Tutoriumsblatt 6
Rechnerarchitektur im SoSe 2020

Zu den Modulen I, J

Tutorium: Die Aufgaben werden in Tutorien-Videos vorgestellt, die am 28. Mai 2020 (17 Uhr) veröffentlicht werden.

Aufgabe T18: Darstellung ganzer Zahlen

(– Pkt.)

a. Geben Sie die folgenden Dezimalzahlen als Dualzahlen in ihrer 1er-Komplement-, 2er-Komplement- und in Sign/Magnitude-Darstellung an (jeweils 10 Bit). Bei der Sign/Magnitude-Darstellung wird das hochwertigste Bit als Vorzeichen interpretiert: $(b_9 \dots b_1 b_0)_2 = (-1)^{b_9} * \sum_{i=0}^{8} b_i 2^i$

(i) $(123)_{10}$

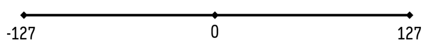
(ii) $(-123)_{10}$

b. Wandeln Sie folgende Dualzahlen in ihre Dezimaldarstellung um. Interpretieren Sie die Dualzahlen jeweils als in 1er- und 2er-Komplement-Darstellung sowie in Sign/Magnitude-Darstellung gegeben.

(i) $(1111101011)_2$

(ii) $(0001011010)_2$

Sign-magnitude Darstellung.



Beispiel mit 8 Bits:

Das höchstwertigste Bit zeigt das Vorzeichen an. Die restlichen Bits (im Beispiel 7 Bits) werden für die Darstellung der Zahl verwendet.

Nachteil: Es gibt 2 Darstellungen für die Null (+0 und -0)

1er-Komplement

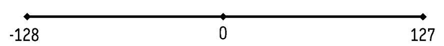
Same as above

$K_1(50) = 00110010$ ← Do nothing

$K_1(-50) = 11001101$ ← Flip bits

2er-Komplement

Only 1 Darstellung for 0 :



Zweierkomplement Operation:

Bitweise invertieren und +1 rechnen der positiven Darstellung der Zahl.

$K_2(50) \quad = 00110010$

$K_2(-50) \quad = 11001101 + 1 = 11001110$

(a)

make sure it's Vorzeichen

only makes a diff if VZ = 1

Darstellung	$(\mathbf{123})_{\mathbf{10}}$	$(\mathbf{-123})_{\mathbf{10}}$
Sign/Magnitude	$(0001111011)_2$	$(1001111011)_2$
Einerkomplement	$(0001111011)_2$	$(1110000100)_2$
Zweiterkomplement	$(0001111011)_2$	$(1110000101)_2$

123 : 2 = 61	Rest: 1
61 : 2 = 30	Rest: 1
30 : 2 = 15	Rest: 0
15 : 2 = 7	Rest: 1
7 : 2 = 3	Rest: 1
3 : 2 = 1	Rest: 1
1 : 2 = 0	Rest: 1

(b)

Darstellung	$(\mathbf{1111101011})_{\mathbf{2}}$	$(\mathbf{0001011010})_{\mathbf{2}}$
Sign/Magnitude	$(-491)_{10}$	$(90)_{10}$
Einerkomplement	$(-20)_{10}$	$(90)_{10}$
Zweiterkomplement	$(-21)_{10}$	$(90)_{10}$

1er Komplement

Vorzeichen

$(1\,1\,1\,1\,1\,0\,1\,0\,1\,1)_2$

$-\;\underbrace{(0\,0\,0\,0\,0\,1\,0\,1\,0\,0)_2}$

$(- \quad 20)_2$

2er-Komplement

Vorzeichen

$(1\,1\,1\,1\,1\,0\,1\,0\,1\,1)_2$

$-\;\underbrace{(0\,0\,0\,0\,0\,1\,0\,1\,0\,0)_2} - 1$

$(- \quad 20)_2 \quad - 1 = (-21)_{10}$

c. Geben Sie jeweils in 1er- und 2er-Komplement-Darstellung und in Sign/Magnitude-Darstellung bei Verwendung von 10 Bits an:

(i) die größte darstellbare positive Zahl,

(ii) die kleinste darstellbare positive Zahl,

(iii) die größte darstellbare negative Zahl (d.h. die negative Zahl, die den geringsten Abstand zur Null hat),

(iv) die kleinste darstellbare negative Zahl (d.h. die negative Zahl, die den größten Abstand zur Null hat),

(v) die Zahl Null.

d. Gibt es einen Unterschied zwischen „2er-Komplement" und „2er-Komplement-Darstellung"? Wenn ja, welchen?

Sign Magnitude / 1er Komplement
$[-(2^{n-1}-1), (2^{n-1}-1)]$

2er - Komplement
$[-2^{n-1}, 2^{n-1}-1]$

Darstellung	Sign/Magnitude	Einerkomplement	Zweierkomplement	Dezimal
größte, pos.	0111111111	0111111111	0111111111	511
kleinste, pos.	0000000001	0000000001	0000000001	1
größte, neg.	1000000001	1111111110	1111111111	-1
kleinste, neg.	1111111111	1000000000	1000000000	-511/-512
Null	0000000000 = 1000000000	0000000000 = 1111111111	0000000000 (eindeutig)	0

(d)

Ja, es existiert ein signifikanter Unterschied

- **2er-Komplement** bezeichnet Rechenoperation auf einem Bitmuster (nämlich: Bits invertieren und 1 addieren)
- **2er-Komplement-Darstellung** ist eine Art der Zahlendarstellung, in der bei der Darstellung negativer Zahlen das 2er-Komplement zum Einsatz kommt
- Leider wird oftmals „2er-Komplement" gesagt, wenn eigentlich „2er-Komplement-Darstellung" gemeint ist

Aufgabe T19: Addition von Dualzahlen

(– Pkt.)

In dieser Aufgabe sollen die Grundlagen der Addition in Einer- bzw. Zweierkomplement-Darstellung vertieft werden. Verwenden Sie zur binären Darstellung sämtlicher vorkommenden Zahlen jeweils 8 Bits.

a. Gegeben seien die Zahlen $(-17)_{10}$ sowie $(7)_{10}$.

   (i) Geben Sie die Einerkomplement-Darstellung der beiden Zahlen an.

   (ii) Geben Sie die Zweierkomplement-Darstellung der beiden Zahlen an.

b. Addieren Sie die Zahlen $(-17)_{10}$ und $(7)_{10}$ binär. Verwenden Sie dazu

   (i) die Einerkomplement-Darstellung.

   (ii) die Zweierkomplement-Darstellung.

c. Addieren Sie nun die Zahlen $(-56)_{10}$ und $(-72)_{10}$ binär. Verwenden Sie dazu

   (i) die Einerkomplement-Darstellung.

   (ii) die Zweierkomplement-Darstellung.

   Beantworten Sie zusätzlich jeweils die Frage, ob ein Überlauf stattgefunden hat. Begründen Sie ihre Antwort kurz.

(a)

Einerkomplement-Darstellung		same	Zweierkomplement-Darstellung	
00010001	(17)		00010001	(17)
11101110	$K_1(-17)$		11101110 + 1	$K_1(-17)$
			11101111	$K_2(-17)$
00000111	$K_1(7)$		00000111	$K_2(7)$

(b)

Einerkomplement-Darstellung

$$\begin{array}{rl} 11101110 & (-17)_{10} \\ +\ 00000111 & (7)_{10} \\ \scriptstyle 111 & \\ \hline 11110101 & (-10)_{10} \end{array}$$

if got übertrag: ignore

Zweierkomplement-Darstellung

$$\begin{array}{rl} 11101111 & (-17)_{10} \\ +\ 00000111 & (7)_{10} \\ \scriptstyle 1111 & \\ \hline 11110110 & (-10)_{10} \end{array}$$

(c)

$(56)_{10} = (00111000)_2$
$\to (-56)_{10} = (11000111)_2$

$(72)_{10} = (01001000)_2$
$\to (-72)_{10} = (10110111)_2$

$$\begin{array}{rl} 11000111 & (-56)_{10} \\ +\ 10110111 & (-72)_{10} \\ \scriptstyle 1\quad 111 & \\ \hline 01111110 & \\ +\ 00000001 & \text{Übertrag als Überlauf in 1er Komplement} \\ \hline 01111111 & (+127)_{10} \end{array}$$

= 0 ⇒ overflow

can only ignore in 2-er komplement

Es hat ein Überlauf stattgefunden, da zwei negative Zahlen addiert werden, jedoch eine positive Zahl als Ergebnis der Addition erhalten wird.

(b)

$(56)_{10} = (00111000)_2$
$\to (-56)_{10} = (11000111)_2 + 1$
$= (11001000)_2$

$(72)_{10} = (01001000)_2$
$\to (-72)_{10} = (10110111)_2 + 1$
$= (10111000)_2$

$$\begin{array}{rl} 11001000 & (-56)_{10} \\ +\ 10111000 & (-72)_{10} \\ \scriptstyle (1)\,1111 & \\ \hline 10000000 & (-128)_{10} \end{array}$$

Übertrag weglassen

Es hat **kein** Überlauf stattgefunden, da das Ergebnis der Addition mit den zur Verfügung stehenden Bits dargestellt werden kann.

Aufgabe T20: Gleitkommazahlen

(– Pkt.)

Nach dem IEEE 754 Standard gilt:

$$(-1)^S \cdot (1 + \text{Signifikant}) \cdot 2^{(\text{Exponent}-\text{Bias})}$$

wobei der Standard

- für das Vorzeichen S ein Bit,
- für den Signifikanten (Mantisse) 23 Bit bei einfacher und 52 Bit bei doppelter Genauigkeit,
- für den Exponenten 8 Bit bei einfacher und 11 Bit bei doppelter Genauigkeit

reserviert und den Bias auf $127 = 2^{8-1} - 1$ bei einfacher bzw. auf $1023 = 2^{11-1} - 1$ bei doppelter Genauigkeit setzt.

a. Geben Sie die Darstellung folgender Zahlen als Gleitkommazahl nach IEEE 754 in einfacher (32-Bit) Genauigkeit an:

(i) $(11,25)_{10}$

(ii) $(0,2)_{10}$

b. Wandeln Sie folgende Zahl, die in Gleitkommadarstellung (IEEE 754) gegeben ist, in ihre Dezimaldarstellung um.

MSB LSB

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	0	1	1	0	1	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S	Exponent								Significand																						

2er - komplement

$2^0/2^{-1}$

links - bündig

2^{-n}

Festkommazahlen

Fix-Point Arithmetic

Sign-Magnitude Darstellung

Das Komma steht an beliebiger, aber fester Stelle

$$111{,}011 = -(1*2^1 + 1*2^0 + 0*2^{-1} + 1*2^{-2} + 1*2^{-3})$$

Probleme

- Man kann mit einer bestimmten Anzahl von Bits nur einen beschränkten Wertebereich abdecken.
- Es muss separat gekennzeichnet oder allgemeingültig für alle Darstellungen vereinbart werden, an welcher Stelle sich das Komma befindet.
- Wenn man sehr große und sehr kleine Zahlen Darstellen möchte braucht man sehr viele Bits

Gleitkommazahlen

Floating point Arithmetic

Gleitkomma-darstellung

IEEE 754 Standard

GKD + Bias

Single Precision : 127

Double Precision : 1023

```
If E = 0    Sig != 0   => NaN
If E = 255  Sig == 0   => Infinity
If E = 255  Sig != 0   => NaN
```

(a) (i)

Schritt 1: Normalisieren
$(11{,}25)_{10} = (1011{,}01)_2 = (1{,}01101)_2 * 2^3$

Schritt 2: IEEE 754

$$(1{,}01101)_2 * 2^{3+127}$$

Sign	Exponent	Signifikant
0	10000010	01101000000000000000000

(ii) Schritt 1: Normalisieren
$(0{,}2)_{10} = (0,\overline{0011})_2 = (1,\overline{1001})_2 * 2^{-3}$

(keep shifting bits forward)

```
2 · 0,2 = 0,4    --> Ziffer: 0
2 · 0,4 = 0,8    --> Ziffer: 0
2 · 0,8 = 1,6    --> Ziffer: 1
2 · 0,6 = 1,2    --> Ziffer: 1
2 · 0,2 = 0,4    --> Ziffer: 0
```

Schritt 2: IEEE 754

$$(1{,}1001)_2 * 2^{-3+127}$$

0

Sign	Exponent	Signifikant
0	01111100	10011001100110011001100

(b)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	0	1	1	0	1	0	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S	Exponent								Significand																						

Sign = 1 → Zahl ist negativ
Exponent = Biased Exponent – Bias = 134 - 127 = 7

Exponent ≠ 0

$$-1{,}101011111 \cdot 2^7 = -11010111{,}11 = (-215{,}75)_{10}$$