

- ☐ Merkblatt zur Eintragung in die Tutorien
- ☐ Tutorien fangen am Montag an
- ☐ Skript ist ausverkauft. Aber ab nächsten Freitag wieder verfügbar

2.1 Zahlensysteme

- ☐ Für Datenverarbeitung im Rechner sind die Zahlensysteme zur Basis 2 (binär), 8 (oktal) und 16 (hexadezimal) relevant
 - Einzelne Binärziffern (0/1) werden als Bit bezeichnet (Kleinste Informationseinheit)
- ☐ Probleme bei der Darstellung von Zahlen im Rechner:
 - Endlicher Speicher → nur Zahlen mit endlicher Genauigkeit darstellbar
 - Überläufe
 - Negative und positive Zahlen müssen unterschieden werden
 - Gleitkommazahlen müssen auch darstellbar sein

Beispiel: Euklidischer Algorithmus

Umwandlung von 15741,233₁₀ ins Hexadezimalsystem

- $16^3 \leq 15741,233 < 16^4$ → höchste Potenz 16^3
- $15741,233 : 16^3 = 3$ Rest 3453,233
- $3453,233 : 16^2 = D$ Rest 125,233
- $125,233 : 16 = 7$ Rest 13,233
- $13,233 : 1 = D$ Rest 0,233
- $0,233 : 16^{-1} = 3$ Rest 0,0455
- $0,0455 : 16^{-2} = B$ Rest 0,00253
- $0,00253 : 16^{-3} = A$ Rest 0,000088593
- $0,000088593 : 16^{-4} = 5$ Rest 0,000012299 (→ Fehler)

→ $15741,233_{10} \approx 3D7D,3BA5_{16}$

Umwandlung des Nachkommanteils

Auch der gebrochene Anteil $\sum_{i=-m}^0 z_i b^i$ einer Zahl lässt sich entsprechend schreiben:

$$Y_b = ((\dots((y_{-m} b^{-1} + y_{-(m+1)}) b^{-1} + y_{-(m+2)}) b^{-1} + \dots + y_{-2}) b^{-1} + y_{-1}) b^{-1}$$

Verfahren:

Sukzessive Multiplikation des Nachkommanteils der Dezimalzahl mit der Basis b des Zielsystems → die y_i in der Reihenfolge der höchstwertigen zur niedrigstwertigen Nachkommaziffer

Zahlendarstellung und Kodierung

- Zahlensysteme
- Darstellung negativer Zahlen
- Darstellung Fest- und Fließkommazahlen
- Kodierungen zur Zahlen und Zeichendarstellung

Zahlen-Umwandlung (1)

Umwandlung vom Dezimalsystem in ein Zahlensystem zur Basis b

1. Methode: Euklidischer Algorithmus:

$$Z = z_n 10^n + z_{n-1} 10^{n-1} + \dots + z_1 10 + z_0 + z_{-1} 10^{-1} + \dots + z_{-m} 10^{-m}$$

$$= y_p b^p + y_{p-1} b^{p-1} + \dots + y_1 b + y_0 + y_{-1} b^{-1} + \dots + y_{-q} b^{-q}$$

Ziffern werden sukzessive, beginnend mit der höchstwertigen Ziffer, berechnet:

Zahlen-Umwandlung (2)

2. Methode: Abwandlung des Horner Schemas

Ganzzahligen und der gebrochenen Anteil getrennt betrachten

Umwandlung des ganzzahligen Anteils:

Eine ganze Zahl $X_b = \sum_{i=0}^n z_i b^i$ kann durch fortgesetztes

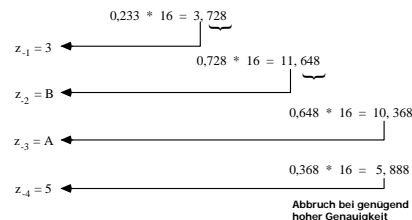
Ausklammern auch in folgender Form geschrieben werden:

$$X_b = (((\dots((z_n b + z_{n-1}) b + z_{n-2}) b + \dots + z_1) b + z_0)$$

Die gegebene Dezimalzahl wird sukzessive durch die Basis b dividiert

Beispiel: Horner Schema

Umwandlung von 0,233₁₀ ins Hexadezimalsystem:



→ $0,233_{10} \approx 0,3BA5_{16}$

- Menschen rechnen gewöhnlich im Dezimalzahlensystem
- Rechner rechnen gewöhnlich im Dualzahlensystem
- eine Konvertierung ist erforderlich
- Weitere Zahlensysteme wie Oktal-Zahlensystem oder Hexadezimal-Zahlensystem zur kompakteren Darstellung der sehr langen Dualzahlen verwendet.
- es ist notwendig, die Zusammenhänge und mathematischen Grundlagen dieser Zahlensysteme zu verstehen

Euklidischer Algorithmus

- Schritt:** Berechne p gemäß der Ungleichung $b^p \leq Z < b^{p+1}$ (setze i=p)
- Schritt:** Ermittle y_i und den Rest R_i durch Division von Z_i durch b^i
 $y_i = Z_i \text{ div } b^i; R_i = Z_i \text{ mod } b^i;$
- Schritt:** Wiederhole 2. Schritt für $i = p-1, \dots$ und ersetze dabei nach jedem Schritt Z durch R_i , bis $R_i = 0$ oder bis b^i (und damit der Umrechnungsfehler) gering genug ist.

• Interpretation von $X_b = \sum_{i=0}^n z_i b^i$ als Polynom
 $X_b = \sum_{i=0}^n z_i x^i$ mit $x=b$

• Beispiel: $X_b = z_2 x^2 + z_1 x + z_0 = ((z_2 x + z_1) x + z_0)$ mit $x=b$

• Sukzessive Division:
 $((z_2 b + z_1) b + z_0) : b = (z_2 b + z_1) b + z_0$ Rest z_0
 $(z_2 b + z_1) b + z_1 : b = z_2 b + z_1$ Rest z_1
 $z_2 b + z_2 : b = z_2$ Rest z_2
 $z_2 : b = 0$ Rest z_2

Umwandlung: Basis b → Dezimalsystem

Werte der einzelnen Stellen nach nach der Stellenwertgleichung aufsummiert

Wert X_b der Zahl ergibt sich dann als Summe der Werte aller Einzelstellen $z_i b^i$:

$$X_b = z_n b^n + z_{n-1} b^{n-1} + \dots + z_1 b + z_0 + z_{-1} b^{-1} + \dots + z_{-m} b^{-m} = \sum_{i=-m}^n z_i b^i$$

Wert X_b der Zahl als Summe der Werte aller Einzelstellen $z_i b^i$: Basis b + Anzahl der Symbole / Ziffern

$$X_b = z_n b^n + z_{n-1} b^{n-1} + \dots + z_1 b + z_0 + z_{-1} b^{-1} + \dots + z_{-m} b^{-m} = \sum_{i=-m}^n z_i b^i$$

Beispiel: $10,012 = 1 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = 2,25_{10}$

Interessante Zahlensysteme in der Informatik:

b	Zahlensystem	Zahlenbezeichnung
2	Dualsystem	Dualzahl
8	Oktalsystem	Oktalzahl
10	Dezimalsystem	Dezimalzahl
16	Hexadezimalsystem <i>Sechszehnersystem</i>	Hexadezimalzahl

$Z = D \cdot Q + R$ $Z \dots$ Dividenden
 $0 \leq R < |D|$ $D \dots$ Divisor
 $Z \text{ div } D = Q$ $Q \dots$ Quotient
 $Z \text{ mod } D = R$ $R \dots$ Rest

Beispiel: $10 \text{ div } 3 = 3$
 $10 \text{ mod } 3 = 1$

Horner Schema

Die jeweiligen ganzzahligen Reste ergeben die Ziffern der Zahl X_b in der Reihenfolge von der niedrigstwertigen zur höchstwertigen Stelle

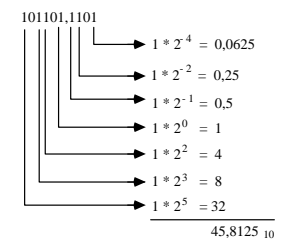
Beispiel: 15741₁₀ ins Hexadezimalsystem

- $15741_{10} : 16 = 983$ Rest 13 (D_{16})
- $983_{10} : 16 = 61$ Rest 7 (7_{16})
- $61_{10} : 16 = 3$ Rest 13 (D_{16})
- $3_{10} : 16 = 0$ Rest 3 (3_{16})

→ $15741_{10} = 3D7D_{16}$

Beispiel: Basis b → Dezimalsystem

Konvertiere 101101,1101₂ ins Dezimalsystem



- Zahl ins Dezimalsystem umwandeln
- Wandlung ins Zielsystem mit Methode 1 oder 2

Spezialfall:

Ist eine Basis eine Potenz der anderen Basis, können einfach mehrere Stellen zu einer Ziffer zusammengefasst werden oder eine Stelle kann durch eine Folge von Ziffern ersetzt werden.

Wandlung von $0110100,110101_2$ ins Hexadezimalsystem

$2^4 = 16 \Rightarrow 4 \text{ Dualstellen} \rightarrow 1 \text{ Hexadezimalstelle}$



Betrag durch 3 Bit:

0000 = 0	1000 = -0
0001 = 1	1001 = -1
0010 = 2	1010 = -2
0011 = 3	1011 = -3
0100 = 4	1100 = -4
0101 = 5	1101 = -5
0110 = 6	1110 = -6
0111 = 7	1111 = -7

Symmetrischer Zahlenbereich

Nachteile:

- Darstellung ändert sich bei Bereichserweiterung
- Bei Addition und Subtraktion müssen die Vorzeichen der Operanden gesondert betrachtet werden \rightarrow Bedingt geeignet zur Implementierung auf dem Rechner
- **Redundanz:** Es gibt zwei Repräsentationen der Null (+0 und -0)

Verbesserung: Einerkomplement-Darstellung

- Symmetrischer Zahlenbereich
- **Vorteil gegenüber der Darstellung mit Vorzeichenbit:** Erste Stelle bei Addition und Subtraktion muss **nicht** gesondert betrachtet werde.
- **Aber:** Es gibt weiterhin zwei Darstellungen der Null
- Beseitigung dieses Nachteils: Zweierkomplement

- Alle anderen negativen Zahlen werden um 1 verschoben, das MSB (Most Significant Bit) bleibt aber gleich 1.
- Aus der ersten Stelle kann das Vorzeichen der Zahl abgelesen werden
- Bitfolge $z_n z_{n-1} \dots z_0$ hat den Dezimalwert:

$$Z = -z_n \cdot 2^n + z_{n-1} \cdot 2^{n-1} + \dots + z_0$$

1101	=	$-1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$	=	-3
1111 11101	=	...	=	-3

Vier verschiedene Formate für die Darstellung negativer Zahlen im Rechner:

- Darstellung mit Betrag und Vorzeichen
- Stellenkomplement-Darstellung (Einerkomplement-Darstellung)
- Zweierkomplement-Darstellung
- Offset-Dual-Darstellung / Exzess-Darstellung

(Auch Stellenkomplement-Darstellung genannt) Stellenkomplement der entsprechenden positiven Zahl.

Um das **Einerkomplement zu bilden, wird jedes Bit der entsprechenden positiven Zahl negiert.**

0000 = 0	1000 = -7
0001 = 1	1001 = -6
0010 = 2	1010 = -5
0011 = 3	1011 = -4
0100 = 4	1100 = -3
0101 = 5	1101 = -2
0110 = 6	1110 = -1
0111 = 7	1111 = -0

Man addiert nach der Stellenkomplementierung noch eine 1

Man erhält so das Zweierkomplement: $z_{zk} = 2^{n+1} - z$



Beispiel (n = 31)

0000 0000 0000 0000 0000 0000 0010	=	2
0000 0000 0000 0000 0000 0000 0001	=	1
0000 0000 0000 0000 0000 0000 0000	=	0
1111 1111 1111 1111 1111 1111 1111	=	-1
1111 1111 1111 1111 1111 1111 1110	=	-2
1000 0000 0000 0000 0000 0000 0000	=	-2^{31}
	=	-2147483648
0111 1111 1111 1111 1111 1111 1111	=	$2^{31} - 1$
	=	2147483 647

Eine Stelle wird als Vorzeichenbit benutzt. Das ist das am weitesten links stehende Bit (MSB, most significant bit):

- MSB = 0 \rightarrow positive Zahl
- MSB = 1 \rightarrow negative Zahl

Beispiel:

0001 0010	=	+18
1001 0010	=	-18

□ Stellenkomplement der entsprechenden positiven Zahl entspricht dem Einerkomplement:

$$z_{ek} = (2^{n+1} - 1) - z$$

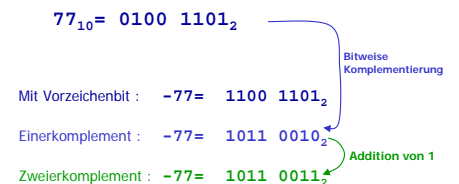
- Negative Zahlen sind wiederum durch ein gesetztes Bit in der ersten Stelle charakterisiert
- Bitfolge $z_n z_{n-1} \dots z_0$ hat den Wert:

$$Z = -z_n \cdot (2^n - 1) + z_{n-1} \cdot 2^{n-1} + \dots + z_0$$

0000 = 0	1000 = -8
0001 = 1	1001 = -7
0010 = 2	1010 = -6
0011 = 3	1011 = -5
0100 = 4	1100 = -4
0101 = 5	1101 = -3
0110 = 6	1110 = -2
0111 = 7	1111 = -1

Unsymmetrischer Zahlenbereich

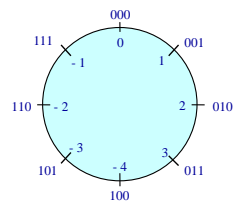
Die Zahl -77₁₀ soll mit 8 Bit dargestellt werden



Nachteil:

Unsymmetrischer Zahlenbereich. Die kleinste negative Zahl ist betragsmäßig um 1 größer als die größte positive Zahl

3-Bit-ZK-Zahlen:



Offset-Dual- (Exzess-) Darstellung

- Wird hauptsächlich bei der Exponenten-Darstellung von Gleitkommazahlen benutzt
- Die Darstellung einer Zahl erfolgt in Form ihrer **Charakteristik**
- Der gesamte Zahlenbereich wird durch Addition einer Konstanten (Exzess, Offset) so nach oben verschoben, dass die kleinste (negative) Zahl die Darstellung **0...0** erhält
- Bei n Stellen ist der Offset 2^{n-1}
- Der Zahlenbereich ist hier auch asymmetrisch

Festkommazahlen

Vereinbarung:

- Das Komma sitzt innerhalb des Maschinenwortes, das eine Dualzahl enthalten soll, an einer festen Stelle
- Meist setzt man das Komma hinter die letzte Stelle
- Andere Zahlen können durch entsprechende Maßstabsfaktoren in die gewählte Darstellungsform überführt werden
- Negative Zahlen:** meist Zweierkomplement-Darstellung

Gleitkomma-Darstellung

- Zur Darstellung von Zahlen, die betragsmäßig sehr groß oder sehr klein sind, verwendet man die **Gleitkomma-Darstellung**.
- Sie entspricht einer halblogarithmischen Form

$$X = \pm \text{Mantisse} \cdot b^{\text{Exponent}}$$
- Die Basis b ist für eine bestimmte Gleitkomma-Darstellung fest (meist 2 oder 16) und braucht damit nicht mehr explizit repräsentiert zu werden.
- Gleitkommazahlen werden meist **nicht** im Zweierkomplement, sondern in Betrag-Vorzeichen-Form dargestellt.

Normalisierung

- Legt man für die Zahl 0 ein spezielles Bitmuster fest, ist die erste Stelle der Mantisse in normalisierter Form immer gleich 1 (d.h. $0,1 \dots$)
- Die erste Stelle der Mantisse braucht im Maschinenformat gar nicht erst dargestellt zu werden:
 - **Man spart ein Bit bei der Speicherung oder gewinnt bei gleichem Speicherbedarf ein Bit an Genauigkeit.**
- Bei arithmetischen Operationen und bei der Konversion in andere Darstellungen darf diese Stelle natürlich nicht vergessen werden.

Zusammenfassung der Möglichkeiten

Darstellung mit				
Dezimalzahl	Betrag + Vorzeichen	Einer-komplement	Zweier-komplement	Charakteristik
-4	- - -	- - -	1 0 0	0 0 0
-3	1 1 1	1 0 0	1 0 1	0 0 1
-2	1 1 0	1 0 1	1 1 0	0 1 0
-1	1 0 1	1 1 0	1 1 1	0 1 1
0	1 0 0 0 0 0	1 1 1 0 0 0	0 0 0	1 0 0
1	0 0 1	0 0 1	0 0 1	1 0 1
2	0 1 0	0 1 0	0 1 0	1 1 0
3	0 1 1	0 1 1	0 1 1	1 1 1

Festkommazahlen

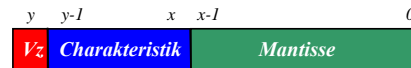
- Datentyp "integer" (Ganzzahlen) ist ein spezielles Festkommaformat.
- Manche Programmiersprachen erlauben die Definition von Ganzzahlen unterschiedlicher Länge.

Beispiel "C":

"short int", "int", "long int", "unsigned"

Gleitkomma-Maschinenformat

$$X = \pm \text{Mantisse} \cdot b^{\text{Exponent}}$$



$$X = (-1)^{Vz} * (0, \text{Mantisse}) * b^{\text{Exponent}}$$

$$\text{Exponent} = \text{Charakteristik} - b^{(y-1) - x}$$

2.3 Fest- und Gleitkommazahlen

Zahlendarstellung auf dem Papier:

Ziffern **0 .. 9**
Vorzeichen **+ -**
Komma **,**

Zahlendarstellung im Rechner:

Binärziffern **0, 1**

→ spezielle Vereinbarungen für die Darstellung von Vorzeichen und Komma im Rechner sind erforderlich.

Probleme bei Festkommazahlen

- Keine ganz großen bzw. ganz kleinen Zahlen darstellbar
Zweierkomplement mit n Vorkommastellen und k Nachkommastellen:
 - Zahlen mit größtem Absolutbetrag: -2^n und $2^n - 2^k$
 - Zahlen mit kleinstem Absolutbetrag: -2^k und 2^k
- Keine Abgeschlossenheit: $2^{n-1} + 2^{n-1}$ ist nicht darstellbar, obwohl die Operanden darstellbar sind
- Assoziativgesetz und Distributivgesetz gelten nicht, da bei der Anwendung der Gesetze evtl. der darstellbare Zahlenbereich verlassen wird

$$(2^{n-1} + 2^{n-1}) - 2^{n-1} \neq 2^{n-1} + (2^{n-1} - 2^{n-1})$$

Gleitkomma-Darstellung

- Bei der **Mantisse** ist die Lage des Kommas wieder durch Vereinbarung festgelegt (meist links vom MSB).
- Der **Exponent** ist eine ganze Zahl, die in Form ihrer Charakteristik dargestellt wird.
- Sowohl für die Charakteristik als auch für die Mantisse wird im Rechner ein **feste Anzahl von Speicherstellen** festgelegt.
- Die Länge der Charakteristik $y-x$ bestimmt die Größe des Zahlenbereichs, die Länge der Mantisse x die Genauigkeit der Darstellung.

Fest- und Gleitkommazahlen

Darstellung des Vorzeichens:

wurde im vorigen Abschnitt behandelt

Darstellung des Kommas: 2 Möglichkeiten

- Festkomma Darstellung
- Gleitkomma Darstellung

• Größte Zahl: $z = 0 \underbrace{1 \dots 1}_{n \text{ Mal}} \underbrace{, 1 \dots 1}_{k \text{ Mal}}$

$$z = 2^{n-1} + 2^{n-2} + \dots + 2^1 + 2^0 + 2^{-1} + 2^{-2} + \dots + 2^{-k+1} + 2^{-k}$$

$$2z = 2^n + 2^{n-1} + 2^{n-2} + \dots + 2^1 + 2^0 + 2^{-1} + 2^{-2} + \dots + 2^{-k+1} + 2^{-k}$$

$$z = 2z - z = 2^n - 2^{-k}$$

• kleinste positive Zahl: z

Normalisierung

Eine Gleitkommazahl heißt **normalisiert**, wenn für den Wert der Mantisse gilt:

$$\frac{1}{b} \leq 0, \text{Mantisse} < 1$$

→ **In dualer Darstellung ist die erste Stelle nach dem Komma gleich 1.**

Ausnahme:

Bei der Zahl 0 sind alle Stellen der Mantisse gleich Null