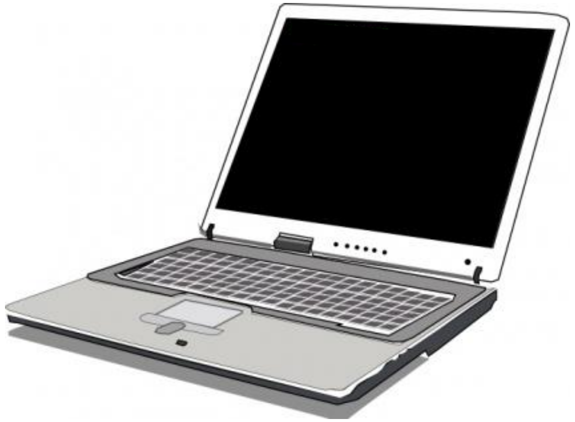
# DIGITALE SCHALTUNGEN

Robert Wille (robert.wille@jku.at)
Sebastian Pointner (sebastian.pointner@jku.at)

Institut für Integrierte Schaltungen
Abteilung für Schaltkreis- und Systementwurf

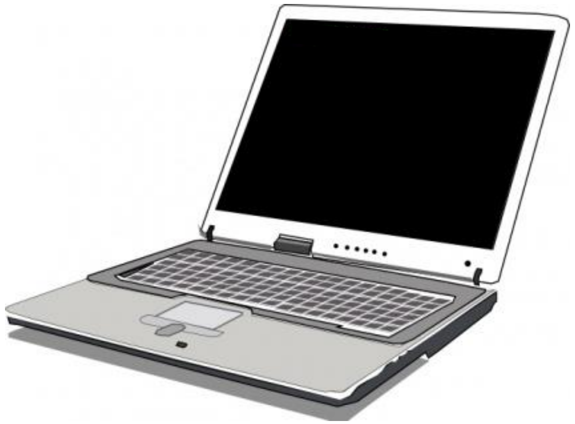
# INHALT DER VORLESUNG



=


- **Grundlagen**
  - ☐ Beschreibungen über „0“ und „1“ (Boolesche Algebra)
  - ☐ Beschreibungen von Schaltungen
- **Rechnen**
  - ☐ Darstellung von Zahlen
  - ☐ Digitale Schaltungen für Addition, Subtraktion, Multiplikation
- **Speichern**
  - ☐ Sequentielle Schaltungen
  - ☐ Speicherelemente
- **Steuern**
  - ☐ Endliche Automaten
  - ☐ Synthese von Steuerwerken
- **Entwerfen**
  - ☐ Synthese von allgemeinen Schaltungen
  - ☐ Logikminimierung

# INHALT DER VORLESUNG



=


- **Grundlagen**
  - Beschreibungen über „0“ und „1“ (Boolesche Algebra)
  - Beschreibungen von Schaltungen
- **Rechnen**
  - **Darstellung von Zahlen**
  - Digitale Schaltungen für Addition, Subtraktion, Multiplikation
- **Speichern**
  - Sequentielle Schaltungen
  - Speicherelemente
- **Steuern**
  - Endliche Automaten
  - Synthese von Steuerwerken
- **Entwerfen**
  - Synthese von allgemeinen Schaltungen
  - Logikminimierung

# ZAHLENDARSTELLUNG

Robert Wille
Sebastian Pointner

robert.wille@jku.at
sebastian.pointner@jku.at

# DEZIMALSYSTEM

- Jede Stelle kann die Werte von 0 bis 9 annehmen
- Stellenwert hängt von der Position in der Zahl ab
- 7689 $= 7*10^3 + 6*10^2 + 8*10^1 + 9*10^0$
  $= 7*1000 + 6*100 + 8*10 + 9*1$

**Formel:** $Z = \sum_i 10^i * a_i, \{a_i \in \mathbb{N} \wedge 0 \leq a_i \leq 9\}$

JKU

# ALLGEMEINE STELLENWERTSYSTEME

- Jede Zahl > 1 kann Basis sein!
  (Meistens jedoch nur ganze Zahlen)
- Häufig verwendete Basen:
  - 2: Binär- oder Dualsystem
  - 8: Oktalsystem
  - 16: Hexadezimal- (oder auch Sedezimal) -system

**Formel:**

$$Z = \sum_i B^i * a_i, \{a_i \in \mathbb{N} \land 0 \leq a_i \leq B - 1\}$$

# ALLGEMEINE STELLENWERTSYSTEME

- Gebräuchliche Zahlensysteme

| Name des Systems | Basis | Alphabet |
|---|---|---|
| dual | 2 | {0,1} |
| oktal | 8 | {0,1,...,7} |
| dezimal | 10 | {0,1,...,9} |
| hexadezimal | 16 | {0,1,...,9, A, B, C, D, E, F} |

- Darstellung
  - Anhängen der tiefgestellten Basis an die Zahl
  - Oktalsystem häufig durch O symbolisiert
  - Hexadezimal häufig durch H symbolisiert
  - Beispiele: $1001_2$, $7345_{10}$, $7345_O$, $CAFE_H$

# DUALSYSTEM

- Jede Stelle kann die Werte von 0 und 1 annehmen

→ Praktisch für digitale Systeme (Computer)

- Stellenwert hängt von der Position in der Zahl ab
- 1101 $= 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0$

  $= 1*8 + 1*4 + 0*2 + 1*1$

**Formel:**

$$Z = \sum_i 2^i * a_i, \{a_i \in \mathrm{B}\}$$

JKU

# BEISPIELE

- $CAFE_H = 12*16^3 + 10*16^2 + 15*16^1 + 14*16^0$
  $= 12*4096 + 10*256 + 15*16 +14$
  $= 51966_{10}$

- $7345_O = 7*8^3 + 3*8^2 + 4*8^1 + 5*8^0$
  $= 7*512 + 3*64 + 4*8 + 5$
  $= 3813_{10}$

- $1001_2 = 1*2^3 + 0*2^2 + 0*2^1 + 1*2^0$
  $= 1*8 + 0 + 0 + 1$
  $= 9_{10}$

# RECHNEN MIT ANDEREN BASEN

- Rechenoperationen lassen sich auf andere Basen übertragen
- Beispiel Addition:

**Dezimalsystem:**

| | |
|---|---|
| Summand1 | 1 0 5 9 |
| Summand2 | 3 4 6 2 |
| Übertrag | 0 1 1 |
| Summe | 4 5 2 1 |

**Hexadezimalsystem:**

| | |
|---|---|
| Summand1 | 7 3 A F |
| Summand2 | 1 E B 8 |
| Übertrag | 1 1 1 |
| Summe | 9 2 6 7 |

# RECHNEN MIT ANDEREN BASEN

- Rechenoperationen lassen sich auf andere Basen übertragen

- Beispiel Multiplikation:

```
  2 3 * 5 A 7 = C 5 D 5
  ---------------
  A F
  1 5 E
      F 5
  ---------------
  C 5 D 5
```

- Beispiel Division:

```
   4 3 A F : 7 E = 8 9   Rest  4 1
 - 3 F 0
   -------
     4 A F
   - 4 6 E
     -----
       4 1
```

# BASISUMWANDLUNG

- Umwandlung aus Darstellung mit Ausgangsbasis in Darstellung mit Zielbasis
- Zwei Varianten möglich

# BASISUMWANDLUNG – VARIANTE 1

1. Jede Stelle der Zahl mit dem Stellenwert in Zieldarstellung multiplizieren
2. Teile aufsummieren

➔ Besonders gut geeignet, wenn Zielbasis die „natürliche" Basis ist

- Beispiele:
  - Binär nach dezimal:
    $10011_2 = 1*2^4 + 1*2^1 + 1*2^0 = 16 + 2 + 1 = 19_{10}$
  - Dezimal nach heximal:
    $327_{10} = 3*64_H + 2*A_H + 7 = 12C_H + 14_H + 7 = 147_H$
  - Dezimal nach oktal:
    $327_{10} = 3*144_O + 2*12_O + 7 = 454_O + 24_O + 7 = 507_O$

# BASISUMWANDLUNG – VARIANTE 2

1. Zahl durch Zielbasis dividieren
2. Rest ist niederwertigste Stelle des Ergebnisses
3. Quotienten durch Zielbasis dividieren
4. Rest ist nächste Stelle des Ergebnisses
5. Solange bei 3. fortsetzen, bis Quotient 0

➔ Besonders gut geeignet, wenn Ausgangsbasis die „natürliche" Basis ist

- Beispiele:
  - Von dezimal nach oktal:
    $327_{10} \rightarrow 507_{O}$

| | | | $8^2$ | $8^1$ | $8^0$ |
|---|---|---|---|---|---|
| $327_{10}$ | $: 8 = 40$ | Rest | | | 7 |
| $40_{10}$ | $: 8 = 5$ | Rest | | 0 | |
| $5_{10}$ | $: 8 = 0$ | Rest | 5 | | |

# BASISUMWANDLUNG – VARIANTE 2

1. Zahl durch Zielbasis dividieren
2. Rest ist niederwertigste Stelle des Ergebnisses
3. Quotienten durch Zielbasis dividieren
4. Rest ist nächste Stelle des Ergebnisses
5. Solange bei 3. fortsetzen, bis Quotient 0

➔ Besonders gut geeignet, wenn Ausgangsbasis die „natürliche" Basis ist

- Beispiele:
  - Von dezimal nach hexadezimal:

    $327_{10} \Rightarrow 147_H$

| | | $16^2$ | $16^1$ | $16^0$ |
|---|---|---|---|---|
| $327_{10}$ :16 = 20 | Rest | | | 7 |
| $20_{10}$ :16 = 1 | Rest | | 4 | |
| $1_{10}$ :16 = 0 | Rest | 1 | | |

# BASISUMWANDLUNG – VARIANTE 2

1. Zahl durch Zielbasis dividieren
2. Rest ist niederwertigste Stelle des Ergebnisses
3. Quotienten durch Zielbasis dividieren
4. Rest ist nächste Stelle des Ergebnisses
5. Solange bei 3. fortsetzen, bis Quotient 0

➔ Besonders gut geeignet, wenn Ausgangsbasis die „natürliche" Basis ist

- Beispiele:
  - Von binär nach dezimal:
    $10011_2 \Rightarrow 19_{10}$

$$
\begin{array}{llllll}
 & & & & 10^1 & 10^0 \\
10011_2 & : \overbrace{1010_2}^{10_{10}} & = 1 & \text{Rest} & & 1001_2 \\
1_2 & : 1010_2 & = 0 & \text{Rest} & 1 &
\end{array}
$$

# FESTKOMMAZAHLEN

- Nachkommastellen lassen sich auch in beliebigen Basen realisieren
- 1. Stelle hinter dem Komma hat den Wert $B^{-1}=1/B$
- n. Stelle hinter dem Komma hat den Wert $B^{-n}=1/B^n$

- Beispiele:
  - $10011{,}101_2 = 19 + 1*0{,}5 + 0*0{,}25 + 1*0{,}125$
    $= 19{,}625_{10}$
  - $0{,}3_{10} = 0{,}010011_2$

| | $2^{-1}$ | $2^{-2}$ | $2^{-3}$ | $2^{-4}$ | $2^{-5}$ | $2^{-6}$ |
|---|---|---|---|---|---|---|
| 0,3 * 2 = 0,6 | 0 | | | | | |
| 0,6 * 2 = 1,2 | | 1 | | | | |
| 0,2 * 2 = 0,4 | | | 0 | | | |
| 0,4 * 2 = 0,8 | | | | 0 | | |
| 0,8 * 2 = 1,6 | | | | | 1 | |
| 0,6 * 2 = 1,2 | | | | | | 1 |

# ZAHLENSYSTEME – DEFINITION

Ein **Stellenwertsystem (Zahlensystem)** ist ein Tripel

$S = (b, Z, \delta)$ mit den folgenden Eigenschaften:

- $b \geq 2$ ist eine natürliche Zahl, die **Basis** des Stellenwertsystems.
- $Z$ ist eine $b$-elementige Menge von Symbolen, den Ziffern.
- $\delta : Z \rightarrow \{0, 1, ..., b\text{-}1\}$ ist eine Abbildung, die jeder Ziffer umkehrbar eindeutig eine natürliche Zahl zwischen 0 und $b$-1 zuordnet.

# FESTKOMMAZAHL – DEFINITION

- Eine Festkommazahl ist
  - eine endliche Folge von Ziffern aus einem Zahlensystem zur Basis b mit Ziffernmenge Z.
  - Sie besteht aus n+1 Vorkommastellen ($n \geq 0$) und $k \geq 0$ Nachkommastellen.
  - Der Wert $<d>$ einer nicht-negativen Festkommazahl
  - $d = d_n \, d_{n-1} \, .... \, d_1 \;\; d_0 \; d_{-1} \, .... \, d_{-k}$
  - mit $d_i \in Z$ ist gegeben durch

$$\boxed{< d >= \sum_{i=-k}^{n} b^i \cdot \delta(d_i)}$$

# BINÄRES ZAHLENSYSTEM

- Verwendung des binären Zahlensystems hat deutliche Vorteile
  - Die einzelnen Stellen können durch einfache physikalische Signale dargestellt werden (Strom oder kein Strom, Licht oder kein Licht)
  - Die Fehleranfälligkeit ist minimal
    (vgl. Dezimalsystem: Schon bei einem Fehler von 5% kann es zu einer Fehlinterpretation kommen)
  - Die Schaltungen zur Verarbeitung der einzelnen Stellenwerte sind extrem einfach (wird später erläutert)
  - Eine Stelle wird ein "Bit" genannt

# NEGATIVE ZAHLEN

- Bisher: Nur positive Zahlen
- Drei gängige Verfahren zur Darstellung negativer Zahlen:
  - Vorzeichen und Betrag
  - Offset-Darstellung
  - Komplement-Darstellung
- Voraussetzung für alle Verfahren:
  - Festlegung der Anzahl der Binärstellen (Tetrade=4Bit, Byte=8Bit, Halbwort=16Bit, …)
- Die Stellenzahl sei im Folgenden 8Bit

# NEGATIVE FESTKOMMAZAHLEN

Bei der Darstellung negativer Zahlen gibt es folgende Alternativen:

- **Vorzeichen/Betrag-Darstellung**

  $[d_n,d_{n-1},\ldots,d_0,d_{-1},\ldots,d_{-k}]_{\mathrm{BV}} := (-1)^{d_n} \Sigma_{i=-k,\ldots,n-1}\, d_i\, 2^i$

- **Offset-Darstellung**

  $[d_n,d_{n-1},\ldots,d_0]_{\mathrm{Off}} := \Sigma_{i=0,\ldots,n}\, d_i\, 2^i - 128$

- **Komplement-Darstellung**

  $[d_n,d_{n-1},\ldots,d_0,d_{-1},\ldots,d_{-k}]_1 := \Sigma_{i=-k,\ldots,n-1}\, d_i\, 2^i - d_n\,(2^n-2^{-k})$

- **2er-Komplement-Darstellung**

  $[d_n,d_{n-1},\ldots,d_0,d_{-1},\ldots,d_{-k}]_2 := \Sigma_{i=-k,\ldots,n-1}\, d_i\, 2^i - d_n\, 2^n$

# VORZEICHEN/BETRAG-DARSTELLUNG

- Höchstwertiges Bit wird als Vorzeichen interpretiert (0=positiv, 1=negativ)

- Beispiele:
  - $37_{10}$ = $0010\ 0101_2$
  - $-20_{10}$ = $1001\ 0100_2$

- Wertebereich: -127 ... 127
- Problem:
  - 0 kommt doppelt vor (positiv und negativ)

# OFFSET-DARSTELLUNG

- Grundidee: Null-Punkt wird verschoben (Offset)
  - z.B.: Festlegung des Offset auf 128
    (eine 0 wird durch 128 kodiert)

- Beispiele:
  - $37_{10}$ = 37 + 128 = 165 = $1010\ 0101_2$
  - $-20_{10}$ = -20 + 128 = 108 = $0110\ 1100_2$

- Offset praktisch immer 2er-Potenz
  (meist Mitte des Intervalls)
- Wertebereich -128 ... 127 (bei gegebenem Offset)

# KOMPLEMENT-DARSTELLUNG

- Bei negativen Zahlen werden alle Stellen negiert (Negation: Umwandlung ins Gegenteil)
- Beispiele:
  - $37_{10} = 0010\ 0101_2$
  - $-20_{10} = \mathit{NEG}(0001\ 0100_2) = 1110\ 1011_2$
- Wertebereich: -127 ... 127
- Problem:
  - 0 kommt doppelt vor

# 2ER-KOMPLEMENT-DARSTELLUNG

- Negative Zahlen werden folgendermaßen gebildet:
  1. Stellenweise negieren
  2. 1 aufaddieren

- Beispiele:
  - $37_{10}$ = $0010\ 0101_2$
  - $-20_{10}$ = NEG($0001\ 0100_2$) +1
    = $1110\ 1011_2$ +1 = $1110\ 1100_2$

- Wertebereich: -128 ... 127

# B-KOMPLEMENT-DARSTELLUNG

- 2er-Komplement kann auch auf andere Basen übertragen werden
  - Für jede Stelle die Differenz zu B-1 bilden
  - 1 aufaddieren

- Beispiele:
  - $-3140_O = 4637_O + 1 = 4640_O$
  - $-2781_{10} = 7218_{10} + 1 = 7219_{10}$

- Analog auch auf einfaches Komplement übertragbar

# NEGATIVE FESTKOMMAZAHLEN

Bei der Darstellung negativer Zahlen gibt es folgende Alternativen:

- **Vorzeichen/Betrag-Darstellung**

  $[d_n,$

- **Offse**

  $[d_n,$

**Welche Darstellung ist am besten?**

- **Komplement-Darstellung**

  $[d_n,d_{n-1},\ldots,d_0,d_{-1},\ldots,d_{-k}]_1 := \Sigma_{i=-k,\ldots,n-1}\, d_i\, 2^i - d_n\,(2^n-2^{-k})$

- **2er-Komplement-Darstellung**

  $[d_n,d_{n-1},\ldots,d_0,d_{-1},\ldots,d_{-k}]_2 := \Sigma_{i=-k,\ldots,n-1}\, d_i\, 2^i - d_n\, 2^n$

JKU

# VORZEICHEN/BETRAG-DARSTELLUNG

- Addition (Fallunterscheidung erforderlich):
  - Beide Vorzeichen gleich
    → Addition der Beträge, Vorzeichen bleibt
  - Vorzeichen unterschiedlich
    → Subtraktion des kleineren Betrags vom größeren Betrag
    → Vorzeichen entspricht dem Vorzeichen des größeren Betrags

- Beispiel:
  - -10 + -31= -(10+31) = -41
  - -20 + 15 = *SIGN*(-20) (20-15) = -5

# VORZEICHEN/BETRAG-DARSTELLUNG

- Subtraktion
  - Rückführung auf Addition
  - Invertieren des Vorzeichen vom Subtrahenden
- Beispiel:
  - 20-30 = 20+(-30) = SIGN(-30)(30-20)=-10
- Multiplikation/Division
  - Getrennte Betrachtung von Vorzeichen und Betrag
- Nachteile:
  - Viele Fallunterscheidungen (schon bei der Addition)
  - Addition und Subtraktion müssen vorhanden sein
- Vorteil:
  - Einfache Erzeugung negativer Zahlen

# OFFSET-DARSTELLUNG

- Addition:
  - Zahlenwerte werden aufsummiert
  - Aber: Offset jetzt zweimal enthalten
    → Offset muss einmal abgezogen werden
- Beispiel:
  - $20_{10} + 30_{10} = (20 + 128)_{Offset} + (30 + 128)_{Offset}$
    $= 148_{Offset} + 158_{Offset} = (306 - 128)_{Offset} = 178_{Offset}$

- Addition, negative Zahlen:
  - Keine Sonderbehandlung für negative Zahlen
  - Addition genau wie bisher (Summieren, dann Offset abziehen)
- Beispiel:
  - $20_{10} + -30_{10} = 148_{Offset} + 98_{Offset} = 118_{Offset}$

# OFFSET-DARSTELLUNG

- Subtraktion:
  - Subtrahend wird abgezogen
  - Aber: Offset jetzt keinmal enthalten
    → Offset muss einmal aufaddiert werden
- Beispiel:
  - $20_{10}$ - $30_{10}$= $148_{Offset}$ - $158_{Offset}$= -10+128 = $118_{Offset}$ = -10
- Vorteil:
  - Keine Fallunterscheidungen
- Nachteil:
  - Komplizierte Bildung negativer Zahlen
  - Subtraktion und Addition müssen vorhanden sein

# 2ER-KOMPLEMENT-DARSTELLUNG

- Addition:
  - Keine Fallunterscheidung bei negativen Zahlen
  - Keine Korrektur des Ergebnisses

- Beispiele:
  - 10 + 20 = 30

$$
\begin{array}{rcccccccc}
 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
+ & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
\hline
 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0
\end{array} \quad =30
$$

  - -17 + 11 = -6

$$
\begin{array}{rcccccccc}
 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
+ & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
 & & & & 1 & 1 & 1 & 1 & \\
\hline
 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0
\end{array} \quad =-(5+1)=-6
$$

# 2ER-KOMPLEMENT-DARSTELLUNG

- Subtraktion:
  - Rückführung auf Addition
  - a-b = a+NEG(b)+1

- Beispiel:
  - $10_{10} - 20_{10} = -10_{10}$

$$
\begin{array}{cccccccccc}
 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & \\
+ & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & \\
+ & & & & & & & & 1 & \\
 & & & & 1 & & 1 & 1 & & \\
\hline
 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & = -(9+1) = -10
\end{array}
$$

# BEREICHSÜBERSCHREITUNGEN

- Addition bei Vorzeichen/Betrag:
  - Gleiche Vorzeichen: Überschreitung, wenn Übertrag in der letzten Stelle
  - Verschiedene Vorzeichen: Keine Überschreitung möglich

- Addition bei Offset:
  - Wenn durch Korrektur Unterlauf entsteht
  - Wenn nach Korrektur Wertebereich immer noch überschritten ist

# BEREICHSÜBERSCHREITUNGEN

- Addition bei Zweierkomplement:
  - Gleiche Vorzeichen: Überschreitung, wenn Übertrag in der letzten Stelle
  - Verschiedene Vorzeichen: Keine Überschreitung möglich
  - $64_{10} + 64_{10} = 128_{10}$

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

  - $-10_{10} + 14_{10} = 4_{10}$

$$
\begin{array}{rccccccccc}
 & & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
+ & & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
 & & 1 & 1 & 1 & 1 & 1 & 1 & & \\
\hline
 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0
\end{array}
$$

JKU

# BEREICHSÜBERSCHREITUNGEN

- Addition bei Zweierkomplement:
  - ~~Gleiche Vorzeichen: Überschreitung, wenn Übertrag in der letzten Stelle~~
  - Verschiedene Vorzeichen: Keine Überschreitung möglich
  - Gleiche Vorzeichen: Überschreitung, wenn Vorzeichenbit des Ergebnis anders

# MULTIPLIKATION

- Multiplikation im Binärsystem trivial
  - Wie schriftliche Multiplikation
  - Multiplikand wird stellengerecht aufsummiert, wenn Multiplikator an der entsprechenden Stelle eine 1 hat

```
1 0 1 1 * 0 1 0 1     11*5
-----------------
0 0 0 0
  1 0 1 1
    0 0 0 0
      1 0 1 1
    1
-----------------
0 1 1 0 1 1 1          55
```

# DIVISION

- Binärdivision wie schriftliche Division
- Beispiel
  - 5/3 $= 1{,}6_{10}$
    $= 1+0.5+0{,}125+\ldots$
    $= 1{,}101_2$

```
  1 0 1 / 1 1 = 0 1 , 1 0 1
- 1 1
  -----
  1 0 1
-   1 1
  -------
  0 1 0 0
-     1 1
  ---------
    0 0 1 0
-       1 1
        -------
        1 0 0
-         1 1
        -------
        0 0 1
```

Gleicher Rest – Binärsequenz wiederholt sich

# PROBLEME BEI FESTKOMMAZAHLEN

- Darstellung mit *n* Vorkommastellen und *k*
  - keine ganz großen bzw. kleinen Zahlen darstellbar!
  - Zahlen mit größtem Absolutbetrag: $-2^n$ und $2^n-2^{-k}$
  - Zahlen mit kleinstem Absolutbetrag: $-2^{-k}$ und $2^{-k}$
  - Operationen sind nicht abgeschlossen
    $2^{n-1}+2^{n-1}$ ist nicht darstellbar, obwohl die Operanden darstellbar sind
  - Assoziativgesetz und Distributivgesetz gelten nicht, da bei Anwendung der Gesetze evtl. der darstellbare Zahlenbereich verlassen wird!
    Bsp.: $(2^{n-1}+2^{n-1})-2^{n-1} \rightarrow\leftarrow 2^{n-1}+(2^{n-1}-2^{n-1})$