Ludwig-Maximilians-Universität München
Institut für Informatik
Lehrstuhl für Mobile und Verteilte Systeme
Prof. Dr. Claudia Linnhoff-Popien

mobile and distributed systems group

Rechnerarchitektur im Sommersemester 2019
Übungsblatt 13

Besprechung: Dieses Übungsblatt dient der Vorbereitung auf die Klausur und **wird nicht besprochen**.

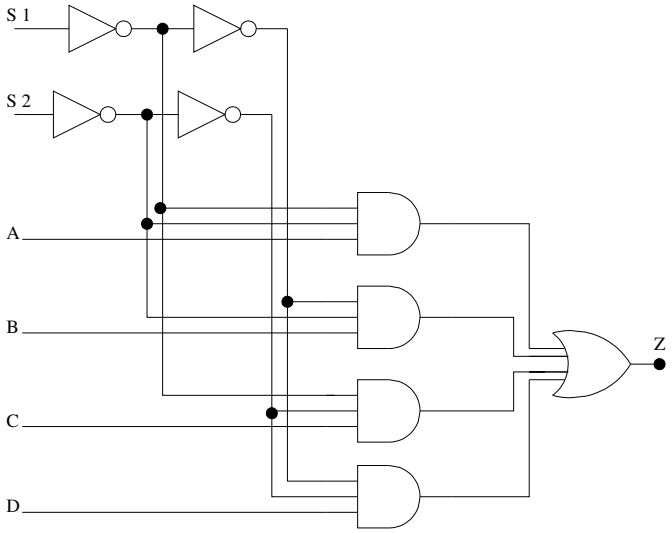
Ankündigungen:

- Am Montag, den **22. Juli 2019 findet von 14.00 – 16.00 Uhr ein Sondertutorium im Hörsaal E 004 (Geschw.-Scholl-Pl. 1)** für alle Studenten statt, an dem nochmal gezielt Fragen zum Stoff gestellt werden können. In der Woche vom 22. – 26. Juli 2019 finden keine reguläre Übungen und auch keine Vorlesung statt.
- Die **Klausur** findet am **26. Juli 2019 von 18.30 – 20.30 Uhr** statt. Bitte melden Sie sich **bis spätestens 18. Juli 2019, 23:59 Uhr** zur Klausur über UniWorX **an** bzw. **ab**.
- Wir wünschen Ihnen viel Erfolg!

Aufgabe 66: (K) Schaltnetze

(– Pkt.)

Betrachten Sie das folgende Schaltbild.



a. Beschreiben Sie das Schaltnetz mittels einer Booleschen Funktion für Z!

b. Ordnen Sie das Schaltnetz einem Ihnen bekannten Schaltungsbaustein höherer Ordnung zu (Name dieses Bausteins). Wozu werden diese Bausteine ganz allgemein benötigt?

Aufgabe 67: (K) Programmierbare logische Arrays

(– Pkt.)

Gegeben sei die Funktion:

$$f(a, b, c) = \bar{b}\bar{c} + a\bar{b}c + \bar{a}c$$

Realisieren Sie diese Funktion durch ein normiertes PLA. Verwenden Sie ausschließlich Bausteine der Typen 0 bis 3. Kennzeichnen Sie zudem in Ihrer Skizze die Und- und die Oder-Ebene. Markieren Sie gesperrte und neutralisierte Eingänge sowohl durch anlegen des entsprechenden Werts, als auch durch die explizite Beschriftung mit den Worten „gesperrt“ bzw. „neutralisiert“. Beschriften Sie in das PLA eingehende Pfeile mit der jeweils anliegenden logischen Funktion. Beschriften Sie anschließend alle aus dem PLA ausgehenden Pfeile mit der jeweils anliegenden logischen Funktion.

Aufgabe 68: (K) Optimierung von Schaltnetzen

(– Pkt.)

a. Gegeben sei die Wahrheitstabelle einer partiellen Booleschen Funktion $g(x_1, x_2, x_3, x_4)$. Undefinierte Ausgaben sind mit einem D gekennzeichnet:

	x_1	x_2	x_3	x_4	$g(x_1, x_2, x_3, x_4)$
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	D
5	0	1	0	1	D
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	0
12	1	1	0	0	D
13	1	1	0	1	1
14	1	1	1	0	D
15	1	1	1	1	0

Minimieren Sie die Funktion g unter Verwendung eines Karnaugh-Diagramms. Beachten Sie dabei die **Don’t Care** Argumente und fassen Sie dabei möglichst viele Felder zusammen. Geben Sie abschließend die minimierte Funktion in disjunktiver Form an.

Aufgabe 69: (K) Zahlendarstellung

(– Pkt.)

Beantworten Sie die folgenden Fragen im Bezug auf die Dualdarstellung von Ganzzahlen und Gleitkommazahlen:

a. Geben Sie die größte und die kleinste Dezimalzahl samt ihrer Binärdarstellung an, die jeweils unter Verwendung von 11 Bit in der Zweierkomplementdarstellung darstellbar sind.

b. Geben Sie die Zweierkomplementdarstellung der beiden Dezimalzahlen -59 und 128 unter Verwendung von 9 Bit an. Berechnen Sie danach die Summe (-59 + 128). Hat bei Ihrer Addition ein Überlauf (Overflow) stattgefunden? Begründen Sie kurz Ihre Antwort.

c. Erläutern Sie kurz, warum man bei der Darstellung einer Gleitkommazahl nach dem Standard IEEE 754 die Bias-Notation verwendet?

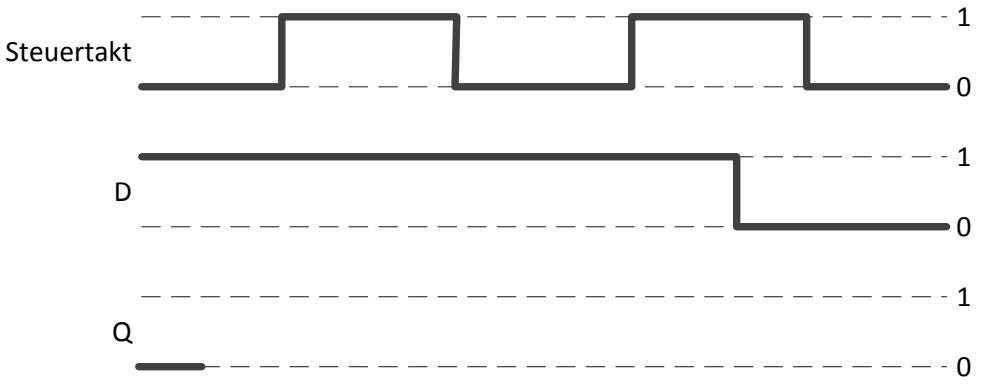
d. Wandeln Sie folgende Zahl, die in Gleitkommadarstellung (IEEE 754) gegeben ist, in ihre Dezimaldarstellung um. Sie dürfen das Ergebnis auch in Bruchdarstellung angeben.

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	0	1	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S	Exponent								Significand																						

Aufgabe 70: (K) Flip-Flop-Schaltungen

(– Pkt.)

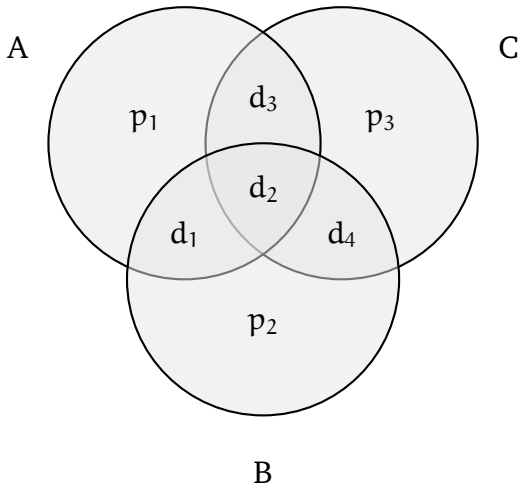
Betrachten Sie das folgende Impulsdiagramm einer D-Flip-Flop-Schaltung. Nehmen Sie an, dass der D-Flip-Flop bei steigender Flanke schaltet. Gehen sie davon aus, dass der Baustein ohne Zeitverzögerung schaltet. Vervollständigen Sie das folgende Impulsdiagramm!



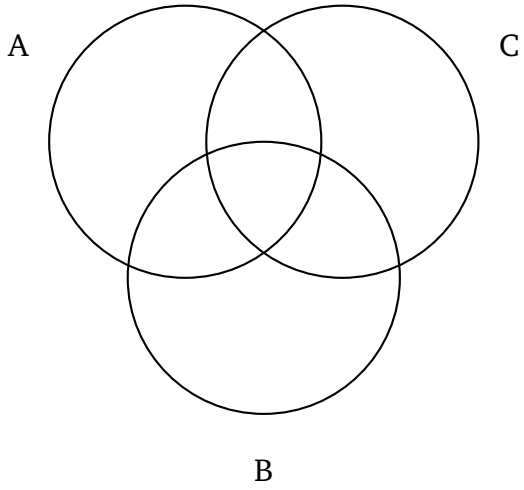
Aufgabe 71: (K) Fehlererkennungscodes

(– Pkt.)

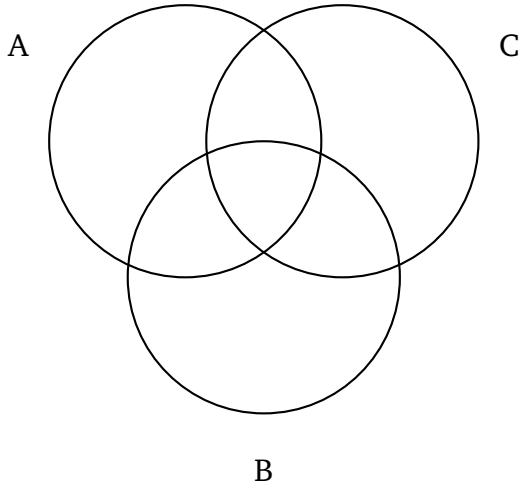
Wir gehen von folgender Struktur der Code-Wörter $d_1 d_2 d_3 d_4 p_1 p_2 p_3$ aus. Wobei $d_i (i \in \{1,2,3,4\})$ für das jeweilige Datenbit und $p_j (j \in \{1,2,3\})$ für das jeweilige Prüf- bzw. Paritätsbit steht. Die Paritätsbits zur Fehlererkennung bzw. Fehlerkorrektur für ein Datenwort $d_1 d_2 d_3 d_4$ können anschaulich mit Hilfe eines Venn-Diagramms berechnet werden, in welchem sich die Bits wie folgt anordnen:



a. Berechnen Sie unter Verwendung des folgenden Venn-Diagramms die Prüfbits für das Datenwort **`0110`**. Verwenden Sie dazu **gerade Parität**. Tragen Sie zunächst die Datenbits in die für die Berechnung sinnvollen (Schnitt-)Mengen ein.



b. Gehen Sie nun davon aus, dass Sie ein mit dem zuvor beschriebenen Code codiertes Code-Wort **`0001101`** empfangen haben. Es wurde **gerade Parität** verwendet. Handelt es sich um ein gültiges Codewort? Falls nein, treffen Sie eine Aussage darüber, an welcher/welchen Stelle/Stellen mutmaßlich (ein) Bitfehler aufgetreten ist/sind. Verwenden Sie zur Berechnung das folgenden Venn-Diagramm. Korrigieren Sie (falls nötig) den/die Fehler **innerhalb** des Venn-Diagramms und geben Sie das (ggf. korrigierte) 4-Bit Datenwort an.



Aufgabe 72: (K) Assemblerprogrammierung unter SPIM

(– Pkt.)

Beantworten Sie die folgenden Fragen zum Thema Assemblerprogrammierung des MIPS-Prozessors. **Hinweis:** Eine Übersicht zu den wichtigsten SPIM-Befehlen finden Sie am Ende des Klausurhefts.

a. Gegeben sei folgendes Programm in SPIM, in dem einige Kommentare eingefügt sind:

```
1 .data
2
```

```
3  x: .word 3, 7, 1, 2
4  y: .word 2, 1, 8, 4
5
6
7  .text
8  main:   li      $s0, 4          # Kommentar 1:
9
10         move    $t0, $zero      # Kommentar 2:
11
12         move    $t1, $zero      # Kommentar 3:
13
14
15 while:  bge     $t0, $s0, end   # Kommentar 4:
16
17         mul     $t2, $t0, 4     # Kommentar 5:
18
19         lw      $t3, x($t2)     # Kommentar 6:
20
21         lw      $t4, y($t2)     # Kommentar 7:
22
23         mul     $t5, $t3, $t4   # Kommentar 8:
24
25         add     $t1, $t1, $t5
26
27         addi    $t0, $t0, 1     # Kommentar 9:
28
29         j       while
30
31
32 end:    li      $v0, 1
33         move    $a0, $t1
34         syscall                 # Kommentar 10:
35         j       exit
36
37
38 exit:   li      $v0, 10
39         syscall                 # Programm beenden
```

Ordnen Sie in diesem Programm jeder Zeile, die mit "`# Kommentar <Nr>:`" versehen ist, den jeweils genau passenden der folgenden Kommentare zu. Tragen Sie dazu die Nummer des richtigen Kommentars aus der folgenden Liste in den obigen Coderahmen hinter der betreffenden Kommentarzeile ein!

Nr.	Kommentar	Nr.	Kommentar
(i)	`while j < 4`	(vi)	`Ergebnis initialisieren`
(ii)	`Ergebnis ausgeben`	(vii)	$temp = x_i \cdot y_i$
(iii)	Lade y_i	(viii)	`Mit Wortlänge multiplizieren`
(iv)	`j = j + 1`	(ix)	`j = 0`
(v)	Lade x_i	(x)	`Lade Länge des Arrays`

b. Beschreiben Sie kurz, was das oben angegebene Programm aus Teilaufgabe a) bewirkt!

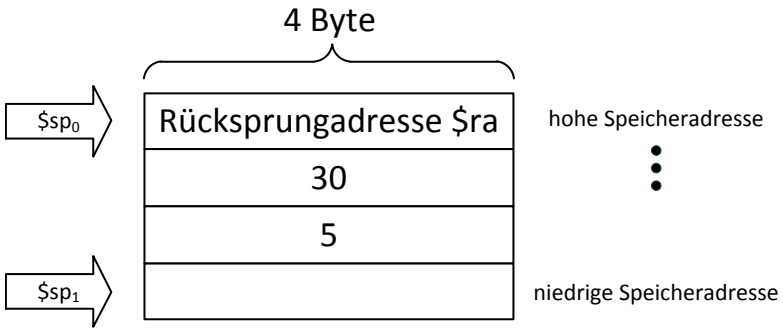
c. Unabhängig von Teilaufgabe a) sei nun die folgende Befehlssequenz eines Unterprogramms gegeben:

```
1          move    $t0, $a0
2          move    $t1, $a1
3          move    $t2, $a2
```

```
4
5           add     $v0, $t0, $t1
6           add     $v0, $v0, $t2
7           div     $v0, $v0, 3
8
9           jr      $ra
```

Um welche Art der Wertübergabe handelt es sich beim Aufruf dieses Unterprogramms?

d. Unabhängig von Teilaufgabe a sei nun folgendes Stack-Layout gegeben:

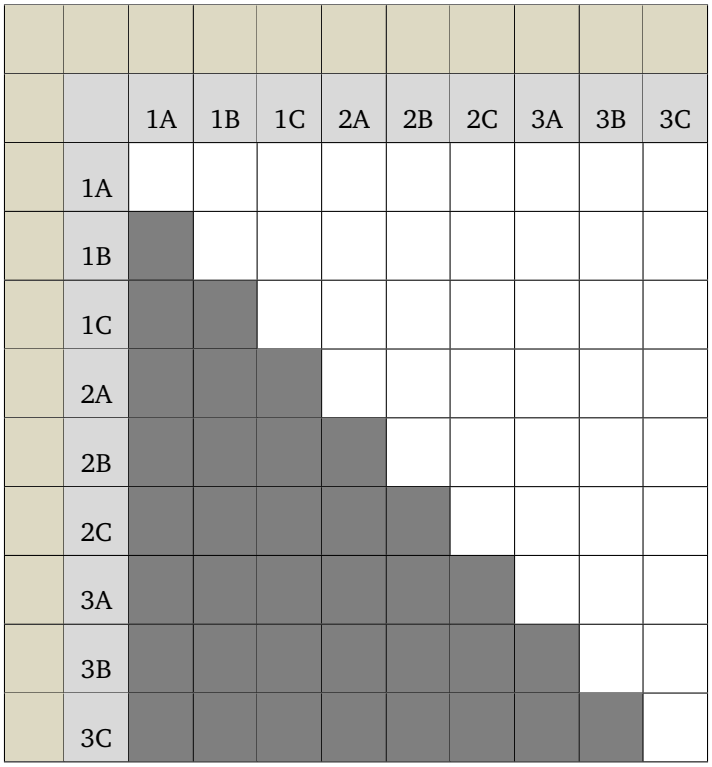


Geben Sie eine Sequenz von SPIM Befehlen an, die dieses Layout erzeugt. Beachten Sie, dass Sie Platz auf dem Stack schaffen müssen und dass dies nach der MIPS-Konvention geschehen soll. Dabei bezeichnet $\$sp_0$ die Position (bzw. den Inhalt) des Stackpointers vor der von Ihnen gegebenen Befehlssquenz und $\$sp_1$ die Position (bzw. den Inhalt) des Stackpointers nach Ihrer Befehlssequenz.

Aufgabe 73: (K) Quantenannealing

(– Pkt.)

Angenommen, es sollen drei Elektroautos $\{1, 2, 3\}$ unterschiedlicher Autohersteller aufgeladen werden. Dafür stehen drei Ladesäulen $\{A, B, C\}$ zur Verfügung. Jede der Ladesäulen wird von einem der Autohersteller, von denen auch die Autos sind, betrieben. Grundsätzlich kann jedes der Autos an einer beliebigen Säule geladen werden. Es ist jedoch besonders günstig, wenn jedes der Autos an der Säule des Herstellers geladen wird, was den Paarungen $(1, B)$, $(2, C)$ und $(3, A)$ entspricht. Unter allen theoretisch denkbaren Zuordnungen gibt es zudem „katastrophale“ Ereignisse.
Füllen Sie folgenden Matrix mit den Zahlenwerten $\{-2, 0, 5\}$, je nachdem, wie günstig eine Zustandskombination zu bewerten ist, so dass die Optimierung (Minimierung) mittels Quantenannealing stattfinden kann.



Aufgabe 74: (K) Einfachauswahlaufgabe: Wiederholung

(– Pkt.)

Für jede der folgenden Fragen ist eine korrekte Antwort auszuwählen („1 aus n“). Nennen Sie dazu in Ihrer Abgabe die jeweils ausgewählte Antwortnummer ((i), (ii), (iii) oder (iv)). Eine korrekte Antwort ergibt jeweils einen Punkt. Mehrfache Antworten oder eine falsche Antwort werden mit 0 Punkten bewertet.

<table>
<tr><td colspan="4">a) Welche Dualzahl entspricht dem hexadezimalen Wert <code>C9</code>?</td></tr>
<tr><td>(i) <code>10000001</code></td><td>(ii) <code>11001001</code></td><td>(iii) <code>10111111</code></td><td>(iv) <code>10101010</code></td></tr>
<tr><td colspan="4">b) Wie lautet eine der De Morganschen Regeln?</td></tr>
<tr><td>(i) $\overline{(a+b)} = \overline{a} \cdot \overline{b}$</td><td>(ii) $a \cdot 0 = 0$</td><td>(iii) $a + \overline{a} = 1$</td><td>(iv) $a \cdot (b+c) = (a \cdot b) + (a \cdot c)$</td></tr>
<tr><td colspan="4">c) Wie lautet die Belegung von <code>$t2</code> nach Ausführung des folgenden SPIM-Codes?
<pre>.data
var: .word 8, 32, 17, 4, 9

.text
main: lw $t1, var
      lw $t2, var+4($t1)</pre></td></tr>
<tr><td>(i) <code>8</code></td><td>(ii) <code>32</code></td><td>(iii) <code>12</code></td><td>(iv) <code>4</code></td></tr>
</table>

d) Sei folgende Wahrheitstafel einer Booleschen Funktion $f : B^2 \to B$ gegeben.
Welcher Ausdruck entspricht **nicht** dieser Funktion?

i	x_1	x_2	$f(x_1, x_2)$
0	0	0	1
1	0	1	0
2	1	0	1
3	1	1	1

(i) $\overline{f(x_1, x_2)} = \overline{(x_1 \cdot x_1)} \cdot x_2$	(ii) $f(x_1, x_2) = x_1 + \overline{x}_2$	(iii) $f(x_1, x_2) = \overline{(\overline{x}_1 \cdot x_2)}$	(iv) $f(x_1, x_2) = (x_1 + \overline{x}_2) \cdot (\overline{x}_1 + \overline{x}_2)$
e) Ein Carry-Save-Addiernetz...			
(i) ...berechnet Teilergebnisse doppelt und selektiert eines davon.	(ii) ...dient der schnellen Addition von mehr als zwei Summanden.	(iii) ...lässt den endgültigen Übertrag (von rechts nach links) durch das Schaltnetz rieseln.	(iv) ...führt die Additionsoperation auf die Subtraktion zurück.
f) Was ist kein Schritt im Pipelining einer klassischen RISC CPU?			
(i) Instruction Fetch	(ii) Instruction Decode	(iii) Erase	(iv) Execute (ALU)
g) Um welche Art von Hazards handelt es sich, wenn Entscheidungen (z.B. Sprünge) erst basierend auf einem späteren Ausführungsschritt oder dem noch ausstehenden Ergebnis einer vorangegangenen Instruktion getroffen werden können und dadurch das Pipelining erschwert wird?			
(i) Control Hazards	(ii) Biohazards	(iii) Data Hazards	(iv) Structural Hazards