 + 1 = 13$

→ **10010_2 = -13**

Alles verstanden?

Wir haben

- $\ell = 5$
- Bias $b = 2^{\ell-1} = 2^4 = 16$
- $z = 10010$

Welche Zahl wird denn nun repräsentiert?

Zweierkomplementdarstellung

- Führendes Bit 1 → negative Zahl
- Zweierkomplement: $10010_2 - 1_2 = 10001_2$
- Komplement von $10001_2 = 01110_2$
- $01110_2 = 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 14$

→ **$10010_2 = -14$**

3.3 Repräsentation ganzer Zahlen

Alles verstanden?

Wir haben

- $\ell = 5$
- Bias $b = 2^{\ell-1} = 2^4 = 16$
- $z = 10010$

Welche Zahl wird denn nun repräsentiert?

- -2 Vorzeichenbetragsdarstellung
- 2 Darstellung mit Bias b = 16 (Exzessdarstellung)
- -13 Einerkomplementdarstellung
- -14 Zweierkomplementdarstellung

Und richtig sind ohne weitere Informationen über den Datentyp alle 4!

3.3 Repräsentation ganzer Zahlen

Beispielsammlung Darstellungen

Wir haben feste Länge $\ell = 5$

z	VZ-Betrag	Bias b=16	Bias b=15	1er-K.	2er-K.
-16	-	00000	-	-	10000
-15	11111	00001	00000	10000	10001
…	…	…	…	…	…
-1	10001	01111	01110	11110	11111
0	00000 10000	10000	01111	00000 11111	0000
1	00001	10001	10000	00001	00001
…	…	…	…	…	…
15	01111	11111	11110	01111	01111
16	-	-	11111	-	-

3. Repräsentation von Daten

3. Repräsentation von Daten

1. Repräsentation von natürlichen Zahlen ✓
2. Repräsentation von Texten ✓
3. Repräsentation ganzer Zahlen ✓
4. **Repräsentation rationaler Zahlen**
5. Repräsentation anderer Daten

3.4 Repräsentation rationaler Zahlen

Wunsch

rationale Zahlen $q \in \mathbb{Q}$ repräsentieren können

Vereinbarung

- feste Repräsentationslänge ℓ Bits
- feste Position des Kommas
- **Festkommadarstellung**

Beispiele für $\ell = 8$ bei 3 Nachkommastellen

- 10,34 → 00 010,340
- 93 847,123 → 93 847,123
- 0 → 00 000,000
- 123 456,78 nicht darstellbar (zu viele Stellen vor dem Komma)
- 12,345678 nicht darstellbar (aber runden denkbar)

3.4 Repräsentation rationaler Zahlen

Beobachtung

- Basis 10 im Bereich der digitalen Rechner nur bedingt brauchbar
- Idee ist aber übertragbar auf Basis 2

Beispiel: 93 847,123

10^4	10^3	10^2	10^1	10^0	10^{-1}	10^{-2}	10^{-3}
10 000	1 000	100	10	1	0,1	0,01	0,001
9	3	8	4	7	1	2	3

Übertragung auf Basis 2: $(10110,101)_2$

2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}
16	8	4	2	1	1/2 =0,5	1/4 = 0,25	1/8 = 0,125
1	0	1	1	0	1	0	1

= 16 + 4 + 2 + 0,5 + 0,125 = 22,625

3.4 Repräsentation rationaler Zahlen

Allgemein gilt

- bei *v* **Vorkommastellen**
- und *m* **Nachkommastellen**

$$a = \sum_{i=-m}^{v-1} z_i \cdot 2^i$$

3.4 Repräsentation rationaler Zahlen

Beobachtung

- nicht alle Zahlen sind darstellbar
  - z. B.: 0,2
  - z. B.: pi
- ist das schlimm?
  - im Dezimalsystem ist 1/3 nicht darstellbar
  - pi hat unendlich viele Stellen, ist generell nicht darstellbar
- Festkommadarstellung hat **Vorteile**
  - bildet Rechenoperationen auf Ganzzahloperationen ab
  - keine Fließkommahardware nötig
  - schnelle Ausführung
  - Einsatz im Bereich der digitalen Signalverarbeitung
- **Nachteile**
  - feste Position des Kommas
  - unflexibel

3.4 Repräsentation rationaler Zahlen

Andere Idee

2017

3.4 Repräsentation rationaler Zahlen

Gleitkommazahlen

- Vorzeichenbit $s \in \{0,1\}$
- Mantisse $m \in \mathrm{Q}$
- Exponent $e \in Z$ (bestimmt die Position des Kommas)

$$z = (-1)^s \cdot m \cdot 10^e$$

Normalisierte Gleitkommazahl

- $1 \leq m < 10$
- Beispiel: $2{,}017 \cdot 10^3$

3.4 Repräsentation rationaler Zahlen

Übertragung auf Basis b=2

- Vorzeichenbit $s \in \{0,1\}$
- Mantisse $m \in \mathrm{Q}$
- Exponent $e \in Z$ (bestimmt die Position des Kommas)

$$z = (-1)^s \cdot m \cdot 2^e$$

Normalisierte Gleitkommazahl

- $1 \leq m < 2$

Beobachtung

- erste Ziffer der Mantisse ist **immer** 1
- 0 ist so nicht darstellbar

3.4 Repräsentation rationaler Zahlen

IEEE-754 1985

- Vorzeichenbit $s \in \{0,1\}$
- Mantisse $m \in Q$
- Exponent $e \in Z$
- $1 \leq m < 2$

$$z = (-1)^s \cdot m \cdot 2^e$$

Festlegung

- führende 1 der Mantisse wird nicht mit abgespeichert: **Implizite Eins**
- Mantisse in Binärcodierung als *Festkommazahlen* mit Ziffern **ausschließlich hinter dem Komma**
- Exponent in Exzessdarstellung mit Bias $b = 2^{\ell_e - 1} - 1$

3.4 Repräsentation rationaler Zahlen

IEEE-754 1985

Definierte Varianten

Typ	Größe (1+r+p)	Exponent (r)	Mantisse (p)	Werte des Exponenten (e)	Biaswert (B)
single	32 bit	8 bit	23 bit	$-126 \leq e \leq 127$	127
double	64 bit	11 bit	52 bit	$-1022 \leq e \leq 1023$	1023

Besonderheiten beim Exponenten

Verkleinerung des zulässigen Bereichs um 1 auf beiden Seiten auf

- $e_{min} = -b + 1 = -(2^{\ell e - 1} - 1) + 1,$
- $e_{max} = 2^{\ell e} - 1 - b - 1 = 2^{\ell e - 1} - 1$

dient der Codierung **besonderer Zahlen**

3.4 Repräsentation rationaler Zahlen

IEEE-754 1985

Besondere Zahlen

Vorzeichen	Exponent	Mantisse	Zahl
0	$e_{max} + 1$	0	$+\infty$
1	$e_{max} + 1$	0	$-\infty$
s	$e_{max} + 1$	$\neq 0$	NaN
0	$e_{min} - 1$	0	0
1	$e_{min} - 1$	0	-0
s	$e_{min} - 1$	$\neq 0$	c.f. denormalisierte Darstellung

NaN: Not a Number

letzte Zeile: denormalisierte Darstellung

IEEE-754 1985

Beispiel 1 (Typ: single)

- $\ell = 32$, $\ell_s = 1$, $\ell_e = 8$, $\ell_m = 23$
- $b = 2^7 - 1 = 127 \rightarrow e_{min} = -b + 1 = -126$, $e_{max} = 2^8 - b - 2 = 127$

Wir wollen −3 darstellen:

- **negativ**, also **Vorzeichenbit** s=1
- **Darstellung als Summe von Zweierpotenzen**
  $3 = 2 + 1 = 2^1 + 2^0 = (2^0 + 2^{-1}) \cdot 2^1$
- **Mantisse** $(11000...)_2$, **implizite Eins** entfällt, also 100 0000 0000 · · ·
- **Exponent** 1 **Exzessdarstellung** 1 + b = 128 darstellen
- $128 = (1000\ 0000)_2$

1	1000 0000	100 0000 0000 0000 0000 0000
Vorzeichenbit	Exponent	Mantisse

3.4 Repräsentation rationaler Zahlen

IEEE-754 1985

Beispiel 2 (Typ: single)

- $\ell = 32$, $\ell_s = 1$, $\ell_e = 8$, $\ell_m = 23$
- $b = 2^7 - 1 = 127 \rightarrow e_{min} = -b + 1 = -126$, $e_{max} = 2^8 - b - 2 = 127$

Wir wollen 0,0546875 darstellen:

- **positiv**, also **Vorzeichenbit** s=0
- **Darstellung als Summe von Zweierpotenzen**
  $0{,}0546875 = 1/32 + 1/64 + 1/128 = 2^{-5} + 2^{-6} + 2^{-7} = (2^0 + 2^{-1} + 2^{-2}) \cdot 2^{-5}$
- **Mantisse** $(11100...)_2$, **implizite Eins** entfällt, also 110 0000 0000 · · ·
- **Exponent** -5 **Exzessdarstellung** -5 + b = 122 darstellen
- $122 = (0111\ 1010)_2$

Vorzeichenbit	Exponent	Mantisse
0	0111 1010	110 0000 0000 0000 0000 0000

3.4 Repräsentation rationaler Zahlen

IEEE-754 1985

Beispiel 3 (Typ: single)

- $\ell = 32$, $\ell_s = 1$, $\ell_e = 8$, $\ell_m = 23$
- $b = 2^7 - 1 = 127 \rightarrow e_{min} = -b + 1 = -126$, $e_{max} = 2^8 - b - 2 = 127$

Gegeben sei 0 1011 0001 010 1001 0000 0000 0000 0000:

0	1011 0001	010 1001 0000 0000 0000 0000
Vorzeichenbit	Exponent	Mantisse

- **Vorzeichenbit** s=0, also **positiv**
- **Exponent** $(1011\ 0001)_2 = 177$
- **Exzessdarstellung** $177 - b = 177 - 127 = 50$
- **Mantisse** $(010\ 1001\ \ldots)_2$, zuzüglich **impliziter Eins** $(1{,}010\ 1001\ \ldots)_2$
- $(2^0 + 2^{-2} + 2^{-4} + 2^{-7}) \cdot 2^{50} = 2^{50} + 2^{48} + 2^{46} + 2^{43} = 1\ 486\ 539\ 720\ 753\ 152$

3.4 Repräsentation rationaler Zahlen

IEEE-754 1985

$$(-1)^s \cdot \left(\sum_{i=1}^{l_m} m_i \cdot 2^{-i}\right) \cdot 2^{e_{min}}$$

Denormalisierte Darstellung

- gilt, wenn Exponent = $e_{min} - 1$
- und Mantisse ≠0
- erlaub es, noch kleinere Zahlen darzustellen

Beispiel

- `If x ≠ y Then z := 1/(x - y)`
- x = 0 0000 0001 **(1)** 000 0000 0000 0000 0000 0001 $2^{-126} + 2^{-149}$
- y = 0 0000 0001 **(1)** 000 0000 0000 0000 0000 0000 2^{-126}

klar x ≠ y, aber ohne denormalisierte Darstellung ist x − y = 0 gerundet

- denormalisiert x – y = 2^{-149}
- x – y = 2^{-149} = 0 0000 0000 000 0000 0000 0000 0000 0001

- auf jeden Fall besserer Programmierstil
  `If x - y ≠ 0 Then z := 1/(x - y)`

3. Repräsentation von Daten

1. Repräsentation von natürlichen Zahlen ✓
2. Repräsentation von Texten ✓
3. Repräsentation ganzer Zahlen ✓
4. Repräsentation rationaler Zahlen ✓
5. **Repräsentation anderer Daten**

3.5 Repräsentation anderer Daten

Was sind andere Daten?

- Bilder, Audiodaten, Videodaten
- Objekte, Felder
- für uns interessant: primitive Daten
  Datentypen, die direkt von der Hardware unterstützt werden

Programme

- Bitmuster fester Länge (z. B.: 32 Bit, 64 Bit)
  - Befehl
  - Operanden

Problem: Was repräsentiert ein Byte im Speicher?

- Typbits höchstens für große Bereiche (Programm, Daten)
- Bei Daten sorgt der Compiler für richtige Interpretation
- Typisierung wichtig!

3.5 Repräsentation anderer Daten

Repräsentation von Datenfolgen

Speicher oft in **Worten** organisiert

- Wort ja nach Rechner 2 Bytes, 4 Bytes, . . .

Heterogene Daten

- werden hintereinander in den Speicher geschrieben
- dabei manchmal Wortgrenzen beachten
- dann leere Zellen (Bytes) möglich

Homogene Daten

- Arrays
- Problem Wie erkennt man das Ende der Folge?
  - feste Anzahl vereinbaren
  - Länge am Anfang speichern
  - spezielles Endezeichen verwenden

3. Repräsentation von Daten

1. Repräsentation von natürlichen Zahlen ✓
2. Repräsentation von Texten ✓
3. Repräsentation ganzer Zahlen ✓
4. Repräsentation rationaler Zahlen ✓
5. Repräsentation anderer Daten ✓