KIT-Fakultät für Informatik
Prof. Dr.-Ing Uwe Hanebeck, Prof. Dr.-Ing. Jörg Henkel

Musterlösungen zur Klausur

Digitaltechnik und Entwurfsverfahren (TI-1)

und

Rechnerorganisation (TI-2)

am 26. Februar 2020, 13:00 – 15:00 Uhr

Name:	Vorname:	Matrikelnummer:
Bond	James	007

Digitaltechnik und Entwurfsverfahren (TI-1)	
Aufgabe 1	7 von 7 Punkten
Aufgabe 2	8 von 8 Punkten
Aufgabe 3	7 von 7 Punkten
Aufgabe 4	15 von 15 Punkten
Aufgabe 5	8 von 8 Punkten

Rechnerorganisation (TI-2)	
Aufgabe 6	12 von 12 Punkten
Aufgabe 7	10 von 10 Punkten
Aufgabe 8	12 von 12 Punkten
Aufgabe 9	11 von 11 Punkten

Gesamtpunktzahl:	90 von 90 Punkten

	Note:	**1,0**

Aufgabe 1 *Schaltfunktionen* (7 Punkte)

1. DNF von $f(c,b,a)$: $\boxed{\text{1 P.}}$

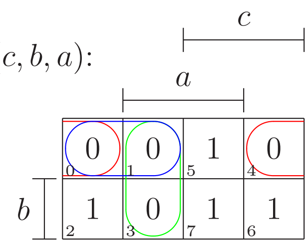
   (Ablesen aus der Tabelle)

$$\begin{aligned} f(c,b,a) &= \text{MINt}(2,5,6,7) \\ &= (\overline{c}\ b\ \overline{a}) \vee (c\ \overline{b}\ a) \vee (c\ b\ \overline{a}) \vee (c\ b\ a) \end{aligned}$$

1 Punkt nur für korrekte DNF

2. KV-Diagramm: $\boxed{\text{3 P.}}$

$f(c,b,a)$:



0,5 P für richtiges Eintragen in das KV-D

0,5 P für richtiges Einzeichnen der Blöcke

0,5 P für Kernimplikate

0,5 P für entbehrliche Implikate

1 P für die KMF

Primimplikate:

- Kernprimimplikate: $b \vee a$, $c \vee \overline{a}$
- Entbehrliche Primimplikate: $c \vee b$

KMF von $f(c,b,a)$: $(b \vee a) \wedge (c \vee \overline{a})$

3. Schaltnetz: $\boxed{\text{1 P.}}$

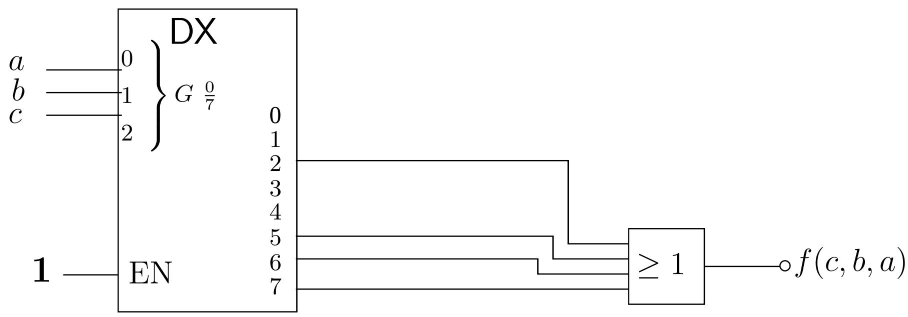


Abbildung 1: Schaltnetz

1 P für richtiges Schaltnetz

4. Schaltnetz: $\boxed{\text{2 P.}}$

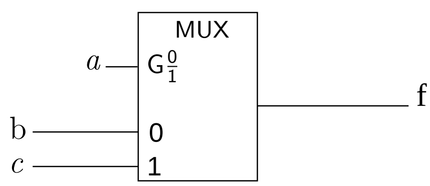
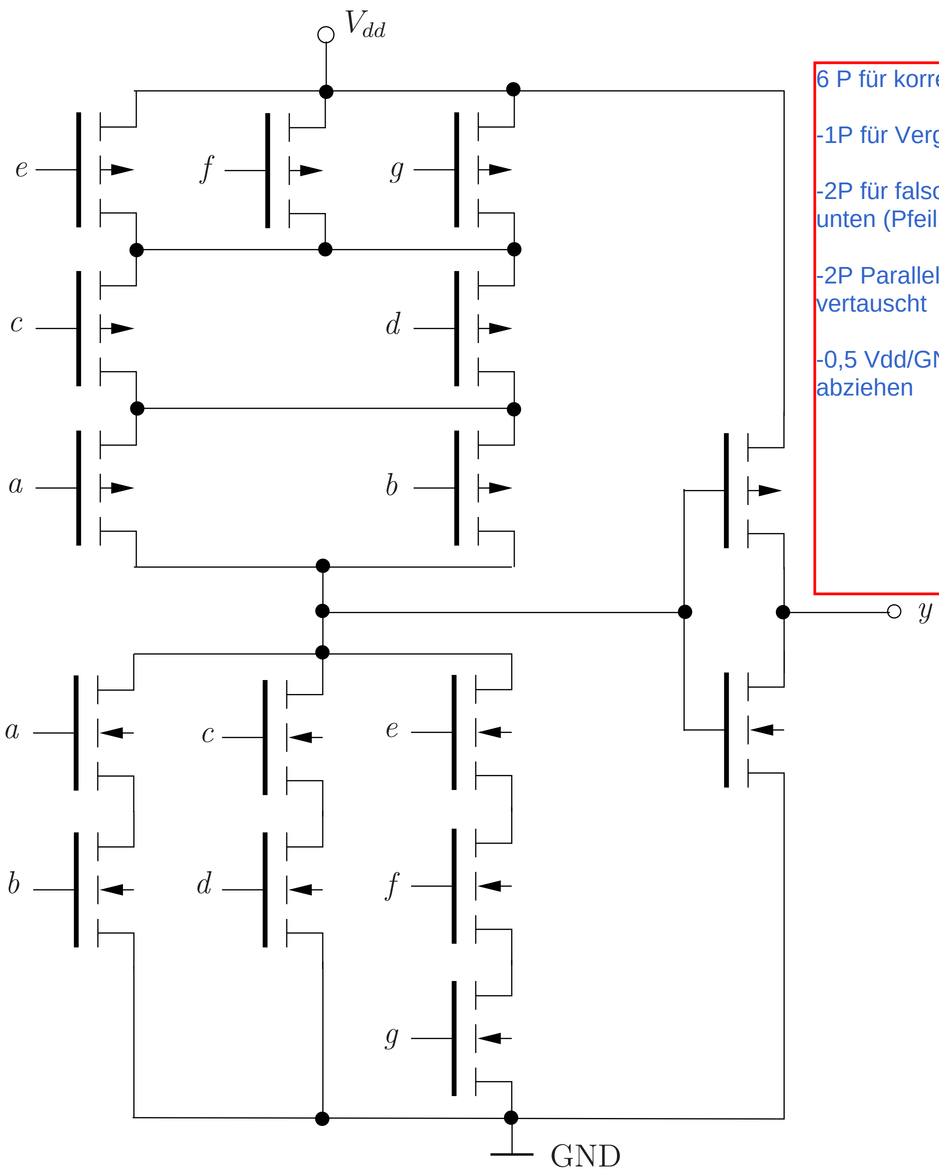


Abbildung 2: Schaltnetz

2 P für richtiges Schaltnetz

Aufgabe 2 *CMOS-Technologie* (8 Punkte)

1. CMOS-Schaltnetz: $\boxed{\text{6 P.}}$



6 P für korrektes Schaubild

-1P für Vergessen des NOTs am Ende

-2P für falsche Transistoren oben und unten (Pfeilrichtung falsch)

-2P Parallel- und Reihenschaltung vertauscht

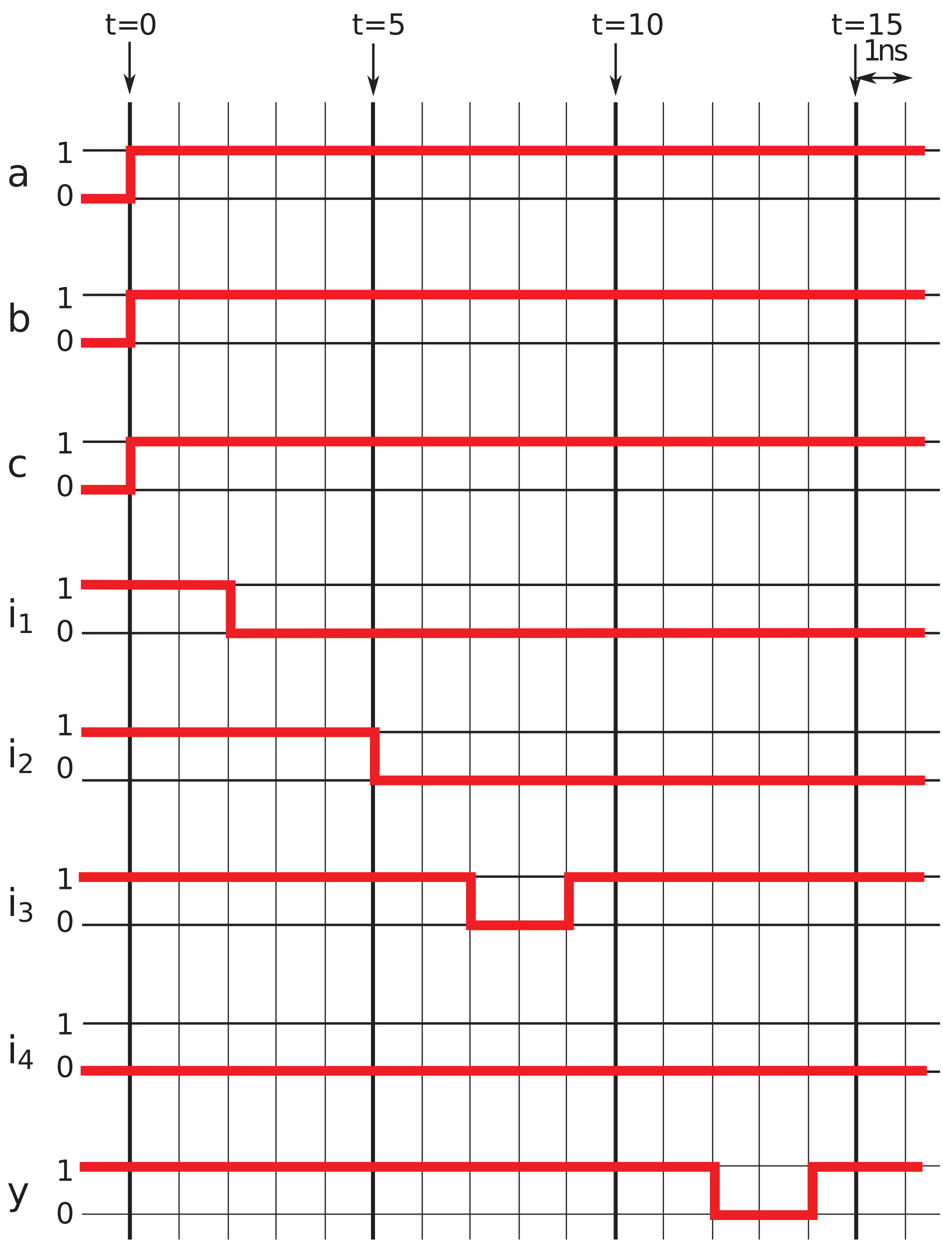
-0,5 Vdd/GND/y vergessen. Nur einmal abziehen

2. Schaltfunktion: $z = \overline{ad \vee bd \vee aef \vee bef \vee ced \vee cf}$ $\boxed{\text{2 P.}}$

2 P für die richtige Schaltfunktion
-0,5 P bei kleiner Abweichung (bei einem Literal etwas verdreht)
Volle Punktzahl für richitge Schaltfunktionen.

Aufgabe 3 *Laufzeiteffekte* (7 Punkte)

1. Zeitdiagramm: 4 P.



4 P für alle korrekten Verläufe

0,75 für jeden Verlauf von i1-i4

1 P für y

2. Hasardfehler (falls ja, Analyse): 3 P.

1 P für "Ja, es tritt auf"

1P für die Begründung (Stelle, nicht Monton, KV-Diagramm nicht zwingend, wenn vernünftig erklärt)

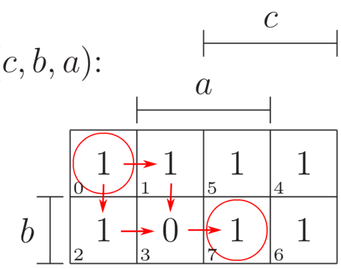
1P für 1-statischer Funktionshazard

Ja, es tritt ein Hasardfehler auf, denn y ist zu Beginn und Ende des Übergangs 1, wechselt während des Übergangs jedoch kurzzeitig auf 0.

Betrachtet man nun die möglichen Folgen von Funktionswerten beim Übergang $(0,0,0) \rightarrow (1,1,1)$, stellt man fest, dass die Folge von Funktionswerten bei bestimmten Eingabewechseln (z.B. a, b, c) nicht monoton ist (siehe KV-Diagramm auf nächster Seite). Somit handelt es sich um einen Funktionshasard.

Insgesamt ist der Hasard also als 1-statischer Funktionshasard zu klassifizieren.

$y(c, b, a)$:



Aufgabe 4 *Schaltwerke* (15 Punkte)

1. Anzahl der Zustände: 2 Flipflops $\Rightarrow 2^2$ Zustände $= 4$ Zustände

1P für richtiges Ergebnis

1 P.

2. Kodierte Ablauftabelle:

6 P.

- Ansteuergleichungen und Ausgabefunktion: Ablesen aus dem Schaltbild

$$
\begin{aligned}
j_B^t &= \overline{A}^t \nleftrightarrow x^t & \qquad k_B^t &= \overline{A}^t \nleftrightarrow x^t \\
j_A^t &= B^t & \qquad k_A^t &= x^t \\
y^t &= \overline{B}^t\,(\overline{A}^t \nleftrightarrow x^t)
\end{aligned}
$$

0,25 für jede Ansteuergleichung

1P für die Ausgabefunktion

- Übergangsgleichungen: Die charakteristische Gleichung des JK-Flipflops lautet:

$$q^{t+1} = (j\,\overline{q} \vee \overline{k}\,q)^t$$

Mit den Ansteuergleichungen der Flipflops erhält man die Zustandsübergangsgleichungen für A, und B:

$$
\begin{aligned}
B^{t+1} &= (\overline{A}^t \nleftrightarrow x^t)\overline{B}^t \vee \overline{(\overline{A}^t \nleftrightarrow x^t)}B^t \\
A^{t+1} &= B^t\,\overline{A}^t \vee \overline{x}\,A^t
\end{aligned}
$$

1P Zustandübergangsgleichung B

1P Zustandsübergangsgleichung A

Wahweise auch richtige Tabelle

- Kodierte Ablauftabelle:

Zustand		Eingabe	Folgezustand		Ausgabe
A^t	B^t	x^t	A^{t+1}	B^{t+1}	y^t
0	0	0	0	1	1
0	0	1	0	0	0
0	1	0	1	0	0
0	1	1	1	1	0
1	0	0	1	0	0
1	0	1	0	1	1
1	1	0	1	1	0
1	1	1	0	0	0

0,25 P für jede richitge Zeile

8 Zeilen ==> 2 P

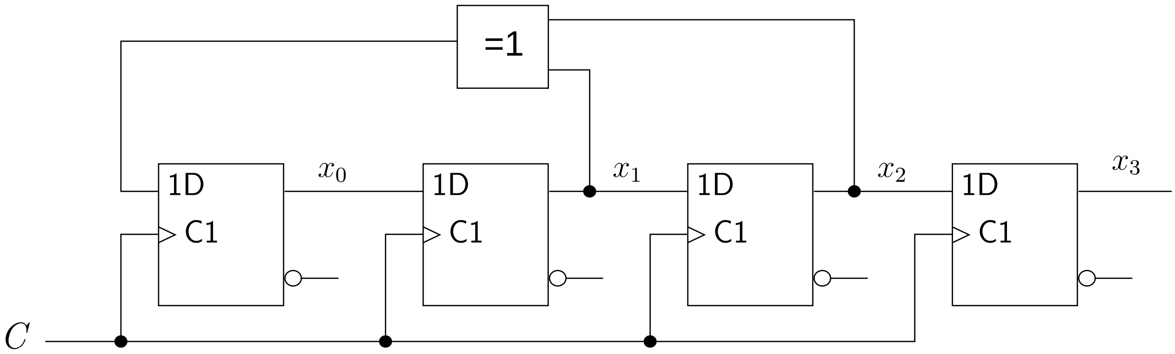
3. Eingabe-, Ausgabe-, und Zustandsfolgen: 4 P.

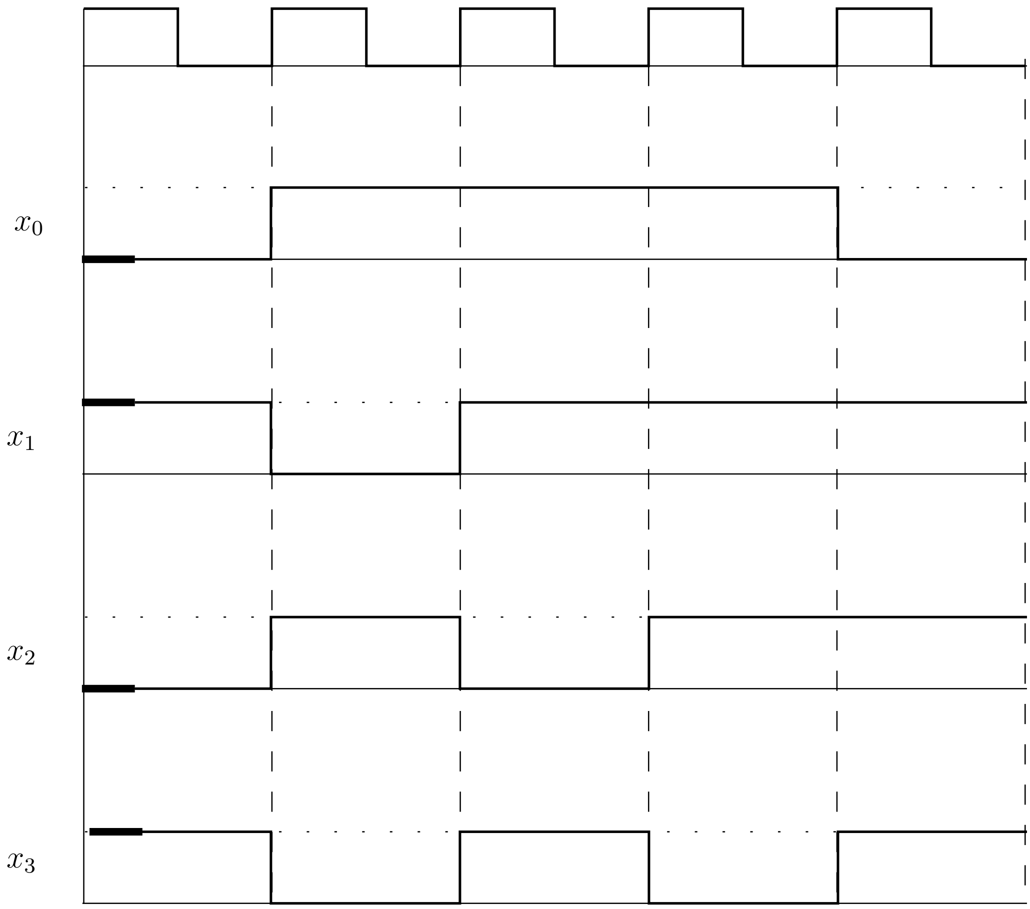
t	1	2	3	4	5	6	7	8	9	10
Z_i	Z_0	$\mathbf{Z_1}$	Z_2	$\mathbf{Z_3}$	Z_2	Z_0	$\mathbf{Z_3}$	Z_2	$\mathbf{Z_0}$	Z_1
$e(t)$	1	0	**1**	1	**0**	0	**1**	**0**	**1**	1
$a(t)$	0	0	0	**1**	**1**	**0**	1	**1**	**0**	1

8 Spalten sind mit Werten zu Füllen

für jede richtige Spalte 0,5 P

4. Verläufe der Signale x_0, x_1, x_2 und x_3: 4 P.





1P für jeden richtigen Signalverlauf.

Aufgabe 5 *Rechnerarithmetik* (8 Punkte)

1. Anzahl der Prüfbits: 1 P.

   Rechenweg 0,5P
   Ergebnis 0,5P

   Aufwand: $2^k \geq m + k + 1$. Hier: m = 100 $\Rightarrow k = 7$

2. $21,11_3$ in eine Dezimalzahl: 1 P.

$$21,11_3 = 2 \cdot 3^1 + 1 \cdot 3^0 + 1 \cdot 3^{-1} + 1 \cdot 3^{-2} = 7,\bar{4}_{10} \qquad (= 7 + \frac{4}{9})$$

   Rechenweg 0,5 P
   Ergebnis 0,5P

3. $F0,A1_{16}$ in eine Zahl zur Basis 8: 1 P.

$$F0,A1_{16} = 1111\ 0000\ ,\ 1010\ 0001_2 = 011\ 110\ 000,101\ 000\ 010_2 = 360,502_8$$

   Rechenweg 0,5 P
   Ergebnis 0,5P

4. Bereiche : 2 P.

   - Vorzeichen
   - Exponent
   - Mantisse

   2 P für alle drei Begründungen
   0,5 P für das Vorzeichen
   0,5 P für die 2er Potenz
   1P für den Überlauf

   Begründung:

   - -4 ist negativ $\Rightarrow$ Das Vorzeichen ändert sich.
   - Die Multiplikation mit einer 2er Potenz ändert den Exponenten.
   - Die Mantisse ändert sich am Randbereich. Wenn der Exponent überläuft, wird die Mantisse auf Null gesetzt.

5. 1001 1000 0000 0000 0000 0000 0001 0100: 3 P.

   (a) BCD: 98000014
   (b) Vorzeichenlose Dualzahl: $2^{31} + 2^{28} + 2^{27} + 2^4 + 2^2$
   (c) Gleitkomma-Zahl im IEEE-754-Standard in einfacher Genauigkeit:

$$\begin{aligned}
VZ &= 1 \\
Char &= 00110000 = 48 \\
Exp &= Char - 127 = -79 \\
M &= 00000000000000000010100 \Rightarrow \\
\\
Z &= (-1)^1 \cdot (1,00000000000000000010100) \cdot 2^{-79} \\
&= -(1 + 2^{-19} + 2^{-21}) \cdot 2^{-79}
\end{aligned}$$

   (a) Ergebnis 0,5P
   (b) Ergebnis 0,5P
   (c) Rechenweg 1P
   Ergebnis 1P

Aufgabe 6 *MIPS-Assembler* (12 Punkte)

1. Kontrollstruktur in MIPS-Assembler: 3 P.

0,5 P für die initialisierung der Zählervariablen

2P für das Umsetzen der for schleife (1P Abbruchbedingung und Sprung, 0,5 P i--, 0,5P Sprng zurück an schleifenanfang)

0,5P j = j*i

```
        addi    $a0, $zero, 100     # i = 100;
loop:   slt     $v0, $zero, $a0     # if 0 < i, dann $v0 = 1
        beq     $v0, $zero, label   # if $v0 = 0 (d.h., i <= 0),
                                    # dann Ende der for-Schleife
        mul     $a1, $a1, $a0       # j = j * i;
        subi    $a0, $a0, 1         # i--;
        b       loop                # gehe zur Marke loop
label:
```

2. Fehlerfreie Version: 3 P.

Grundlegend 3 Fehler und für jeden Fehler mit Korrektur 1 Punkt
- keine Initialisierung der Summe mit einem Null
- falsches Laden des Datums aus dem Array
- falsches Abbruchkriterium

```
           add   $v0, $zero, $zero    # summe = 0
array_sum: lw    $t0, 0($a0)          # Nächstes Element lesen
           add   $v0, $v0, $t0        # Addieren
           addi  $a0, $a0, 4          # Zeiger auf das nächste Element
           addi  $a1, $a1, -1         # Zähler dekrementieren
           bgtz  $a1, array_sum       # Abbruchbedingung
```

3. Inhalte der Zielregister: 3 P.

Pro richtige Zeile 0,5P.

keine Folgefehler!

Wenn alles richtig ist und kein Fehler auftrat --> 0,5P

Befehl	Zielregister = (z. B. `$s6 = 0x0000 F00A`)
`subi $s1, $zero, 0x2`	`$s1 = 0xFFFF FFFE`
`srl $s2, $s1, 4`	`$s2 = 0x0FFF FFFF`
`slti $s3, $s2, 100`	`$s3 = 0x0000 0000`
`lui $s4, 0x40`	`$s4 = 0x0040 0000`
`xor $s5, $s1, $s4`	`$s5 = 0xFFBF FFFE`

4. (a) Registerinhalte: 2 P.

Pro richtiges Register 0,5P

keine Folgefehler!

Register	Inhalt
`$t1`	`$t1 = 0x0000 000C`
`$t2`	`$t2 = 0x0000 001F`
`$t3`	`$t3 = 0x0000 0013`
`$t4`	`$t4 = 0x0000 0029`

(b) MIPS-Code zur Speicherung der Adresse von `vec` im Register `$s0`: 1 P.

1 P für den korrekten Code

```
la $s0, vec
```

Aufgabe 7 *Pipelining* (10 Punkte)

1. Datenabhängigkeiten: 3 P.

   - Echte Abhängigkeiten (*True Dependence* δ^t):

     `S1->S3 (R5)`   `S1->S5 (R5)`   `S2->S4 (R1)`   `S3->S4 (R4)`

   - Ausgabeabhängigkeiten (*Output Dependence* δ^o):

     `S2->S4 (R1)`

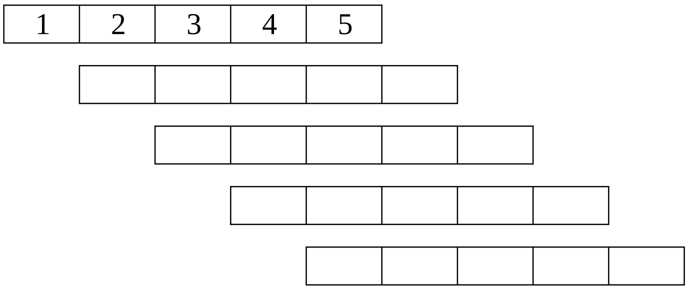
   - Gegenabhängigkeit (*Anti Dependence* δ^a):

     `S2->S3 (R4)`

Für jede richtig zugeordnete Abhängigkeit 0,5 P

6 x 0,5P = 3 P

2. Pipelinekonflikte: 3 P.



Bei echten Abhängigkeiten müssen mindestens 3 Befehle dazwischen liegen. Es führen folgende Abhängigkeiten zu Konflikten:

`S1->S3 (R5)`   `S2->S4 (R1)`   `S3->S4 (R4)`

Für jede gefundene Abhängigkeit 1P
3 x 1P = 3P

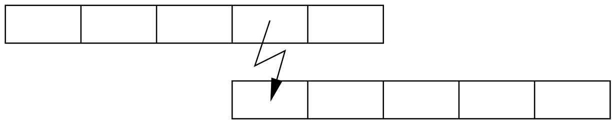
3. Beseitigung der Konflikte: 2 P.

```
S1:  add   R5, R2, R2
S2:  sub   R1, R4, R2
     NOP
     NOP
S3:  sub   R4, R5, R3
     NOP
     NOP
     NOP
S4:  add   R1, R1, R4
S5:  addi  R6, R5, 42
```

0,5P für je die richtige Stelle

0,5P für die richitge Anzahl an NOPs

4. Problem: 2 P.



Es entsteht ein Konflikt zwischen der Befehlshol-Phase und der Speicherzugriffs-Phase, da evtl. beide aus dem Cache lesen wollen.

1P für den "Ort" des Konflikts
1P für die richtige Begründung

Aufgabe 8 *Cache-Speicher* (12 Punkte)

0,5P Rechenweg
0,5P Ergebnis

1. (a) Blockgröße in Bytes: 5 Bit Byte-Offset ⇒ Blockgröße $= 2^5 = 32$ Byte **1 P.**

   (b) Anzahl der Einträge: **1 P.**

0,5P Rechenweg
0,5P Ergebnis

$$\text{Anzahl der Einträge} = \frac{\text{Kapazität}}{\text{Blockgröße}} = \frac{512\ KByte}{32\ Byte} = 16\ K\ \text{Einträge}$$

   (c) Cache-Organisation: **2 P.**

1P Rechenweg
1P Ergebnis

12 Bit Index-Feld ⇒ Es lassen sich $2^{12} = 4$ K Sätze im Cache addressieren

$$\text{Assoziativität} = \frac{16\ K}{4\ K} = 4$$

Der Cache ist als 4-fach assoziativer Speicher (*4-way set associative*) organisiert

2. Speicherbedarf: **3 P.**

0,5P Daten pro Zeile
0,5P Taggröße
1P Speicherbedarf einer Zeile
0,5P Rechenweg gesamter Cache
0,5P Ergebnis in Byte

Für jede Zeile sind (Tag + 1 Statusbit + Daten pro Zeile) Bits erforderlich.

- Daten pro Zeile 8 Byte $\times$ 8 = 64 Bit
- Tag = 32 - 8 - 3 = 21 Bit (8 Bit Satzindex und 3 Byte-Offset)

Speicherbedarf für eine Zeile: $21 + 1 + 64$ Bit $= 86$ Bits

Speicherbedarf für den gesamten Cache: $86 \cdot 256 \cdot 2 = 44032$ Bits $= 5504$ Byte

5 P.

3.

10 Spalten, für jede richtige Spalte 0,5P
Leer ergibt dennoch 0,5P

Adresse	**0**	**32**	**96**	**16**	**112**	**32**	**16**	**112**	**64**	**0**
read/write	r	r	r	r	r	r	r	w	w	r
Index	0	2	2	1	3	2	1	3	0	0
Tag	0	0	1	0	1	0	0	1	1	0
Hit/Miss	Miss	Miss	Miss	Miss	Miss	Miss	Hit	Hit	Miss	Hit

Aufgabe 9 *Virtuelle Speicherverwaltung* (11 Punkte)

1. Virtuelle und physikalische Adresse: $\boxed{2\text{ P.}}$

virtuelle Adresse	(V−P) Bits	P Bits
↓	↓	
physikalische Adresse	(M−P) Bits	P Bits

Für jedes richtige Feld 0,5P

1P pyh. Seiten
1P virt. Seiten

2. Es können $\left(\frac{2^M}{2^P} = 2^{M-P}\right)$ Seiten auf einmal im physikalischen Adressraum gespeichert werden. Anzahl der Einträge in der Seitentabelle ist gleich: $\left(\frac{2^V}{2^P} = 2^{V-P}\right)$ $\boxed{2\text{ P.}}$

1P Anzahl Bits pro Eintrag (Formel)
1P Anzahl Bits pro Eintrag (Ergebnis)
2P Anzahl der Seitentabelleneinträge (Rechenweg 1P Ergebnis 1p)
Mit 14 weitergerechnet, dennoch volle Punktzahl.

3. Anzahl der Bits pro Eintrag in der Seitentabelle ist gleich: $(M - P + 2)$. $\boxed{4\text{ P.}}$

   $V = 26$, $M = 20$ und $P = 8$ $\Rightarrow$ 14 Bits $\Rightarrow$ pro Eintrag sind 2 Bytes notwendig.

   2^{V-P} Einträge $\times$ 2 Bytes $= 2^{19}$ Bytes $=$ 512 KByte

   Anzahl der benötigten Seiten für die Seitentabelle: Eine Seite ist 2^8 Byte $=$ 256 Byte groß $\Rightarrow$ die Seitentabelle benötigt $\frac{512\text{KByte}}{256Byte} = 2K = 2048$ Seiten.

pro Adresse:

1P Rechenweg
0,5P Ergebnis

4. Physikalische Adresse 5348: $\boxed{3\text{ P.}}$

   $5348/2048 = 2 +$ Rest $1252 \Rightarrow$ virtuelle Seitennummer 2

   Aus der Tabelle $\Rightarrow$ physikalische Seitennummer 7

   $\Rightarrow$ physikalische Adresse ist: $7 * 2048 + 1252 = 15588$ oder $5348 + 5 * 2048 = 15588$

   Physikalische Adresse 6484:

   $6484/2048 = 3 +$ Rest $340 \Rightarrow$ virtuelle Seitennummer 3

   Aus der Tabelle $\Rightarrow$ physikalische Seitennummer 5

   $\Rightarrow$ physikalische Adresse ist: $5 * 2048 + 340 = 10580$ oder $6484 + 2 * 2048 = 10580$