ERA Tutorium 1

	Termin:		Raum:	Zulip:
Leopold Cario	· Mittwoch	10:00	03.13.010	ERA Tutorium – Mi – 1000 – 1
	· Donnerstag	12:00	00.08.059	ERA Tutorium – Do – 1200 – 1

School of Computation, Information and Technology
Technische Universität München

Lehrstuhl für
Rechnerarchitektur & Parallele Systeme
Prof. Dr. Martin Schulz
Dominic Prinz
Jakob Schäffeler

Lehrstuhl für
Design Automation
Prof. Dr.-Ing. Robert Wille
Stefan Engels
Benjamin Hien

Einführung in die Rechnerarchitektur

Wintersemester 2025/2026

Übungsblatt 1: Zahlensysteme 20.10.2025 – 24.10.2025

1 Zahlensysteme

In der Informatik werden Zahlen in der Regel im sog. Stellenwertsystem dargestellt. Der Wert einer Zahl hängt dabei von der Position der Ziffern ab:

B:10: $3125 = 5\cdot10^0 + 2\cdot10^1 + 1\cdot10^2 + 3\cdot10^3$
$= 5\cdot1 + 2\cdot10 + 1\cdot100 + 3\cdot1000$

$$W = \sum_{i=0}^{n-1} a_i \cdot B^i, \tag{1}$$

B=2: $1011_0 = 0\cdot2^0 + 1\cdot2^1 + 1\cdot2^2 + 0\cdot2^3 + 1\cdot2^4$
$= 0\cdot1 + 1\cdot2 + 1\cdot4 + 0\cdot8 + 1\cdot16 = 22$

wobei B die Basis, n die Anzahl der Stellen und a_i die i-te Ziffer aus dem Ziffernbereich von 0 bis $B-1$ ist. Im Alltag ist die Basis meist 10 (Dezimalsystem). Weitere häufig verwendete Stellenwertsysteme sind das Dualsystem (Binärsystem, $B = 2$), das Oktalsystem ($B = 8$) und das Hexadezimalsystem ($B = 16$).

$0010.1110.0110_2 = 0x2E6$
2 E 6

a) Überlegen Sie sich einen Algorithmus, der beliebig lange Zahlen vom Dezimalsystem in das Binärsystem umwandelt. Testen Sie den Algorithmus mit den Zahlen: $(42)_{10}$, $(100)_{10}$ und $(1.000)_{10}$.

	Rest	
42 : 2 = 21	0	LSB
21 : 2 = 10	1	
10 : 2 = 5	0	
5 : 2 = 2	1	
2 : 2 = 1	0	
1 : 2 = 0	1	MSB

$42_{10} = 10.1010_2$

Lösungsvorschlag

a) Hierfür eignet sich beispielsweise die Divisionsmethode [1]:

42	/	2	=	21	0
21	/	2	=	10	1
10	/	2	=	5	0
5	/	2	=	2	1
2	/	2	=	1	0
1	/	2	=	0	1

$\Rightarrow (42)_{10} = (10.1010)_2$

100	/	2	=	50	0
50	/	2	=	25	0
25	/	2	=	12	1
12	/	2	=	6	0
6	/	2	=	3	0
3	/	2	=	1	1
1	/	2	=	0	1

$\Rightarrow (100)_{10} = (110.0100)_2$

1000	/	2	=	500	0
500	/	2	=	250	0
250	/	2	=	125	0
125	/	2	=	62	1
62	/	2	=	31	0
31	/	2	=	15	1
15	/	2	=	7	1
7	/	2	=	3	1
3	/	2	=	1	1
1	/	2	=	0	1

$\Rightarrow (1.000)_{10} = (11.1110.1000)_2$

b) Überlegen Sie sich einen Algorithmus, der beliebig lange Zahlen vom Binärsystem in das Dezimalsystem umwandelt. Testen Sie den Algorithmus mit den Zahlen: $(1.0101)_2$ und $(1110.0011)_2$.

$$(1.0101)_2 = 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = (21)_{10}$$

$$(1110.0011)_2 = 2^7 + 2^6 + 2^5 + 2^1 + 2^0 = (227)_{10}$$

c) Lösen Sie mit Hilfe von „Tricks“ folgende Umwandlungen:

- $(1111.1111)_2 = (?)_{10}$ $1.0000.0000 - 1 = 2^8 - 1 = 256 - 1 = 255$
- $(1.0000.0000)_2 = (?)_{10}$ $2^8 = 256$
- $(65)_{10} = (?)_2$ $64 + 1 = 2^6 + 1 = 1000000 + 1 = 100.0001$
- $(1.0000.1001.0010)_2 = (4242)_{10}$
  $(10.0001.0010.0100)_2 = (?)_{10}$ $4242 \cdot 2 = 8484$

Multiplikation mit 2^n ≙ Linksshift um n
Division durch 2^n ≙ Rechtsshift um n

d) Wandeln Sie die folgenden Zahlen vom Binär- ins Hexadezimalsystem bzw. umgekehrt um:

10 = A
11 = B
12 = C
13 = D
14 = E
15 = F

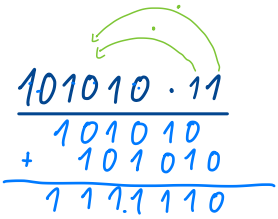
- $(\overbrace{1111}^{15}.\overbrace{1111}^{15})_2 = \texttt{0x?}$ FF
- $(\overbrace{1010}^{A}.\overbrace{1100}^{C}.\overbrace{0011}^{3})_2 = \texttt{0x?}$ AC3
- $\texttt{0x1234} = (?)_2$ 0b1.0010.0011.0100
- $\texttt{0xC0FFEE} = (?)_2$ 0b1100.0000.1111.1111.1110.1110

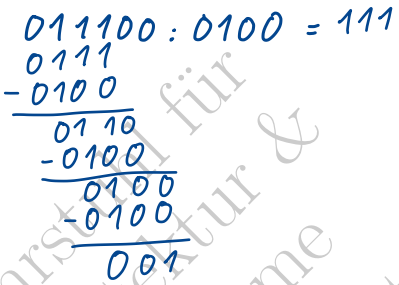
e) (Optional) Schreiben Sie auf die Vorderseite eines Stück Papiers eine Umwandlungsaufgabe (ähnlich zu 1 a-d) und ihre Lösung auf die Rückseite. Tauschen Sie diese Aufgabe mit einem Ihrer Kommilitonen. Überprüfen Sie die Lösung!

2 Arithmetik und negative Zahlen

a) Die vier Grundrechenarten Addition, Subtraktion, Multiplikation und Division verhalten sich im Binärsystem wie im Dezimalsystem: die einzelnen Ziffern werden stellenweise verarbeitet. Lösen Sie die folgenden Rechenaufgaben (alle Zahlen sind positiv):

- $(10.1010)_2 + (11.0011)_2 = (?)_2$
- $(11.0011)_2 - (10.1010)_2 = (?)_2$
- $(10.1010)_2 \cdot (11)_2 = (?)_2$
- $(01.1100)_2 : (0100)_2 = (?)_2$





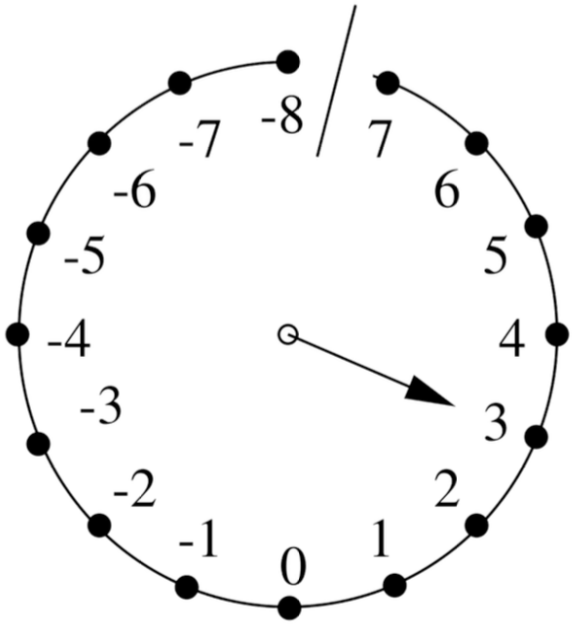
b) Bisher haben wir lediglich positive Zahlen betrachtet. Wie könnten negative Zahlen im Binärsystem dargestellt werden? Vergleichen Sie anhand der Zahl $-(42)_{10}$ die Vor- und Nachteile der Darstellungsarten. Betrachten Sie dazu 8 binäre Stellen.

Art	Beschreibung	Vor-/Nachteil
Vorzeichenbit	Ein extra Bit nur für das Vorzeichen	⊕ Analog zu Dezimalsystem ⊖ Doppelte Null ⊖ neue Arithmetik nötig
Einerkomplement	Alle Bits invertieren	⊕ intuitiv ⊖ Doppelte Null ⊖ neue Arithmetik nötig
Zweierkomplement	Einerkomplement +1	⊕ volle Ausnützung des Zahlenraums ⊕ Arithmetik wieder verwendbar ⊖ Zahl nicht direkt ablesbar

Variante: Das Zweierkomplement

Beispiel mit 4 Bits:

$$
\begin{aligned}
3_{10} &= 0011_2 \\
2_{10} &= 0010_2 \\
1_{10} &= 0001_2 \\
0_{10} &= 0000_2 \\
-1_{10} &= \overset{0\,0\,0\,1}{1110_2} + 1_2 && = 1111_2 \\
-2_{10} &= \overset{0\,0\,1\,0}{1101_2} + 1_2 && = 1110_2 \\
&\dots \\
-7_{10} &= \overset{0\,1\,1\,1}{1000_2} + 1_2 && = 1001_2 \\
-8_{10} &= \overset{1\,0\,0\,0}{0111_2} + 1_2 && = 1000_2
\end{aligned}
$$



① Bits invertieren
② +1 addieren

Bsp. 8 Bit:

$26_{10} = 0001.1010_2$

① 1110.0101_2

②
$$
\begin{array}{r}
1110.0101 \\
+\quad\quad\;\; 1 \\
\hline
1110.0110
\end{array}
$$

c) Berechnen Sie den Wert des Terms $(0011.0011)_2 - (0010.1010)_2 = (?)_2$ (bekannt aus Aufgabe 2) indem Sie den Subtrahenden negieren und anschließend auf den Minuenden aufaddieren. Verwenden Sie das Zweierkomplement.

51 42

I. 0010 1010 -> Zweierkomplement

① invertieren: 1101.0101

② 1101.0101 +1 = 1101.0110 = -42

II. Addition

$$\begin{array}{r} 0011.0011 \\ + \quad 1101.0110 \\ \hline 0000\,1001 \end{array} \quad = 9$$

d) Lösen Sie die folgenden Aufgaben, indem Sie die Zahlen zuerst ins Binärsystem mit jeweils 5 binären Stellen umwandeln und dann das Ergebnis im Binärsystem ausrechen. Benutzen Sie das Zweierkomplement. *Hinweis: Sie düfen davon ausgehen, dass die Ergebnisse wieder mit 5 binären Stellen darstellbar sind*

- $(-1) - 1$ = (-1) + (-1)
- $(-2) \cdot (-3)$
- $(-8) : 2$

- $-1 + (-1) = 1111_2$

$-1 = 11111$

$11111 + 11111 = 111110$

- $-2 \cdot (-3) = 0011_2$

$-2 = 11110$

$-3 = 11101$

$11110 \cdot 00101$

$11110 + 00000 + 11110 = 10010110$ Zwischenergebnis 1

$11110 + 11101 = 110000110$ Zw. 2

$11000 + 11110 = 1\,10011\,0110$

- $-8 : 2 = 11100$

$-8 = -(01000) = 11000 + 1 = 11001$

$-8 : 2 = 11000 : 00010 = 01100 \neq -4$

$11000 : 00010 = 01100$

-10

010

-10

000

-00

00

→ Division funktioniert nicht im Zweierkomplement

Division mit Beträgen, danach Vorzeichen ergänzen!

$8 : 2 = 01000 : 00010 = 00100$

$010 - 10 = 000 - 00 = 00 - 00 = 00$

$-(00100) = 11100 = -4$

Typname in C	Bits	Vorzeichen	min	max
char	8	ja	-128	127
unsigned char		nein	0	255
short	16	ja	-32.768	32.767
unsigned short		nein	0	65.535
int	32	ja	-2.147.483.648	2.147.483.647
unsigned int		nein	0	4.294.967.295
long long	64	ja	-9.223.372.036.854.775.808	9.223.372.036.854.775.807
unsigned long long		nein	0	18.446.744.073.709.551.615

3 Zahlenbereiche

Welcher Zahlenbereich kann mit den folgenden Binärformaten dargestellt werden?

- 8-Bit vorzeichenlos (unsigned char)

  2^8 Zahlen $\hat{=}$ 0 bis $2^8 - 1$

- 8-Bit vorzeichenbehaftet im Zweierkomplement (char)

  2^8 Zahlen $\hat{=}$ -2^7 bis $2^7 - 1$

  -128 bis 127

- 16-Bit vorzeichenlos (unsigned short)

  2^{16} Zahlen $\hat{=}$ 0 bis $2^{16} - 1$

- 32-Bit vorzeichenbehaftet im Zweierkomplement (int)

  2^{32} Zahlen $\hat{=}$ -2^{31} bis $2^{31} - 1$

4 RISC-V Simulator Einrichtung

In ERA wird zur Simulation eines RISC-V Prozessors `QtRvSim` – ein Projekt der Tschechischen Technischen Universität – verwendet. Dieser wird ab Woche 2 für die Übungen und Hausaufgaben benötigt. Für die erste Einrichtung folgen Sie bitte den folgenden Schritten:

1. Laden Sie sich die passende Installationsdatei für Ihr Betriebssystem herunter: `https://github.com/cvut/qtrvsim/releases/tag/v0.9.8`

   - Ubuntu-User können auch folgendes `PPA` verwenden: `ppa:qtrvsimteam/ppa`
   - Windows-User benutzen die Datei mit `mingw32` im Namen

   oder verwenden Sie die Web-Version (experimentell): `https://comparch.edu.cvut.cz/qtrvsim/app`

2. Belassen Sie die Einstellungen wie sie sind: „No pipeline no cache“ und klicken Sie auf „Example“.

3. In der oberen Hälfte sehen Sie die Register inkl. der zugehörigen Mnemonics und Werte.

4. Links sehen Sie die auszuführenden Instruktionen. Die Instruktionen eines Programms beginnen wie im RISC-V Ökosystem üblich bei Adresse `0x200`.

5. Sie können mithilfe der Reiter „Core“ und „template.S“ zwischen der Prozessor- und Source Code-Ansicht wechseln.

6. Klicken Sie auf „Compile Source and update memory“ (blauer Pfeil nach unten) um den aktuell ausgewählten Source Code zu kompilieren und zu laden.

7. Anschließend können Sie das Programm mit dem Play-Button starten. Als Beispiel wird der Text „Hello world.“ rechts auf dem Terminal ausgegeben.

8. Weitere Beispiele finden Sie hier: `https://gitlab.fel.cvut.cz/b35apo/stud-support/-/tree/master/seminaries/qtrvsim`

9. Tipp: Probieren Sie sich hier aus!

5 Binärarithmetik (Hausaufgabe 01)

5.1 Generelles

Es gibt in ERA lediglich sog. *public-tests*, d.h. Sie sehen direkt, ob Ihre Abgabe richtig ist. Sie haben außerdem bis zum zuvor genannten Zeitpunkt unbegrenzt viele Versuche.

5.2 Aufgabe

Bearbeitung und Abgabe auf `https://artemis.tum.de/courses/516` bis **Sonntag, den 26.10.2025, 23:59 Uhr**.