technische universität
dortmund

Rechnerstrukturen WS 2012/13

- Repräsentation von Daten
  - Repräsentation natürlicher Zahlen (Wiederholung)
  - Repräsentation von Texten
  - Repräsentation ganzer Zahlen
  - Repräsentation rationaler Zahlen
  - Repräsentation anderer Daten

Hinweis: *Folien teilweise a. d. Basis von Materialien von Thomas Jansen*

9. Oktober 2012

b-adische Darstellung – Eigenschaften

Theorem

Für jede Basis $b \in \mathbb{N} \setminus \{1\}$ *und jede natürliche Zahl* $n \in \mathbb{N}$ *lässt sich* n *eindeutig darstellen als*

$$n = n_0 \cdot b^0 + n_1 \cdot b^1 + \cdots + n_l \cdot b^l = \sum_{i=0}^{l} n_i \cdot b^i$$

mit $l \in \mathbb{N}_0$, $n_0, n_1, \ldots, n_l \in \{0, 1, \ldots, b-1\}$ *und* $n_l > 0$.

Umrechnung Basis 10 $\rightsquigarrow$ Basis b

Algorithmus 1

Eingabe Basis $b \in \mathbb{N} \setminus \{1\}$, Betragszahl $n \in \mathbb{N}$

Ausgabe Repräsentation $(n_l\, n_{l-1} \cdots n_1\, n_0)_b$

1. $l := -1$
2. So lange $n > 0$ gilt
3. $\quad l := l + 1$
4. $\quad n_l := n - b \cdot \lfloor n/b \rfloor$
5. $\quad n := \lfloor n/b \rfloor$

b	16	16	16	16	16
n	38 835	2 427	151	9	0
l	-1	0	1	2	3
n_0		3	3	3	3
n_1			B	B	B
n_2				7	7
n_3					9

also $38\,835 = (9\,7\,\mathrm{B}\,3)_{16}$

Umrechnung Basis 10 $\rightsquigarrow$ Basis b

Algorithmus 1

Eingabe Basis $b \in \mathbb{N} \setminus \{1\}$, Betragszahl $n \in \mathbb{N}$

Ausgabe Repräsentation $(n_l\, n_{l-1} \cdots n_1\, n_0)_b$

1. $l := -1$
2. So lange $n > 0$ gilt
3. $\quad l := l + 1$
4. $\quad n_l := n - b \cdot \lfloor n/b \rfloor$
5. $\quad n := \lfloor n/b \rfloor$

Definition

Unter einem *Algorithmus* (auch Lösungsverfahren) versteht man eine genau definierte Handlungsvorschrift zur Lösung eines Problems oder einer bestimmten Art von Problemen in endlich vielen Schritten.

Repräsentation von Texten

„Wir können alles aufschreiben.“

$\Rightarrow$ Repräsentation von Texten zentral

naiver Ansatz Codierung A $\rightsquigarrow$ 0, B $\rightsquigarrow$ 1, ...
und dann Binärcodierung

Probleme

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

Lösung Standards

ASCII

American Standard Code for Information Interchange

- verabschiedet 1963 von der American Standards Organization (ASO)
- 7 Bit Code
- für die USA gedacht
- codiert Zeichen und Steuercodes

ASCII

	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	{	|	}	∼	DEL

ASCII

Problem viele wichtige Zeichen fehlen

Lösung „längere Codierung“, Code erweitern

Anmerkung Erweitern ist besser als ersetzen.

bekannteste Erweiterung hier ISO-8859-1
auch ISO Latin 1 genannt

ISO-8859-1 (ISO Latin 1)

International Organization for Standardization
(gegründet 1947)

ISO Latin 1

- 8 Bit Code (8 Bits nennt man 1 Byte)
- enthält viele Sonderzeichen für westeuropäische Sprachen (z. B. Umlaute (ä, ö, ü, ...))
- enthält nicht alle gewünschten Zeichen (z. B. €, japanische Schriftzeichen, ...)

Unicode

aktueller Standard Unicode 6.2 (26. September 2012)

- verwaltet vom Unicode-Konsortium (`http://www.unicode.org`)
- unterstützt verschiedene Codierungsformate (Unicode Transformation Format): UTF-8, UTF-16, UTF-32 mit 8, 16, 32 Bits
- längere Formate erweitern kürzere Formate
- vereinbart auch weitere Informationen (z. B. Schreibrichtung, Kombination von Zeichen (Codepoints))

Repräsentation ganzer Zahlen

Wunsch ganze Zahlen $z \in \mathbb{Z}$ repräsentieren können

Vereinbarung feste Repräsentationslänge l

Wie viele verschiedene Zahlen kann man so höchstens repräsentieren?

l Positionen, jeweils 2 Möglichkeiten

$\Rightarrow 2^l$ verschiedene Bitmuster der Länge l

Wie Zahlenbereich verwenden – positive und negative Zahlen unterscheiden?

Vorzeichenbetragsmethode

erstes Bit	Vorzeichen
restliche Bits	Betragszahl (Binärdarstellung)

Vorzeichenbit	$s = 1 \Leftrightarrow$ Vorzeichen negativ
Begründung	„Zahl $= (-1)^s \cdot$ Betragszahl“

kleinste darstellbare Zahl	$-\left(2^{l-1} - 1\right)$
größte darstellbare Zahl	$2^{l-1} - 1$

Eigenschaften

- \+ symmetrisch
- \+ Vorzeichenwechsel einfach
- – 0 nicht eindeutig
- – Vergleich von Zahlen schwierig

Darstellung mit Bias (Exzessdarstellung)

Wähle feste Verschiebung b (**Bias**).

z wird als $z + b$ dargestellt (Binärdarstellung)

übliche Werte für Bias $b = 2^{l-1}$ oder $b = 2^{l-1} - 1$

kleinste darstellbare Zahl	$-b$
größte darstellbare Zahl	$2^l - 1 - b$

Eigenschaften

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

Einerkomplementdarstellung

nicht-negative Zahlen	Binärdarstellung
negative Zahlen	Komplement der Binärdarstellung

kleinste darstellbare Zahl	$-\left(2^{l-1}-1\right)$
größte darstellbare Zahl	$2^{l-1}-1$

Eigenschaften

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

Zweierkomplementdarstellung

nicht-negative Zahlen	Binärdarstellung
negative Zahlen	„(Komplement der Binärdarstellung) + 1“

kleinste darstellbare Zahl	$-\left(2^{l-1}\right)$
größte darstellbare Zahl	$2^{l-1}-1$

Eigenschaften

- \+ 0 eindeutig repräsentiert
- \+ erstes Bit wie Vorzeichenbit
- \+ alle Codes ausgenutzt
- – Vergleich von Zahlen schwierig
- ▶ üblichste Darstellung

Zweierkomplementdarstellung II

Beachte: Ex. geschlossene Form zur Berechnung der im Zweierkomplement repräsentierten Zahl

Theorem

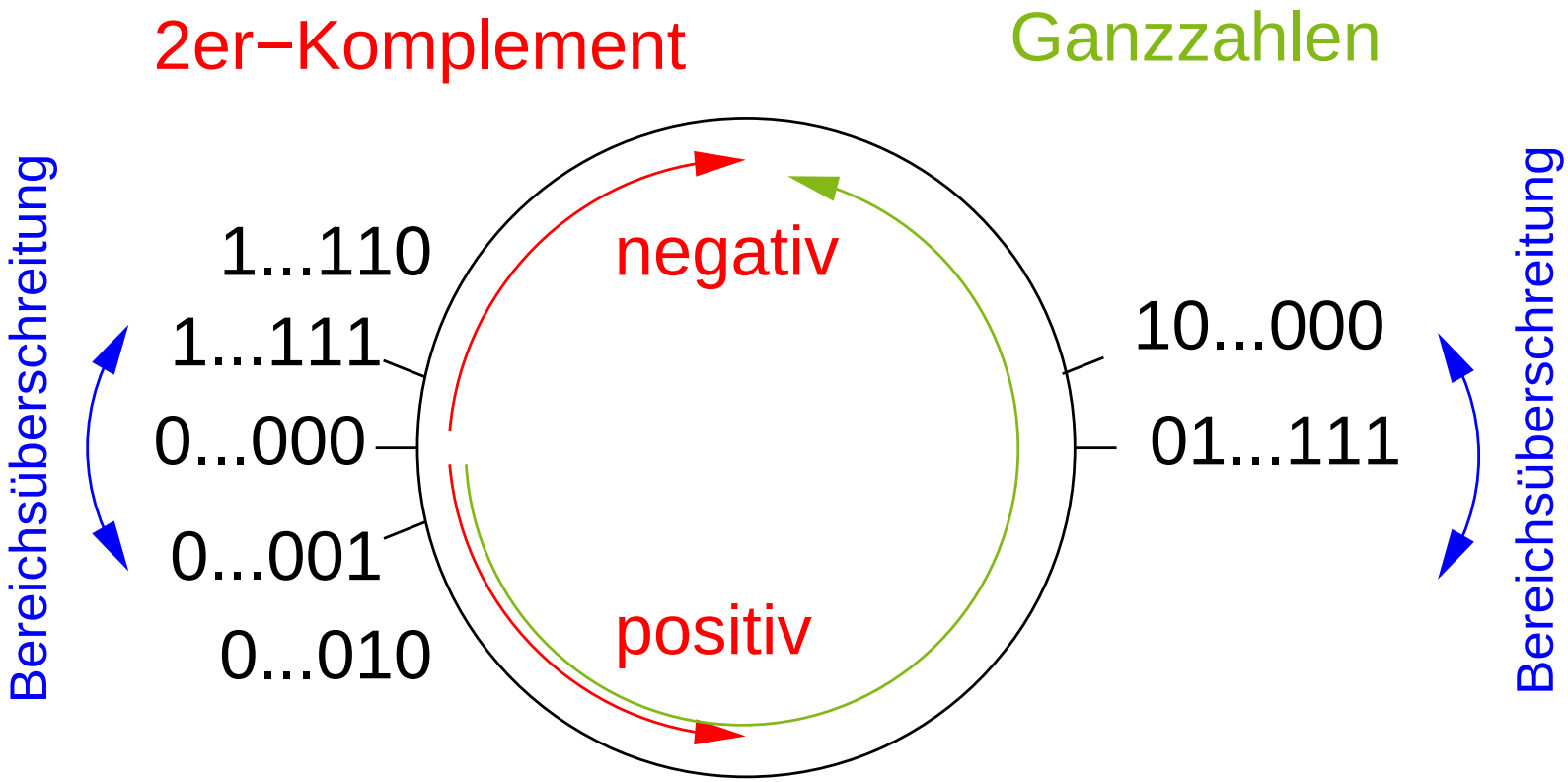
Ein Bitvektor $a = (a_{n-1}, ..., a_0)$ repräsentiert bei einer Kodierung im Zweierkomplement die Zahl

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

Z.B.: $\text{int}(1101) = -1 \cdot 2^3 + \left\{1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2\right\} = (-3)_{10}$

Zweierkomplementdarstellung III

Hinweis: Für die Zahlendarstellung im Zweierkomplement ergibt sich folgendes "Zahlenrad"



Alles verstanden?

haben $l = 5,\ b = 2^{l-1} = 2^4 = 16,\ z = 10010$

Welche Zahl wird denn nun repräsentiert?

Beobachtung Das hängt von der Darstellung ab.

Vorzeichenbetragsdarstellung

$$\underbrace{1}_{\text{Vorzeichen}}\ \underbrace{0010}_{\text{Betrag}}$$

$(0010)_2 = 0 \cdot 2^0 + 1 \cdot 2^1 = 2$

also $(-1)^1 \cdot 2 = -2$

Alles verstanden?

haben $l = 5,\ b = 2^{l-1} = 2^4 = 16,\ z = 10010$

Welche Zahl wird denn nun repräsentiert?

Beobachtung Das hängt von der Darstellung ab.

Darstellung mit Bias $b = 16$ (Exzessdarstellung)

$(10010)_2 = 0 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 0 \cdot 2^3 + 1 \cdot 2^4 = 2 + 16 = 18$

also $\underbrace{18}_{\text{Zahl}} - \underbrace{16}_{\text{Bias}} = 2$

Alles verstanden?

haben $l = 5,\ b = 2^{l-1} = 2^4 = 16,\ z = 10010$

Welche Zahl wird denn nun repräsentiert?

Beobachtung Das hängt von der Darstellung ab.

Einerkomplementdarstellung

erstes Bit 1, also negativ
Komplement $\overline{10010} = 01101$
$(01101)_2 = 1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 + 1 \cdot 2^3 = 1 + 4 + 8 = 13$
also -13

Alles verstanden?

haben $l = 5,\ b = 2^{l-1} = 2^4 = 16,\ z = 10010$

Welche Zahl wird denn nun repräsentiert?

Beobachtung Das hängt von der Darstellung ab.

Zweierkomplementdarstellung

erstes Bit 1, also negativ

$10010 - 1 = 10001$

Komplement $\overline{10001} = 01110$

$(01110)_2 = 0 \cdot 2^0 + 1 \cdot 2^1 + 1 \cdot 2^2 + 1 \cdot 2^3 = 2 + 4 + 8 = 14$

also -14

Beispielsammlung Darstellungen

feste Länge $l = 5$, ($2^l = 32$, $2^{l-1} = 16$, $2^{l-1} - 1 = 15$)

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

Repräsentation rationaler Zahlen

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

Vereinbarung feste Repräsentationslänge l Bits

Idee wenn schon feste Darstellungslänge,
dann auch feste Position des Kommas

Beispiele $l = 8$, 3 Nachkommastellen

$10{,}34 \rightsquigarrow 00010{,}340$

$93\,847{,}123 \rightsquigarrow 93\,847{,}123$

$0 \rightsquigarrow 00000{,}000$

$123\,456{,}78 \rightsquigarrow$ nicht darstellbar

$12{,}345678 \rightsquigarrow$ nicht darstellbar, aber runden denkbar

Festkommazahlen

klar Basis 10 für uns nicht brauchbar

zum Glück Übertragung auf Basis 2 ganz einfach

Beispiel

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

$= 22{,}625$

allgemein bei v Vorkomma- und m Nachkommastellen

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

Binärdarstellung rationaler Zahlen

Beobachtung unabhängig von Darstellungslänge
nicht alle Zahlen darstellbar

Beispiel 0,2 nicht als binäre Festkommazahl darstellbar

Beweis durch Widerspruch (Annahme Doch!)

dann $0{,}2 = \frac{1}{5} = \sum_{i=-m}^{-1} z_i \cdot 2^i$ mit $m \in \mathbb{N}$, $z_i \in \{0, 1\}$

$$\text{also } \frac{1}{5} = \sum_{j=1}^{m} z_{-j} \cdot 2^{-j} = 2^{-m} \sum_{j=1}^{m} z_{-j} \cdot 2^{m-j} \quad | \text{ mit } k = m - j$$

$$= 2^{-m} \sum_{k=0}^{m-1} z_{k-m} \cdot 2^k = \frac{a}{2^m} \text{ mit } a, m \in \mathbb{N}$$

also $5a = 2^m$ Widerspruch

Ist die Binärdarstellung schlecht?

Nein! Es gibt für jede Basis nicht darstellbare rationale Zahlen, die in anderen Basen darstellbar sind.

Beispiel $\frac{1}{3} = (0{,}1)_3$, zur Basis 10 nicht darstellbar

Ist Festkommadarstellung schlecht?

jedenfalls unflexibel

eher für spezielle Anwendungen geeignet

Gleitkommazahlen

andere Idee Position des Kommas nach Bedarf

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

mit $s \in \{0, 1\}$, $e \in \mathbb{Z}$, $m \in \mathbb{Q}$ und $1 \leq m < 10$

- Exponent e bestimmt Position des Kommas
- „Mantisse“ m

normalisierte Gleitkommazahlendarstellung

Gleitkommazahlen

für uns besser Basis 2

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

mit $s \in \{0, 1\}$, $e \in \mathbb{Z}$, $m \in \mathbb{Q}$ und $1 \leq m < 2$

Beobachtungen

- „Mantisse“ hat immer erste Ziffer 1
- 0 ist so nicht darstellbar

immer besser an Standards halten

IEEE-754 1985

Institute of Electrical & Electronics Engineers

$z = (-1)^s \cdot m \cdot 2^e$
mit $s \in \{0, 1\}$, $e \in \mathbb{Z}$, $m \in \mathbb{Q}$ und $1 \leq m < 2$

Festlegungen

- führende 1 der „Mantisse“ wird nicht mit abgespeichert (heißt „implizite Eins“)
- Mantisse in Binärcodierung (Festkommazahlen mit Ziffern ausschließlich hinter dem Komma)
- Exponent in Exzessdarstellung mit Bias $b = 2^{l_e - 1} - 1$

IEEE 754-1985

feste Anzahlen von Bits

Gesamtlänge	Vorzeichen	Exponent	Mantisse	*Precision*
32	1	8	23	*single*
64	1	11	52	*double*

Hinweis: In IEEE 754-2008 auch 16/128 bit Binär- sowie 32/64 bit Dezimalformate

Besonderheiten beim Exponenten

Verkleinerung des zulässigen Bereichs um 1 auf beiden Seiten
auf $e_{\min} = -b + 1 = -\left(2^{l_e - 1} - 1\right) + 1$,
und $e_{\max} = 2^{l_e} - 1 - b - 1 = 2^{l_e - 1} - 1$

dient der Codierung „besonderer Zahlen“

IEEE 754-1985: ein Beispiel

$l = 32,\ l_s = 1,\ l_e = 8,\ l_m = 23$
$b = 2^7 - 1 = 127,\ e_{\min} = -b + 1 = -126,\ e_{\max} = 2^8 - b - 2 = 127$

Wir wollen -3 darstellen.

negativ, also Vorzeichenbit 1

Darstellung als Summe von Zweierpotenzen
$3 = 2 + 1 = 2^1 + 2^0 = \left(2^0 + 2^{-1}\right) \cdot 2^1$

Exponent 1 Exzessdarstellung $1 + b = 128$ darstellen

$128 = (1000\,0000)_2$

Mantisse 1,1, implizite Eins entfällt, also $100\,0000\,0000\cdots$

Ergebnis $\underbrace{1}_{\text{VZ}}\ \underbrace{1000\,0000}_{\text{Exponent}}\ \underbrace{100\,0000\,0000\,0000\,0000\,0000}_{\text{Mantisse}}$

IEEE 754-1985: noch ein Beispiel

$l = 32,\ l_s = 1,\ l_e = 8,\ l_m = 23$
$b = 2^7 - 1 = 127,\ e_{\min} = -b + 1 = -126,\ e_{\max} = 2^8 - b - 2 = 127$

Wir wollen 0,0546875 darstellen.

positiv, also Vorzeichenbit 0

Darstellung als Summe von Zweierpotenzen
$0{,}0546875 = \frac{1}{32} + \frac{1}{64} + \frac{1}{128} = 2^{-5} + 2^{-6} + 2^{-7} = \left(2^0 + 2^{-1} + 2^{-2}\right) \cdot 2^{-5}$

Exponent -5 Exzessdarstellung $-5 + b = 122$ darstellen
$122 = (0111\,1010)_2$

Mantisse 1,11, implizite Eins entfällt, also $110\,0000\,0000\cdots$

Ergebnis $\underbrace{0}_{\text{VZ}}\ \underbrace{0111\,1010}_{\text{Exponent}}\ \underbrace{110\,0000\,0000\,0000\,0000\,0000}_{\text{Mantisse}}$

IEEE 754-1985: ein letztes Beispiel

$l = 32,\ l_s = 1,\ l_e = 8,\ l_m = 23$
$b = 2^7 - 1 = 127,\ e_{\min} = -b + 1 = -126,\ e_{\max} = 2^8 - b - 2 = 127$

Wir haben $0\,1011\,0001\,010\,1001\,0000\,0000\,0000\,0000$.

$$\underbrace{0}_{\text{VZ}}\ \underbrace{1011\,0001}_{\text{Exponent}}\ \underbrace{010\,1001\,0000\,0000\,0000\,0000}_{\text{Mantisse}}$$

Vorzeichenbit 0, also positiv

Exponent $(1011\,0001)_2 = 177$, stellt $177 - b = 177 - 127 = 50$ dar

Mantisse zuzüglich impliziter Eins 1,0101001
$(1{,}0101001)_2 = 1 + \frac{1}{4} + \frac{1}{16} + \frac{1}{128}$

also $\left(2^0 + 2^{-2} + 2^{-4} + 2^{-7}\right) \cdot 2^{50} = 2^{50} + 2^{48} + 2^{46} + 2^{43}$
$= 1\,486\,539\,720\,753\,152$

IEEE 754-1985: Besondere Zahlen

VZ	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	$e_{min}-1$	$m \neq 0$	$(-1)^s \cdot \left(\sum_{i=1}^{l_m} m_i \cdot 2^{-i}\right) \cdot 2^{e_{min}}$

NaN not a number

Welches Bitmuster repräsentiert die Null?

letzte Zeile denormalisierte Darstellung

Wozu denormalisierte Zahlen?

klar noch kleinere Zahlen darstellbar

Beispiel `If` $x \neq y$ `Then` $z := 1/(x-y)$

$x = 0\,0000\,0001\,000\,0000\,0000\,0000\,0000\,0001 \quad 2^{-126} + 2^{-149}$

$y = 0\,0000\,0001\,000\,0000\,0000\,0000\,0000\,0000 \quad 2^{-126}$

klar $x \neq y$

aber ohne denormalisierte Darstellung $x - y = 0$ gerundet

denormalisiert $x - y = 2^{-149}$:
$0\,0000\,0000\,000\,0000\,0000\,0000\,0000\,0001$

auf jeden Fall besser `If` $x - y \neq 0$ `Then` $z := 1/(x-y)$

Repräsentation anderer Daten

hier für uns interessant „primitive Daten“
(von Hardware direkt unterstützt)

zentral Programme
in der Regel Bitmuster fester Länge (z.B. 4 Byte): Befehl, Operand

Problem Was repräsentiert ein Byte im Speicher?

kaum verwendet: Typbits

Repräsentation von Datenfolgen

Speicher oft in Worten organisiert

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

heterogene Daten hintereinander in den Speicher schreiben
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