Universität Konstanz

Konzepte der Informatik

Informationscodierung und -speicherung

Barbara Pampel

Universität Konstanz, WiSe 2023/2024

Inhalt

Zahlensysteme

Zahlensysteme

- Unäre Zahlensysteme
  - Jede natürliche Zahle wird durch eine Anzahl von Zeichen (Symbolen) repräsentiert z.B. „Strichsystem”

Zahlensysteme

- Unäre Zahlensysteme
  - Jede natürliche Zahle wird durch eine Anzahl von Zeichen (Symbolen) repräsentiert z.B. „Strichsystem”
- Stellenwertsysteme
  - bestehen aus endlich vielen Ziffernsymbolen
  - Vorschrift, wie man aus der Reihung der Symbole beliebig große Zahlen darstellen kann

Zahlensysteme

- Unäre Zahlensysteme
  - Jede natürliche Zahle wird durch eine Anzahl von Zeichen (Symbolen) repräsentiert z.B. „Strichsystem"
- Stellenwertsysteme
  - bestehen aus endlich vielen Ziffernsymbolen
  - Vorschrift, wie man aus der Reihung der Symbole beliebig große Zahlen darstellen kann
- Beispiele
  - Römisches Zahlensystem
    - hybrid — teilweise unär, Stelle hat aber auch Bedeutung
    - III = 3, MMXI = 2011, MCMLXXIX = 1979
  - Arabisches Zahlensystem (modern-arabisch)
    - Stellenwertsystem
    - nur 10 verschiedene Ziffern, inklusive Darstellung für 0

Arabisches Zahlensystem

- Stellenwertsysteme zur Basis β
- Basis β wird auch *Radix* genannt
  - Radix 10: Dezimaldarstellung
  - Radix 2: Binärdarstellung
  - Radix 8: Oktaldarstellung
  - Radix 16: Hexadezimaldarstellung
- Kennzeichnung der Basis β im Zweifelsfall als Subskript
  - $7_{10} = 7_8 = 111_2$
  - $9_{10} = 11_8 = 1001_2$

Umrechnungen zwischen Zahlensystemen I

Umrechnungen zwischen Zahlensystemen I

- Überführung einer Oktalzahl ins Binärsystem
  - Ziffern $0 - 7$ genau mit 3 Binärstellen darstellbar

  $\Rightarrow$ jede einzelne Ziffer als $3-$stellige Binärzahl angeben

Umrechnungen zwischen Zahlensystemen I

- Überführung einer Oktalzahl ins Binärsystem
  - Ziffern $0-7$ genau mit 3 Binärstellen darstellbar
  - $\Rightarrow$ jede einzelne Ziffer als $3-$stellige Binärzahl angeben
- Überführung einer Hexadezimal ins Binärsystem
  - Ziffern $0-9$ und Buchstaben $A-F$ mit genau 4 Binärstellen darstellbar
  - $\Rightarrow$ jede einzelne Ziffer als $4-$stellige Binärzahl angeben

Beispiele

- $2345_8 = 010011100101_2$
- $65C7_{16} = 0110010111000111_2$

Umrechnungen zwischen Zahlensystemen II

- Überführung einer l-stelligen Binärzahl $n \geq 0$ ins Dezimalsystem

Umrechnungen zwischen Zahlensystemen II

- Überführung einer l-stelligen Binärzahl $n \geq 0$ ins Dezimalsystem (n_k bezeichne die $k-$te Stelle von n)

Algorithmus

Schritt 1: $k := 0$; $sum := 0$
Schritt 2: $sum := sum + n_k \cdot 2^k$; $k := k + 1$
Schritt 3: Falls $k < l$ *gehe zu Schritt 2*
Schritt 4: Gib sum *aus*

Umrechnungen zwischen Zahlensystemen II

- Überführung einer l-stelligen Binärzahl $n \geq 0$ ins Dezimalsystem (n_k bezeichne die $k-$te Stelle von n)

Algorithmus

Schritt 1: $k := 0$; $sum := 0$
Schritt 2: $sum := sum + n_k \cdot 2^k$; $k := k + 1$
Schritt 3: Falls $k < l$ *gehe zu Schritt 2*
Schritt 4: Gib sum *aus*

- Allgemein für l-stellige Zahl n zur Basis β
  - $\sum_{k=0}^{l-1} n_k \cdot \beta^k$
- Nachkommastellen analog
- Bei Umrechnung zwischen anderen Systemen empfiehlt sich der Umweg über das Dezimalsystem

Beispiele

Wert einer Dezimalzahl

- $234_{10} = 4 \cdot 10^0 + 3 \cdot 10^1 + 2 \cdot 10^2$

Oktal nach Dezimal

- $657_8 = 7 \cdot 8^0 + 5 \cdot 8^1 + 6 \cdot 8^2 = 431_{10}$

Hexadezimal nach Dezimal

- $4B6_{16} = 6 \cdot 16^0 + 11 \cdot 16^1 + 4 \cdot 16^2 = 1\ 206_{10}$

Umrechnungen zwischen Zahlensystemen III

- Überführung von Dezimalzahl $n \geq 0$ zu Binärzahl

Umrechnungen zwischen Zahlensystemen III

- Überführung von Dezimalzahl $n \geq 0$ zu Binärzahl
  (x_k bezeichne die $k-$te Stelle der Binärzahl)

Algorithmus

Schritt 1: $k := 0$
Schritt 2: $x_k := n \mod 2$*;* $n := \lfloor \frac{n}{2} \rfloor$*;* $k := k + 1$
Schritt 3: Falls $n > 0$ *gehe zu Schritt 2*
Schritt 4: Gib $x_{k-1} ... x_0$ *aus*

Umrechnungen zwischen Zahlensystemen IV

- Nachkommastellen n zu Binärzahl $0, x_{-1}x_{-2}x_{-3}\ldots$

Algorithmus

Schritt 1: $k := -1$
Schritt 2: $x_k := \lfloor n \cdot 2 \rfloor$; $n := n \cdot 2 - x_k$; $k := k - 1$
Schritt 3: Falls $n > 0$ *gehe zu Schritt 2*
Schritt 4: Gib $0, x_{-1}x_{-2}\ldots x_{k+1}$ *aus*

Umrechnungen zwischen Zahlensystemen V

- Überführung von Dezimalzahl in $n \geq 0$ zu Hexadezimalzahl $..x_2x_1x_0$
- Für $x_k > 9$
  - $10_{10} =$ A$_{16}$
  - $11_{10} =$ B$_{16}$
  - ...

Algorithmus

Schritt 1: $k := 0$
Schritt 2: $x_k := n \mod 16$; $n := \lfloor \frac{n}{16} \rfloor; k := k + 1$
Schritt 3: Falls $n > 0$ *gehe zu Schritt 2*
Schritt 4: Gib $x_{k-1}..x_2x_1x_0$ *aus*

- Umrechung analog für alle anderen natürlichen Zahlensysteme

Kodierung von Zahlen I

- Repräsentation im Rechner immer als Binärzahl (fester Länge)

Beispiele

- $133,25_{10} = 1 \cdot 2^7 + 1 \cdot 2^2 + 1 \cdot 2^0 + 1 \cdot 2^{-2} = 10000101,01_2$
- $1,1_{10} =$

Kodierung von Zahlen I

- Oktal- und Hexadezimalzahlen gruppieren Binärzahlen

Beispiele

- Dreiergruppen im Oktalsystem: $010|000|101,010 = 205,2_8$
- Vierergruppen im Hexadezimalsystem: $1100|0101,0100 = \mathrm{C}5,4_{16}$

- Repräsentation im Rechner immer als Binärzahl (fester Länge)

Beispiele

- $133,25_{10} = 1 \cdot 2^7 + 1 \cdot 2^2 + 1 \cdot 2^0 + 1 \cdot 2^{-2} = 1000\ 0101,01_2$
- $1,1_{10} = 1,00011001100110\ldots_2$

- BCD = binary coded decimal
  - jede Dezimalziffer wird durch vier Binärziffern dargestellt

Kodierung von Zahlen II

- Vorteile: exakte Überführung zwischen dezimal und binär möglich
- Nachteil: Verschwendung von sechs Werten

Beispiele

- $133,25_{10} = 0001\ 0011\ 0011,0010\ 0101_{BCD}$
- $1,1_{10} = 0001,0001_{BCD}$

Negative Zahlen I

Negative Zahlen I

- Verschiebung des Nullpunkts in die Mitte der k Stellen
  - $z' = z - 2^{k-1} + 1$
  - feste Verschiebung für alle Zahlen, viele Werte nicht nutzbar
  - einfache binäre Arithmetik funktioniert nicht mehr

Negative Zahlen I

- Verschiebung des Nullpunkts in die Mitte der k Stellen
  - $z' = z - 2^{k-1} + 1$
  - feste Verschiebung für alle Zahlen, viele Werte nicht nutzbar
  - einfache binäre Arithmetik funktioniert nicht mehr
- Einerkomplement
  - höchstwertiges Bit gibt Vorzeichen an
  - $k - 1$ Stellen zur Darstellung des Betrags
  - Negation durch Invertieren der Zahl (bitweises Vertauschen von 0 und 1)

Beispiel

37 = 0010 0101, -37 = 1101 1010

  - zwei Darstellungen für 0: $0000\ 0000 \equiv 1111\ 1111$
  - Subtraktion als gesonderte Rechenoperation

Beispiele Einerkomplement (1 Byte)

$$
\begin{aligned}
32_{10} &= 0010\ 0000_2 \\
-37_{10} &= 1101\ 1010_2 \\
& \ 1111\ 1010_2 = -5_{10} \\
37_{10} &= 0010\ 0101_2 \\
-37_{10} &= 1101\ 1010_2 \\
& \ 1111\ 1111_2 = 0_{10} \\
38_{10} &= 0010\ 0110_2 \\
-37_{10} &= 1101\ 1010_2 \\
& \ 0000\ 0000_2 = 0_{10} \\
45_{10} &= 0010\ 1101_2 \\
-37_{10} &= 1101\ 1010_2 \\
& \ 0000\ 0111_2 = 7_{10}
\end{aligned}
$$

Negative Zahlen II

- Zweierkomplement
  - Invertieren der Zahl UND anschließend Addition von 1

Beispiel

-37	Positive Darstellung	$37_{10} = 0010\ 0101_2$
	Invertieren	$1101\ 1010_2$
	Addition von 1	$1101\ 1011_2 = -37_{10}$

  - nur eine Darstellung für 0
  - Subtraktion durch Addition durchführbar

Negative Zahlen II

- Zweierkomplement
  - Invertieren der Zahl UND anschließend Addition von 1

Beispiel

-37

Positive Darstellung	$37_{10} = 0010\ 0101_2$
Invertieren	$1101\ 1010_2$
Addition von 1	$1101\ 1011_2 = -37_{10}$

  - nur eine Darstellung für 0
  - Subtraktion durch Addition durchführbar

$$\begin{aligned} 45_{10} &= 0010\ 1101_2 \\ -37_{10} &= 1101\ 1011_2 \\ & \ 0000\ 1000_2 = 8_{10} \end{aligned}$$

Zweierkomplement