Prof. Dr. Bernd Becker
Dipl.-Inf. Bettina Braitling
Dipl.-Inf. Sven Reimer

Freiburg, 17. Januar 2013

Technische Informatik
Übungsblatt Testat

Dieses Übungstestat dient Ihnen zur Vorbereitung auf die Abschlussklausur. Die Aufgaben sind vergleichbar mit einer realen Klausur, sowohl in Hinblick auf die Schwierigkeit als auch auf die Länge.

Das Übungstestat zählt nicht zum Zulassungskriterium, Sie müssen auch keine Lösung abgeben. Die angegebenen Punkte dienen lediglich zur Orientierung. Würde es sich hier um eine echte Klausur handeln, wären **40 Punkte** hinreichend zum Bestehen.

Wir empfehlen Ihnen, das Übungstestat mit einen **Zeitlimit von 90 Minuten** und **ohne Hilfsmittel** an einem Stück zu bearbeiten, um Klausurbedingungen zu simulieren.

Ab dem 23. Januar 2013 werden in den den Übungen nach und nach die Lösungen der Aufgaben vorgestellt.

Aufgabe 1 $(1 + 2 + 2 + 2 + 2$ Punkte)

a) Sei $A = \{a_1, \ldots, a_n\}$ ein endliches Alphabet und c eine Abbildung $c\colon A \to \mathbb{B}^*$ oder $c\colon A \to \mathbb{B}^n$ für ein beliebiges, aber festes $n \in \mathbb{N}$. Welche Bedingung muss gelten, damit c ein Code ist? Begründen Sie kurz.

b) Ersetzt man im Morsecode jeden Punkt „·“ durch eine 0 und jeden Strich „—“ durch eine 1, so erhält man die folgenden Abbildung:

Bst.	Code
A	01
B	1000
C	1010
D	100
E	0
F	0010

Bst.	Code
G	110
H	0000
I	00
J	0111
K	101
L	0100

Bst.	Code
M	11
N	10
O	111
P	0110
Q	1101
R	010

Bst.	Code
S	000
T	1
U	001
V	0001
W	011
X	1001

Bst.	Code
Y	1011
Z	1100

Ist der Morsecode ein Code? Begründen Sie kurz.

c) Was gilt zusätzlich für einen *Präfixcode*? (Es ist nicht notwendig, „Präfix“ zu definieren) (keine Begründung)

d) Nennen Sie sowohl

- einen Code, der ein Präfixcode ist als auch
- einen Code, der *kein* Präfixcode ist.

e) Welches Problem tritt bei einem Code auf, der *kein* Präfixcode ist? Illustrieren Sie anhand eines Beispiels.

Aufgabe 2 (2 + 8 Punkte)

a) Geben Sie die Interpretationsfunktion $[\cdot]_2$ für Zweierkomplementzahlen mit $n+1$ Vor- und k Nachkommastellen (also $d_n d_{n-1} \dots d_1 d_0 d_{-1} \dots d_{-k}$) an.

b) Zeigen Sie, dass für Zweierkomplementzahlen gilt:

$$[\overline{a}]_2 + 2^{-k} = -[a]_2$$

$\overline{a}$ entspricht dabei der bitweisen Komplementierung von a.

Hinweis: Für den Beweis darf die folgende Gleichung unbewiesen verwendet werden:

$$\sum_{i=-k}^{n-1} 2^i = 2^n - 2^{-k}$$

Aufgabe 3 (9 Punkte)

Weisen Sie die Korrektheit der Consensusregel durch Zurückführung auf die Axiome der Booleschen Algebra nach. Zeigen Sie also:

$$(x \cdot y) + (\neg x \cdot z) \quad = \quad (x \cdot y) + (\neg x \cdot z) + (y \cdot z)$$

und

$$(x + y) \cdot (\neg x + z) \quad = \quad (x + y) \cdot (\neg x + z) \cdot (y + z)$$

Geben Sie dabei bei jedem Schritt an, welche Axiome Sie verwenden.

Axiome der Boolschen Algebra:

(i)	Kommutativität	$x + y = y + x$	$x \cdot y = y \cdot x$
(ii)	Assoziativität	$x + (y + z) = (x + y) + z$	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$
(iii)	Absorption	$x + (x \cdot y) = x$	$x \cdot (x + y) = x$
(iv)	Distributivität	$x + (y \cdot z) = (x + y) \cdot (x + z)$	$x \cdot (y + z) = (x \cdot y) + (x \cdot z)$
(v)	Komplementregel	$x + (y \cdot \neg y) = x$	$x \cdot (y + \neg y) = x$

Aufgabe 4 (10 + 3 Punkte)

Gegeben sei die Funktion $f : \mathbb{B}^4 \to \mathbb{B}$ durch ihre **OFF**-Menge

$$\mathbf{OFF}(f) \quad = \{(0010), (0110), (0111), (1010), (1011)\}.$$

a) Berechnen sie alle Primimplikanten von f nach dem *Verfahren von Quine-McCluskey*. Geben Sie alle Zwischenschritte (d.h. die Mengen L_i^M und *Prim*) an.

b) Zeigen Sie anhand einer Primimplikantentafel (PIT), dass alle Primimplikanten wesentlich sind.

Aufgabe 5 (6 + 2 + 2 Punkte)

Hinweis: eine Kante eines Graphen kann beschrieben werden durch Quelle Q und Ziel Z der Kante. Beispiel: sei e eine Kante mit $Q(e) = v_1$ und $Z(e) = v_2$. Dann kann die Kante mit (v_1, v_2) beschrieben werden.

Gegeben sei der Schaltkreis SK_1 wie folgt:
$SK_1 := (X_3, G, typ, IN, Y)$, wobei

$$\begin{aligned}
X_3 &= \{x_1, x_2, x_3\} \\
Y &= \{y_1\} \\
G &= (V, E) \\
V &= X_3 \cup \{v_1, \ldots, v_7\} \cup Y \\
E &= \{(v_4, v_2), (v_6, v_2), (v_1, v_4), (x_2, v_5), (x_1, v_3), (x_3, v_5), \\
&\quad (v_3, v_7), (v_7, v_1), (v_2, y_1), (x_3, v_6), (x_1, v_7), (v_5, v_4)\} \\
typ &= \{(v_i \mapsto \texttt{NOT}) \mid i \in \{1, 3, 6\}\} \cup \{(v_i \mapsto \texttt{AND}) \mid i \in \{2, 7\}\} \cup \{(v_i \mapsto \texttt{OR}) \mid i \in \{4, 5\}\} \\
IN &= \{(v_1 \mapsto (v_7)), (v_2 \mapsto (v_4 v_6)), (v_3 \mapsto (x_1)), (v_4 \mapsto (v_1 v_5)), \\
&\quad (v_5 \mapsto (x_2 x_3)), (v_6 \mapsto (x_3)), (v_7 \mapsto (x_1 v_3))\}
\end{aligned}$$

a) Zeichnen Sie SK_1.

b) Geben Sie für SK_1 eine topologische Sortierung an.

c) Bestimmen Sie die Kosten sowie die Tiefe von SK_1, wobei die Kosten sich aus der Anzahl der benutzen Gatter ergibt und die Tiefe entspricht der Anzahl Gatter, die auf dem längsten Pfad von G liegen.

Aufgabe 6 $((3 + 3) + 5$ Punkte)

a) Betrachten Sie den PLA in Abbildung 1, der eine boolesche Funktion $g\colon \mathbb{B}^3 \to \mathbb{B}$ realisiert.

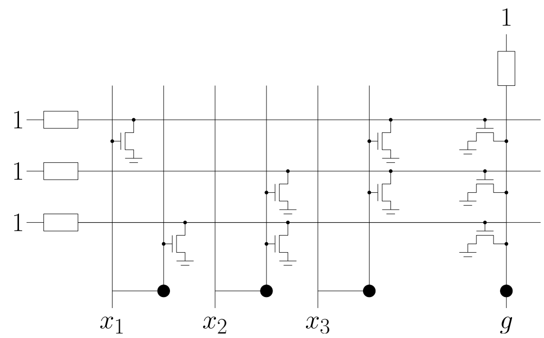


Abbildung 1: Ein PLA

1) Geben Sie das vom PLA dargestellte Polynom p_g zu g an.
2) Zeigen Sie, dass das Polynom p_g kein Minimalpolynom von g ist.

b) Zeigen Sie, dass es zu einer Booleschen Funktion f verschiedene Minimalpolynome geben kann.

Hinweis: Es genügt hierzu, z.B. eine Funktion $f\colon \mathbb{B}^3 \to \mathbb{B}$ zusammen mit ihren Primimplikanten anzugeben (ohne Beweis, dass die genannten Monome Primimplikanten sind), und aus diesen Primimplikaten zwei verschiedene Minimalpolynome zu bilden (ohne Beweis, dass die Polynome minimal sind). Sie können zur Herleitung der Minimalpolynome den 3-dimensionalen Würfel in Abb. 2 benutzen.

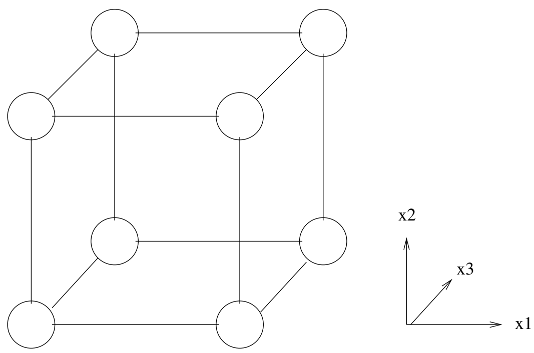


Abbildung 2: 3D-Würfel

Aufgabe 7 (3 + 9 Punkte)

a) Kommentieren Sie ausführlich die Auswirkungen jedes einzelnen Befehles des folgendes Programms. Eine Befehlsübersicht des ReTI-Rechners finden Sie am Ende des Blattes.

b) Das Programm ist in den Speicherzellen $M[0]$ bis $M[5]$ abgelegt. Wie ändert sich der Inhalt der Speicherzelle $M[0]$ während des Programmablaufs?
Welchen Inhalt hat die Speicherzelle $M[0]$ nach Ablauf des Programms?

Hinweise: Befehlscodierung für `LOADI IN2`: $I[31:24] = 01110010$.

Beachten Sie die Notation aus der Vorlesung:

$$b^j = \underbrace{(\mathtt{b}, \ldots, \mathtt{b})}_{\text{j mal}} \text{ für } b \in \{0, 1\}$$

Wie oft wird die Schleife durchlaufen? Begründen Sie Ihre Antwort.

`Programmcode:`

```
0 LOADI IN2 1^24
1 LOAD ACC 0
2 ADDI ACC 1
3 OR ACC 0
4 STORE 0
5 JUMP≥ -4
```

Aufgabe 8 (4 + 4 Punkte)

Prüfen Sie, ob die folgenden Befehle mit den vorgestellten Datenpfaden der ReTI und der vorgestellten groben zeitlichen Planung durch idealisierte Timing-Diagramme realisierbar sind. Vernachlässigen Sie dabei eventuelle Probleme mit der Codierung der Befehle und der Unterbringung neben den bereits definierten Befehlen.

Für jeden der Befehle:

- Ergänzen Sie (falls nötig) in dem entsprechenden Diagramm eine minimale Menge von zusätzlichen Treibern, um den Befehl für alle $S, D \in \{ACC, IN1, IN2, PC\}$ ausführen zu können.
- Markieren Sie exemplarisch für $S = IN1$ und $D = IN2$ die in der Execute-Phase aktiven Datenpfade bzw. die aktiven Treiber.
- Geben Sie im Kasten neben der ALU an, welche Operation die ALU ausführen muss.

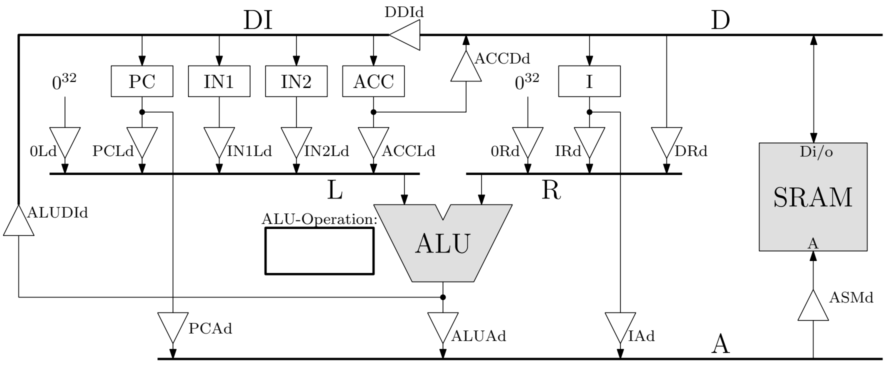
  Die ALU unterstütze dabei wie üblich die Operationen $([l] + [r])$, $([l] - [r])$, $([r] - [l])$, $(l \vee r)$, $(l \wedge r)$ und $(l \oplus r)$. l sei hierbei das Wort auf dem Bus L, r das Wort auf dem Bus R.

- Ist ein Befehl selbst mit zusätzlichen Treibern nicht realisierbar, begründen Sie dies kurz am Ende der Aufgabe.

---

a)

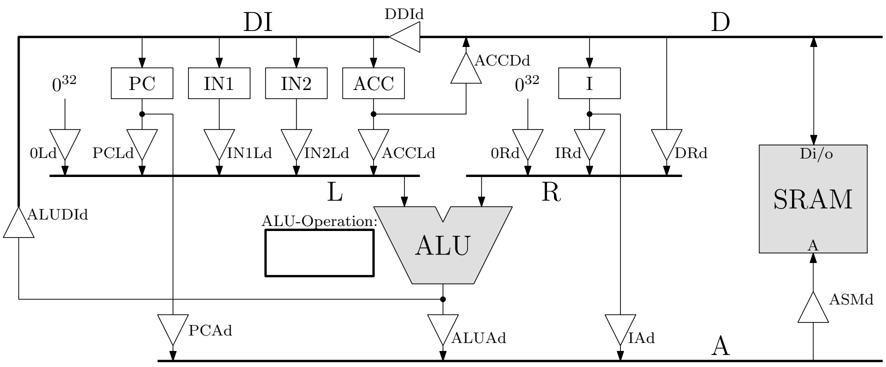
Befehl	Wirkung
LOADREL D i	$D := M(\langle PC \rangle + [i])$



---

b)

Befehl	Wirkung
STOREREL S i	$M(\langle PC \rangle + [i]) := S$



Aufgabe 9 (8 Punkte)

Zur Berechnung der Funktion $f = a \oplus b \oplus c \oplus d$ kann die Realisierung aus Abbildung 3 verwendet werden. Die Anstiegs- und Abfallzeiten an den primären Eingängen sind kleiner als $\delta = 0.13\,ns$. Weiterhin sind die Anstiegs- und Abfallzeiten an den Ausgängen eines Gatters kleiner als δ, falls die Anstiegs- und Abfallzeiten an den Eingängen des Gatters kleiner als δ sind. Alle primären Eingänge schalten zum Zeitpunkt t_0 auf die neuen logischen Werte, d.h. in dieser Aufgabe bezieht sich t_0 *nicht* auf M, sondern auf den Zeitpunkt zuvor, an dem die primären Eingänge umgeschaltet werden.

Bis zu welchem Zeitpunkt liegt an Signal f mindestens der alte logische Wert an und ab welchem Zeitpunkt liegt sicher der neue logische Wert an, wenn

a) ein $\oplus$-Gatter durch die Realisierung aus Abbildung 4 zusammengesetzt wird?

b) ein $\oplus$-Gatter durch die Realisierung aus Abbildung 5 zusammengesetzt wird?

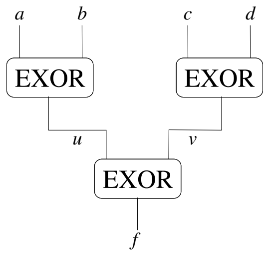


Abbildung 3: Realisierung der $\oplus$-Funktion mit 4 Eingängen

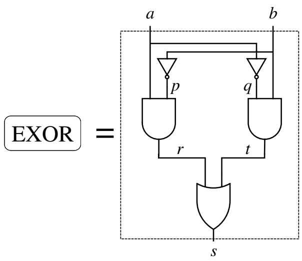


Abbildung 4: $\oplus$-Gatter mit NOT/AND/OR

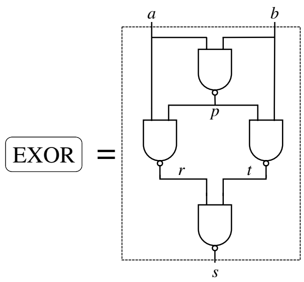


Abbildung 5: $\oplus$-Gatter mit NAND

	AND		NAND		OR		NOT	
	min	max	min	max	min	max	min	max
t_{PLH}	0.02	0.12	0.01	0.15	0.02	0.11	0.01	0.15
t_{PHL}	0.02	0.12	0.01	0.12	0.04	0.14	0.00	0.08

Tabelle 1: Verzögerungszeiten Grundgatter von NanGate

Load Befehle	$I[25,24] = D$		
$I[31,28]$	Befehl	Wirkung	
0100	LOAD D i	$D := M(\langle i \rangle)$	$\langle PC \rangle := \langle PC \rangle + 1$, falls $D \neq PC$
0101	LOADIN1 D i	$D := M(\langle \mathit{IN1} \rangle + [i])$	
0110	LOADIN2 D i	$D := M(\langle \mathit{IN2} \rangle + [i])$	
0111	LOADI D i	$D := 0^8 i$	

Store Befehle	MOVE: $I[27,26] = S$, $I[25,24] = D$		
$I[31,28]$	Befehl	Wirkung	
1000	STORE i	$M(\langle i \rangle) := ACC$	$\langle PC \rangle := \langle PC \rangle + 1$
1001	STOREIN1 i	$M(\langle \mathit{IN1} \rangle + [i]) := ACC$	
1010	STOREIN2 i	$M(\langle \mathit{IN2} \rangle + [i]) := ACC$	
1011	MOVE S D	$D := S$	$\langle PC \rangle := \langle PC \rangle + 1$, falls $D \neq PC$

Compute Befehle	$I[25,24] = D$		
$I[31,26]$	Befehl	Wirkung	
000010	SUBI D i	$[D] := [D] - [i]$	$\langle PC \rangle := \langle PC \rangle + 1$, falls $D \neq PC$
000011	ADDI D i	$[D] := [D] + [i]$	
000100	OPLUSI D i	$D := D \oplus 0^8 i$	
000101	ORI D i	$D := D \vee 0^8 i$	
000110	ANDI D i	$D := D \wedge 0^8 i$	
001010	SUB D i	$[D] := [D] - [M(\langle i \rangle)]$	$\langle PC \rangle := \langle PC \rangle + 1$, falls $D \neq PC$
001011	ADD D i	$[D] := [D] + [M(\langle i \rangle)]$	
001100	OPLUS D i	$D := D \oplus M(\langle i \rangle)$	
001101	OR D i	$D := D \vee M(\langle i \rangle)$	
001110	AND D i	$D := D \wedge M(\langle i \rangle)$	

Jump Befehle		
$I[31,27]$	Befehl	Wirkung
11000	NOP	$\langle PC \rangle := \langle PC \rangle + 1$
11001	$\text{JUMP}_{>}$ i	$\langle PC \rangle := \begin{cases} \langle PC \rangle + [i], & \text{falls } [ACC]\ c\ 0 \\ \langle PC \rangle + 1 & \text{sonst} \end{cases}$
11010	$\text{JUMP}_{=}$ i	
11011	$\text{JUMP}_{\geq}$ i	
11100	$\text{JUMP}_{<}$ i	
11101	$\text{JUMP}_{\neq}$ i	
11110	$\text{JUMP}_{\leq}$ i	
11111	JUMP i	$\langle PC \rangle := \langle PC \rangle + [i]$

Kodierung der Register: PC 00 / IN1 01 / IN2 10 / ACC 11

Tabelle 2: Befehlstabelle der ReTI

A1 a) injektiv

b) f^{-1}

c) Präfixcode darf kein Präfix eines anderen Codes sein

d) · Huffman

· Morse

A2 a) $\sum_{i=-k}^{n-1} d_i 2^i - d_n 2^n = [d_n d_{n-1} \dots d_{-k}]_2$

z.z.: $[\bar{a}]_2 + 2^{-k} = -[a]_2$

Beweis

$$[\bar{a}_n \bar{a}_{n-1} \dots \bar{a}_{-k}]_2 + 2^{-k} = \left(\sum_{i=-k}^{n-1} \bar{a}_i 2^i - \bar{a}_n 2^n\right) + 2^{-k}$$

$$= -(1-a_n)2^n + 2^{-k} + \sum (1-a_i) 2^i$$

$$= -2^n + 2^n a_n + 2^{-k} + \sum 2^i - \sum a_i 2^i$$

$$= \underbrace{\left(\sum a_i 2^i - 2^n a_n\right)}_{[a]_2} - 2^n + 2^{-k} + \sum 2^i$$

$$= -[a]_2 \underbrace{- 2^n + 2^{-k} + \sum 2^i}_{0}$$

$$= -[a]_2 \qquad \blacksquare$$

$\boxed{A3}$ 2te Gleichung:

Prinzip der Dualität

Zz. $(x \cdot y) + (\bar{x} \cdot z) = (xy) + (\bar{x}z) + (yz)$

Beweis: $(xy) + (\bar{x}z) \overset{\text{Absorption}}{=} (xy) + ((xy)z) + (\bar{x}z) + ((\bar{x}z)y))$

Kommutativität

$= (xy) + (\bar{x}z) + ((xy)z) + ((\bar{x}z)y))$

Assoziativität

$= (xy) + (\bar{x}z) + (x(yz)) + (\bar{x}(zy))$

Kommutativität

$= (xy) + (\bar{x}z) + (x(\overset{yz}{\cancel{zy}})) + (\bar{x}(yz))$

Distributivität

$= (xy) + (\bar{x}z) + ((x+\bar{x})(yz))$

Kommutativität

$= (xy) + (\bar{x}z) + ((yz)(x+\bar{x}))$

Komplementregel

$= (xy) + (\bar{x}z) + (yz)$ ∎

Teil 2.

Gilt durch Dualitätsprinzip

$\boxed{A4}$ a) $ON(f) = \mathbb{B}^4 \setminus OFF(f)$

$ON(f) = \{(0000), (0001), (0011), (0100), (0101), (1000), (1001), (1100), (1101), (1110), (1111)\}$

$L_0^{\{x_1, x_2, x_3, x_4\}} = ON(f)$

$Prim_0 = \emptyset$

$L_1^{\{\cancel{x_1}, x_2, x_3, x_4\}} = \begin{matrix} -000 \\ -001 \\ -100 \\ -101 \end{matrix}$

$L_1^{\{x_1, -, x_3, x_4\}} = \begin{matrix} 0-00 \\ 0-01 \\ 1-00 \\ 1-01 \end{matrix}$

$L_1^{\{x_1, x_2, x_3, -\}} = \begin{matrix} 000- \\ 010- \\ 100- \\ 110- \\ 111- \end{matrix}$

$L_1^{\{x_1, x_2, -, x_4\}} = \begin{matrix} 00-1 \\ 11-0 \\ 11-1 \end{matrix}$

$Prim = \emptyset$

$L_2^{\{-, -, x_3, x_4\}} = \begin{matrix} --00 \\ --01 \end{matrix}$

$L_2^{\{-, x_2, x_3, -\}} = \begin{matrix} -00- \\ -10- \end{matrix}$

$L_2^{\{x_1, -, x_3, -\}} = \begin{matrix} 0-0- \\ 1-0- \end{matrix}$

$L_2^{\{x_1, x_2, -, -\}} = 11--$

Rest! L_2-Mengen sind <u>leer</u>.

$Prim = \{00-1\}$

zu 4)

$L_3^{\{-,-,x_3,-\}} = --0-$

Restliche L_3-Mengen sind leer.

$Prim = \{(00-1), (11--)\}$

$L_4 = \emptyset$

$Prim = \{(00-1), (11--), (--0-)\}$

$\Rightarrow \bar{x}_1\bar{x}_2x_4 + x_1x_2 + \bar{x}_3$

b)

PIT	0	1	3	4	5	8	9	12	13	14	15
00_1		1	1								
11__								1	1	1	1
--0-	1	1		1	1	1	1	1	1		

$\boxed{A5}$ $SK = (X_3, G, typ, IN, Y)$ wobei

$X_3 = \{x_1, x_2, x_3\}$

$Y = \{y_1\}$

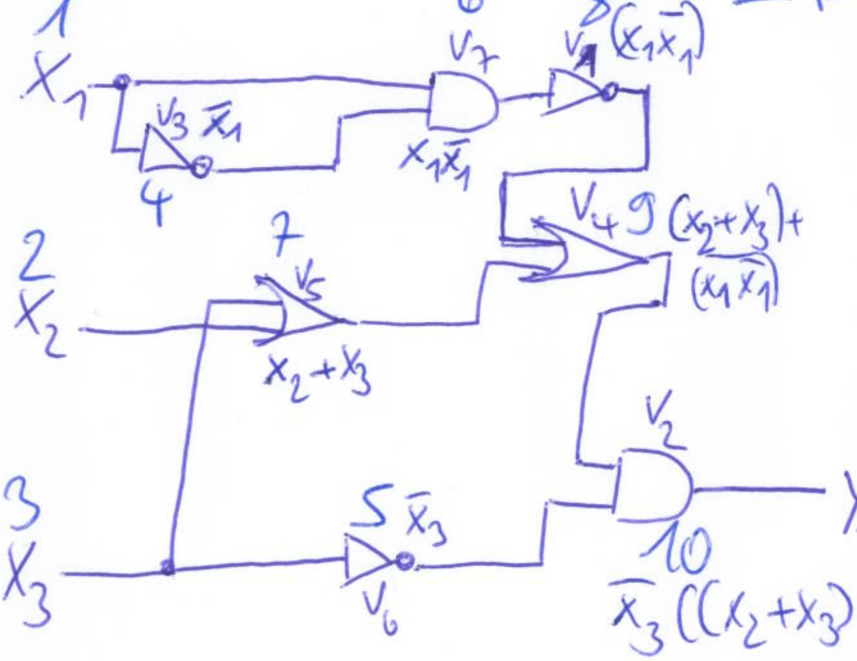
$G = \{V, E\}$

$V = X_3 \cup \{v_1, \dots, v_7\} \cup Y$

$E = \{(v_7, v_2), (v_6, v_2) \dots\}$

$typ = \{(v_i) \longmapsto NOT \mid i \in \dots\}$

$IN = \{(v_1) \longmapsto (v_7) \dots\}$



$\bar{x}_3((x_2+x_3)+\overline{(x_1\bar{x}_1)}$

$\bar{x}_3((x_2+x_3)+\overline{(x_1\bar{x}_1)})$

Topologische Sortierung:

1. X_1	6. v_7
2. X_2	7. v_5
3. X_3	8. v_1
4. v_3	9. v_4
5. v_6	10. v_2
	11. y_1

$f_{sort} = \{(x_1 \longmapsto 1), (v_3 \longmapsto 4), (x_2 \longmapsto 2), \dots\}$

ODER $\to \{(x_1, 1), (v_3, 4), \dots\}$

Tiefe: 5 Kosten: 7

A6 a) $P_g = \bar{x}_1 x_3 + x_2 x_3 + x_1 x_2$

II. $P'_g = \bar{x}_1 x_3 + x_1 x_2$ (Consensus Regel)

b) Sei $f = \bar{x}_1 x_2 + \bar{x}_2 x_3 + \bar{x}_3 x_1$

$prim(f) = \{\bar{x}_1 x_2, \bar{x}_2 x_3, \bar{x}_3 x_2, \bar{x}_1 x_3, \bar{x}_3 x_2\}$

$P_f = \bar{x}_1 x_2 + \bar{x}_2 x_3 + \bar{x}_3 x_1$

$P'_f = \bar{x}_2 x_1 + \bar{x}_1 x_3 + \bar{x}_3 x_2$

A7
```
LOADI IN2 1^24    //Lade 0^8 1^24 in IN2
LOAD ACC 0        // Lade M(0) in ACC
ADDI ACC 1        // ACC = ACC+1
OR ACC 0          // bitweise verodern ACC und M(0)
STORE 0           //speichere ACC in M(0)
JUMP GT -4        //Wenn ACC ≥ 0 -> PC := PC - 4
```

zu A7] Init: $M(0) = 01110010\ 1^{24}$

1. Schleife: $01110011\ 1^{24}$
2. Schleife: $01110111\ 1^{24}$
3. Schleife: $01111111\ 1^{24}$
4. Schleife: $11111111\ 1^{24}$

$+1 \rightarrow 01110100\ 0^{24}$
$\rightarrow 01110011\ 1^{24}$
OR $\rightarrow 01110111\ 1^{24}$
↓
usw.
↓

In M(0) steht dann $11111111\ 1^{24}$, die Bedingung ≥ 0 ist nicht mehr gegeben. Somit terminiert das Programm nach 4 Schleifen durchlaufen.

A8

a) realisierbar
aktive Treiber: PCLd, IRd, ALUAd, ASMd, DDId
ALU: [l]+[r]

b) nur r mit PCDd, IN1Dd, IN2Dd
aktive Treiber: IRd, PCLd, ALUAd, ASMd, SDd
ALU: [l]+[r]

A9 I: $t(q) = [0.0 + t_0,\ 0.15 + t_0] = t(p)$

a)

$$t(t) = [0.0 + t_0,\ 0.15 + t_0] + [0.02,\ 0.12] = [0.02 + t_0,\ 0.27 + t_0] = t(r)$$

$$t(s) = [0.02 + t_0,\ 0.27 + t_0] + [0.02,\ 0.14] = [0.04 + t_0,\ 0.41 + t_0]$$

$$\Rightarrow t(u) = t(v) = t(s)$$

$$t(r') = \overset{t(s)}{[0.04 + t_0,\ 0.41 + t_0]} + \overset{NOT}{[0.0,\ 0.15]} + \overset{AND}{[0.02,\ 0.12]}$$

$$= [0.06 + t_0,\ 0.68 + t_0] = t(t')$$

$$t(t) = [0.06 + t_0,\ 0.68 + t_0] + [0.02,\ 0.14] = [0.08 + t_0,\ 0.82 + t_0]$$

min: $0.08 + t_0 + \delta = (-0.05 + t_0)\,ns$

max: $0.82 + t_0 + 2\delta = (1.08 + t_0)\,ns$

b) analog