Institut für Informatik

Prof. Dr. Christoph Scholl
Dr. Paolo Marin

Freiburg, 30. Januar 2014

Testat

Technische Informatik

Name: _____ **Matrikel-Nr.:** _____

Umfang: 24 Seiten **Bearbeitungszeit:** 90 Minuten

Erlaubte Hilfsmittel: Keine **Übungsgruppe:** _____

Bitte prüfen Sie, ob Sie **alle Aufgabenblätter** erhalten haben und tragen Sie auf **allen** verwendeten Blättern (auch den zusätzlich ausgeteilten) Ihren **Namen** und Ihre **Matrikelnummer** ein. Blätter ohne diese Information werden nicht berücksichtigt.

Aufgabe	Punktzahl möglich	erreicht
1	6	
2	12	
3	8	
4	9	
5	12	
6	19	
7	8	
8	8	
9	8	
Summe	90	

Das Erreichen von **40** Punkten wre hinreichend zum Bestehen, wrde es sich bei dem Übungstestat um eine echte Klausur handeln.

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 Testat zählt nicht zum Zulassungskriterium. Die angegebenen Punkte dienen lediglich zur Orientierung. Würde es sich hier um eine echte Klausur handeln, wären **40 Punkte** hinreichend zum Bestehen.

Sie dürfen das Übungstestat mit einen **Zeitlimit von 90 Minuten** und **ohne Hilfsmittel**, um Klausurbedingungen zu simulieren.

Nächste Woche werden in den Übungen die Lösungen der Aufgaben vorgestellt.

Aufgabe 1 (6 Punkte)

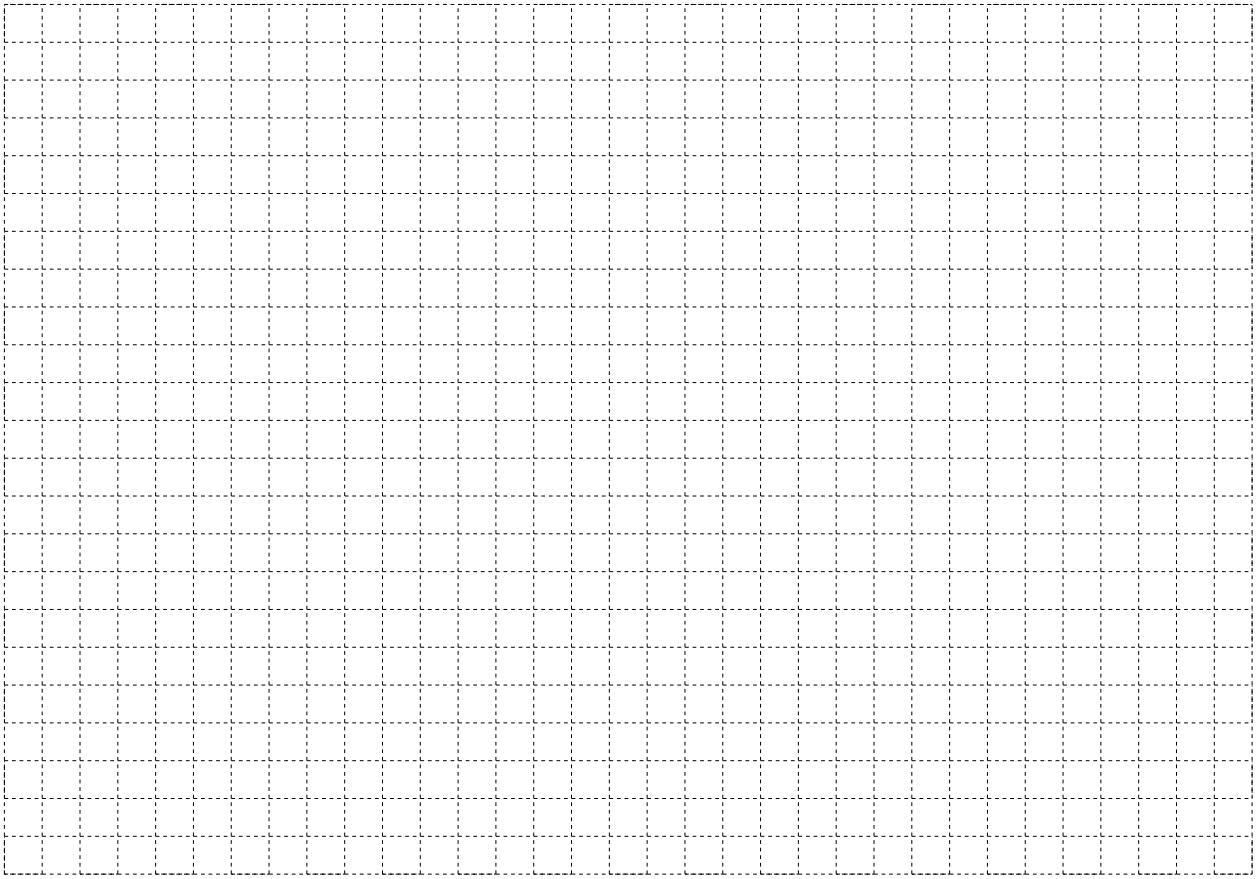
Sei $\mathcal{B} = (\mathbf{M}, \cdot, +, \overline{\ })$ eine *Boolesche Algebra*. Beweisen Sie die folgende Beziehung:

$$\forall x_1, x_2 \in M : \overline{x_1 \cdot \overline{x_2} + \overline{x_1} \cdot x_2} \;=\; x_1 \cdot x_2 + \overline{x_1} \cdot \overline{x_2}$$

Benutzen Sie für den Beweis nur die im folgenden angegebenen Axiome und Regeln der Booleschen Algebra und geben Sie in jedem Schritt an, welches Axiom Sie benutzen. Ausnahme: Kommutativität und Assoziativität müssen nicht angegeben werden. Ein "Beweis" mit einer Funktionstabelle ist nicht zulässig, da diese Beziehung für beliebige Boolesche Algebren gilt.

(i) Kommutativität	$a + b = b + a$	$a \cdot b = b \cdot a$
(ii) Assoziativität	$a + (b + c) = (a + b) + c$	$a \cdot (b \cdot c) = (a \cdot b) \cdot c$
(iii) Absorption	$a + (a \cdot b) = a$	$a \cdot (a + b) = a$
(iv) Distributivität	$a + (b \cdot c) = (a + b) \cdot (a + c)$	$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
(v) Komplementregel	$a + (b \cdot \overline{b}) = a$	$a \cdot (b + \overline{b}) = a$
(vi) de-Morgan	$\overline{(a + b)} = \overline{a} \cdot \overline{b}$	$\overline{(a \cdot b)} = \overline{a} + \overline{b}$
(vii)Doppeltes Komplement	$\overline{\overline{a}} = a$	

Ihre Lösung zu Aufgabe 1:



Aufgabe 2 (6 + 2 + 2 + 2 Punkte)

Gegeben sei ein Alphabet A mit $A = \{a, b, c, d, e, f\}$. Die Zeichen des Alphabets A treten mit folgenden Wahrscheinlichkeiten in Nachrichten auf:

$P(a) = 0.46$, $P(b) = 0.30$, $P(c) = 0.08$, $P(d) = 0.07$, $P(e) = 0.05$, $P(f) = 0.04$

Für die Datenübertragung von Nachrichten über dem Alphabet A soll eine Huffman-Kodierung verwendet werden.

a) Konstruieren Sie einen entsprechenden Huffman-Baum. Ordnen Sie hierfür die Blätter von links nach rechts mit absteigender Häufigkeit an. Markieren Sie die linken Kanten mit 0 und die rechten Kanten mit 1.

b) Geben Sie das Huffman-Codewort $c(x)$ für jedes Zeichen $x \in A$ an.

c) Kodieren Sie die Nachricht *baff* mit Ihrer Huffman-Kodierung. Zur besseren Lesbarkeit fügen Sie bitte jeweils ein Komma zwischen die einzelnen Codewörter ein.

d) Betrachten Sie folgenden Code c über das Alphabet $B = \{u, v, w, x, y, z\}$:

$a \in B$	$c(a)$
u	111
v	01
w	1101
x	1100
y	0
z	10

Kann es sich hierbei um einen Huffman-Code handeln? Begründen Sie Ihre Antwort.

Ihre Lösung zu Aufgabe 2:

Ihre Lösung zu Aufgabe 2 (Fortsetzung):

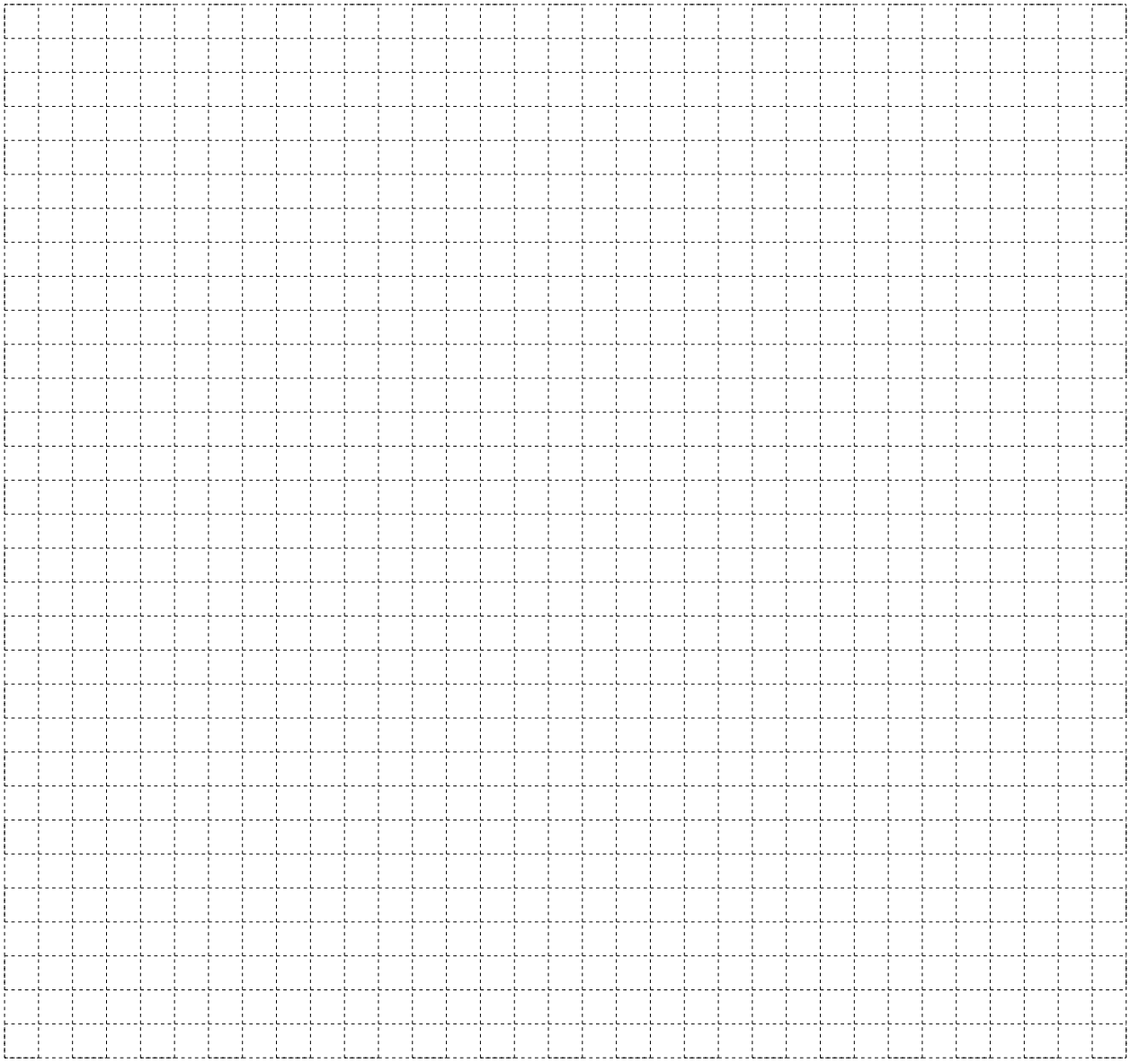
Aufgabe 3 (2 + 8 Punkte)

Sei $d = d_n d_{n-1} \dots d_0$ eine Festkommazahl mit $n+1$ Vorkomma- und 0 Nachkommastellen in *Einer-Komplement-Darstellung*.

a) Geben Sie die Interpretationsfunktion $[\cdot]_1 : \mathbb{B}^{n+1} \to \mathbb{N}$ für Einer-Komplement-Zahlen mit $n+1$ Vorkomma- und 0 Nachkommastellen an.

b) Zeigen Sie formal: Einer-Komplement-Zahlen mit $n+1$ Vorkomma- und 0 Nachkommastellen können negiert werden, indem man jedes Bit komplementiert.

Ihre Lösung zu Aufgabe 3:



Aufgabe 4 (3 + 6 Punkte)

Eine *topologische Sortierung* der Knoten eines gerichteten Graphen $G = (V, E)$ wird definiert als eine bijektive Abbildung

$$tsort : V \mapsto \{1, \ldots, |V|\},$$

für die gilt:

$$\forall v \in V : (\forall w \in V : ((w, v) \in E \Rightarrow tsort(w) < tsort(v)))$$

Eine topologische Sortierung eines Schaltkreises ist eine topologische Sortierung der Knoten des Schaltkreis-Graphens.

a) Zu einem azyklischen, gerichteten Graphen $G = (V, E)$ existiere die topologische Sortierung $tsort_1$. Ist diese eindeutig? (Beweis oder Gegenbeispiel)

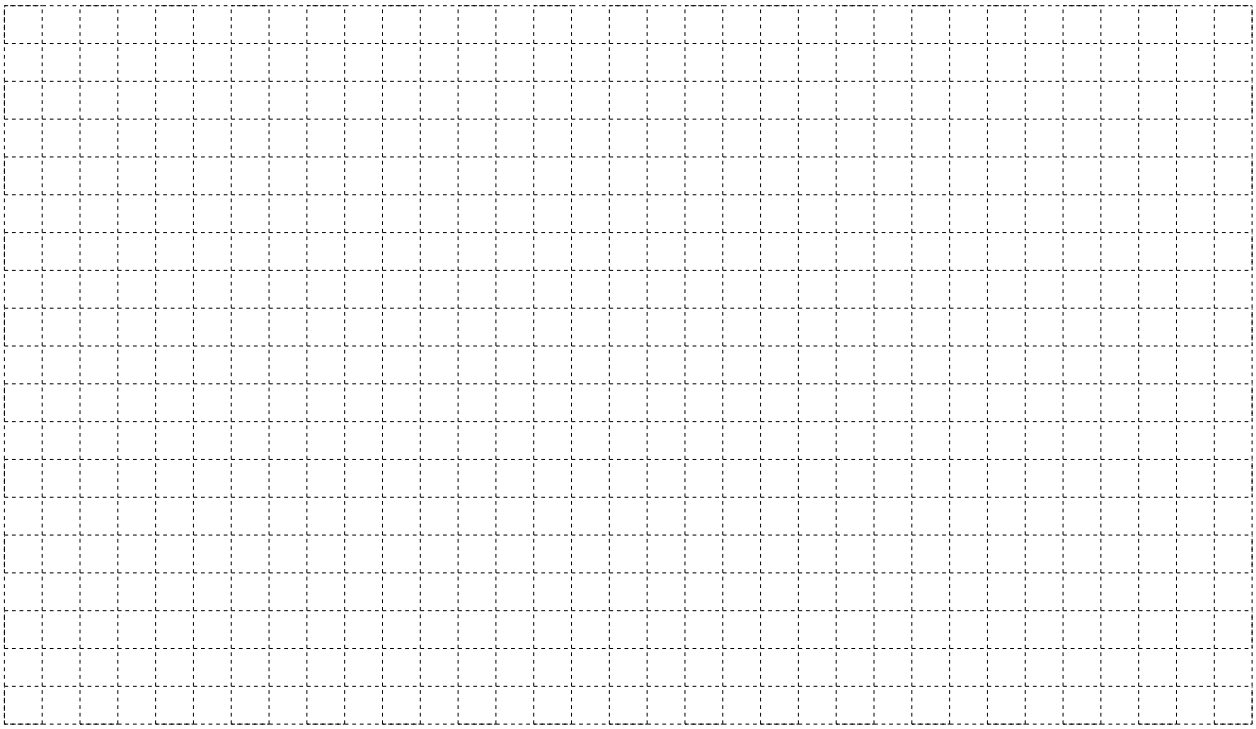
b) Zeigen Sie: Für jeden azyklischen, gerichteten Graphen $G = (V, E)$ existiert eine topologische Sortierung.

Hinweis: Zeigen Sie zuerst, dass gilt:

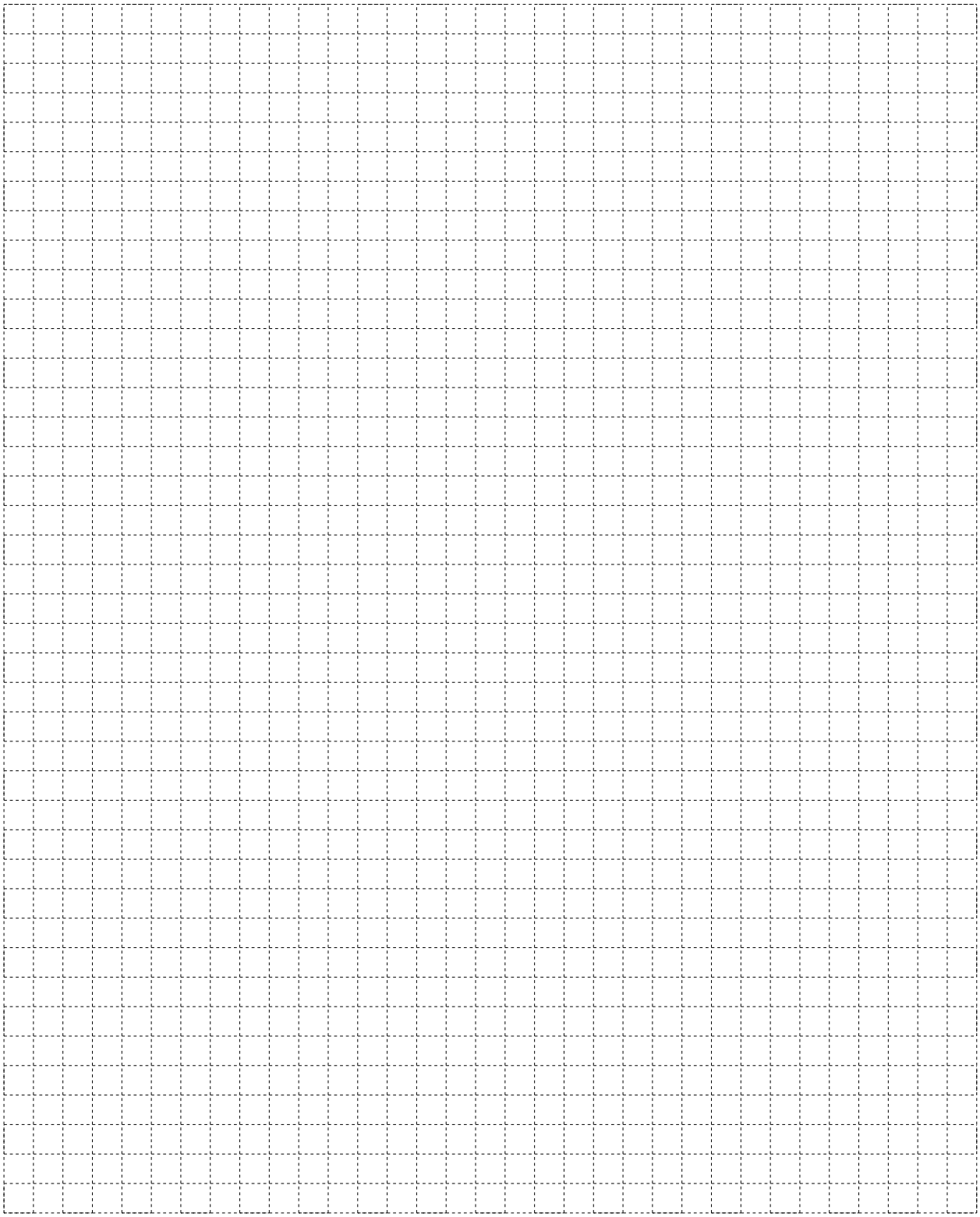
$$G = (V, E)\ azyklisch \Rightarrow \exists v \in V : indeg(v) = 0.$$

Beweisen sie anschliessend die Korrektheit der Aussage durch Induktion über die Anzahl von Knoten des Graphen.

Ihre Lösung zu Aufgabe 4:



Ihre Lösung zu Aufgabe 4 (Fortsetzung):



Aufgabe 5 (5 + 2 + 3 + 2 Punkte)

Gegeben sei der Schaltkreis SK_1 wie folgt:
$SK_1 := (X_3, G, typ, IN, Y_2)$, wobei

$$
\begin{aligned}
X_3 &= (x_1, x_2, x_3) \\
Y_2 &= (v_4, v_2) \\
G &= (V, E) \\
V &= \{x_1, x_2, x_3\} \cup \{v_1, v_2, v_3, v_4, v_5\} \\
E &= \{(v_1, v_2), (v_3, v_2), (x_1, v_1), (x_2, v_1), (x_1, v_5), (x_2, v_5), \\
&\quad (v_5, v_3), (x_3, v_3), (v_5, v_4), (x_3, v_4)\} \\
typ &= \{(v_1 \mapsto and_2), (v_2 \mapsto or_2), (v_3 \mapsto and_2), (v_4 \mapsto exor_2), (v_5 \mapsto exor_2)\} \\
IN &= \{(v_1 \mapsto ((x_1, v_1), (x_2, v_1))), (v_2 \mapsto ((v_1, v_2), (v_3, v_2))), \\
&\quad (v_3 \mapsto ((v_5, v_3), (x_3, v_3))), (v_4 \mapsto ((v_5, v_4), (x_3, v_4))), (v_5 \mapsto ((x_1, v_5), (x_2, v_5)))\}
\end{aligned}
$$

Hierbei wurde, anders als in der Vorlesung, eine verkürzte Schreibweise für die Kanten gewählt: wir definieren die Kantenmenge als $E \subseteq V \times V$. Gilt für eine Kante $e = (v_i, v_j)$, so ist $Q(e) = v_i$ und $Z(e) = v_j$.

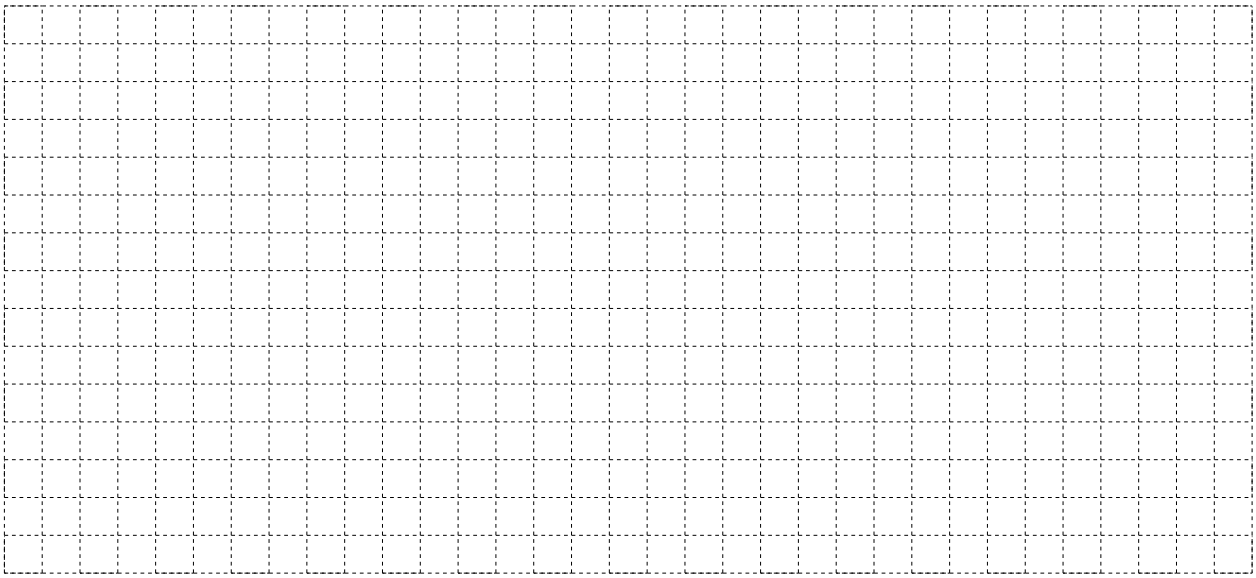
a) Zeichnen Sie SK_1.

b) Geben Sie für SK_1 eine topologische Sortierung an.

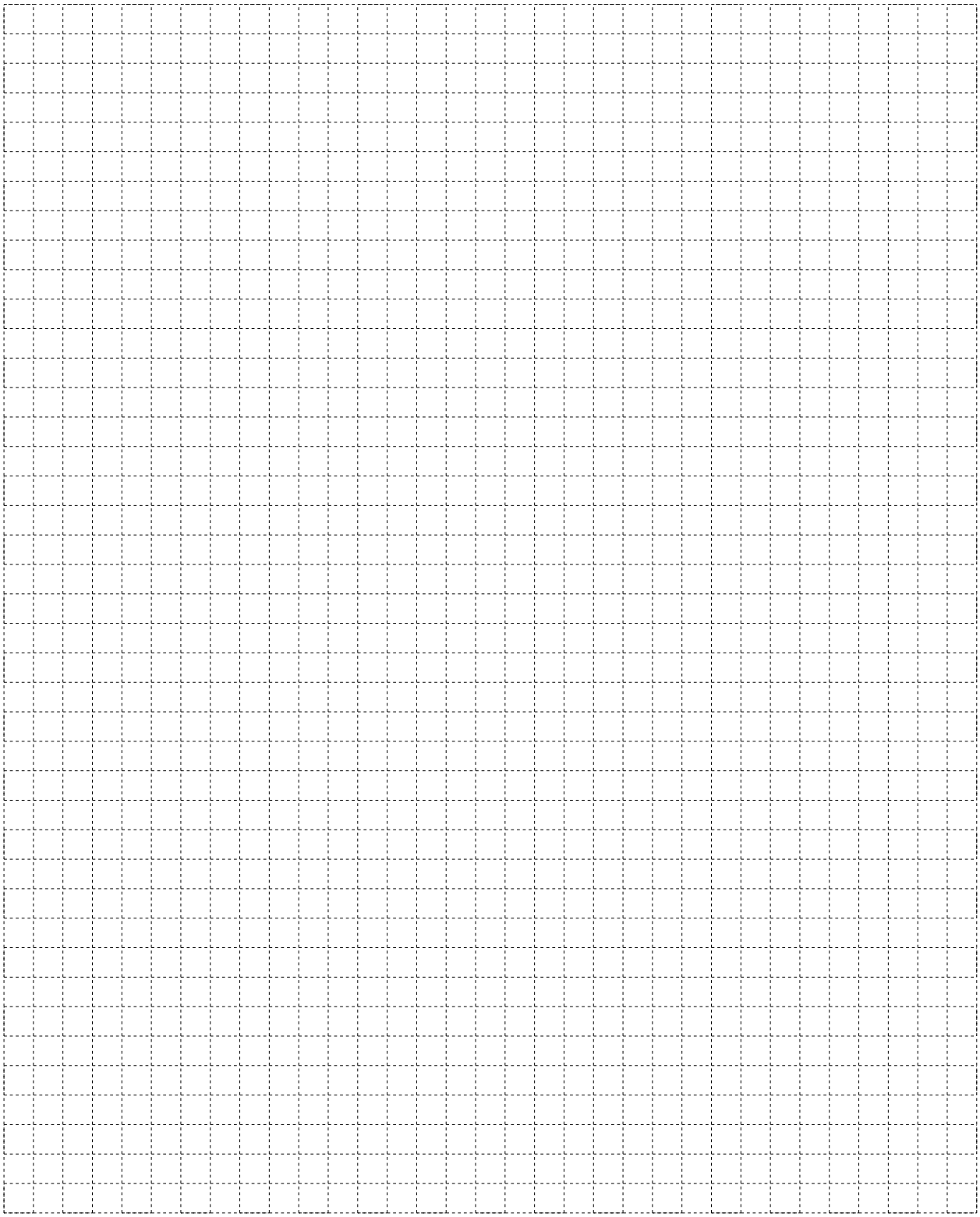
c) Führen Sie für SK_1 eine symbolische Simulation durch.

d) Bestimmen Sie die Kosten sowie die Tiefe von SK_1.

Ihre Lösung zu Aufgabe 5:



Ihre Lösung zu Aufgabe 5 (Fortsetzung):



Aufgabe 6 (6 + 3 + 8 + 2 Punkte)

Betrachten Sie den PLA in Abbildung 1, der die beiden Funktionen $f_1, f_2 \in \mathbb{B}_4$ realisiert. Inverter sind in dieser Abbildung als schwarze Punkte dargestellt.

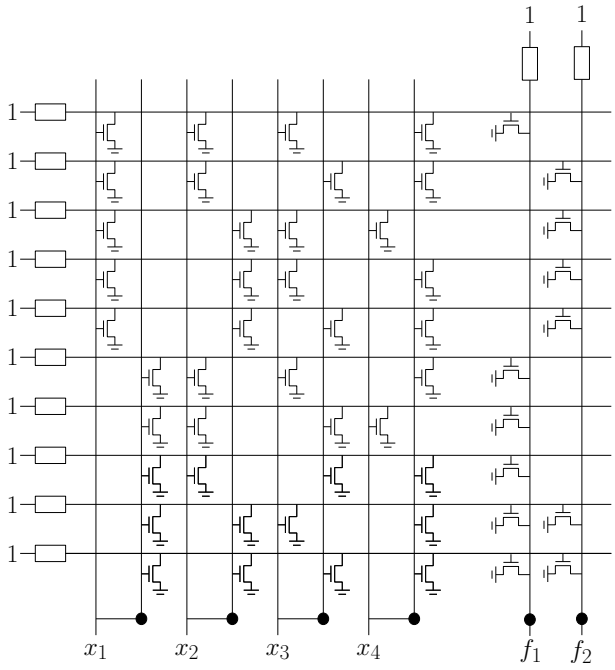
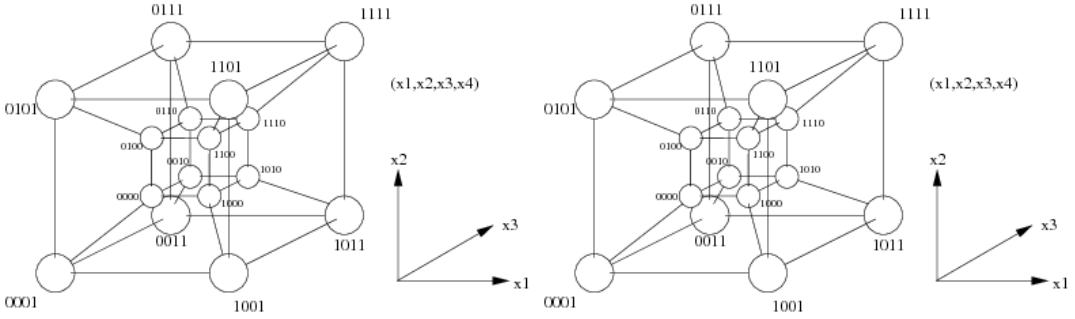


Abbildung 1: Ein PLA

a) Erstellen Sie die Polynome p_1 und p_2, die durch die PLA-Realisierung der Funktionen f_1 und f_2 gegeben ist. (Sie sollten Disjunktion von Mintermen erhalten).

   Geben Sie die Kosten $cost(p_1, p_2) = (cost_1(p_1, p_2), cost_2(p_1, p_2))$ des in Abbildung 1 gegebenen PLA an. (ohne Begründung)

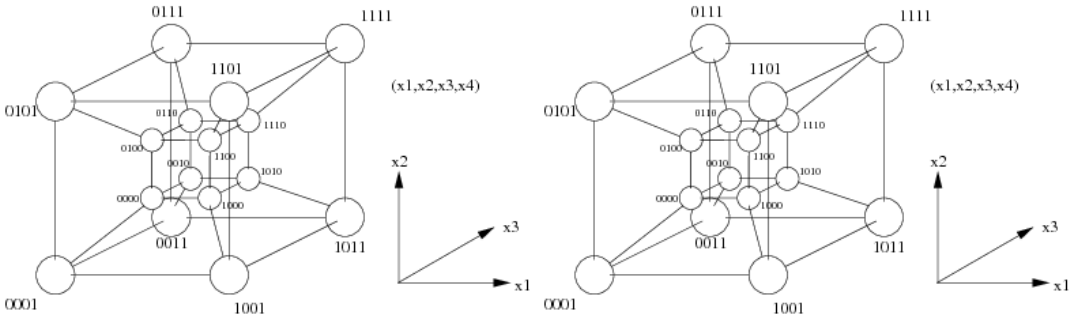
b) Zeichnen Sie für f_1 und f_2 jeweils einen 4-dimensionalen Würfel (Hypercube), in dem $ON(f_1)$ bzw. $ON(f_2)$ markiert ist.



c) Bestimmen Sie getrennt für p_1 und p_2 mit Hilfe des Quine-McCluskey-Algorithmus' die jeweiligen Primimplikanten. Geben Sie hierbei jeweils für jeden Schritt i des Algorithmus' die

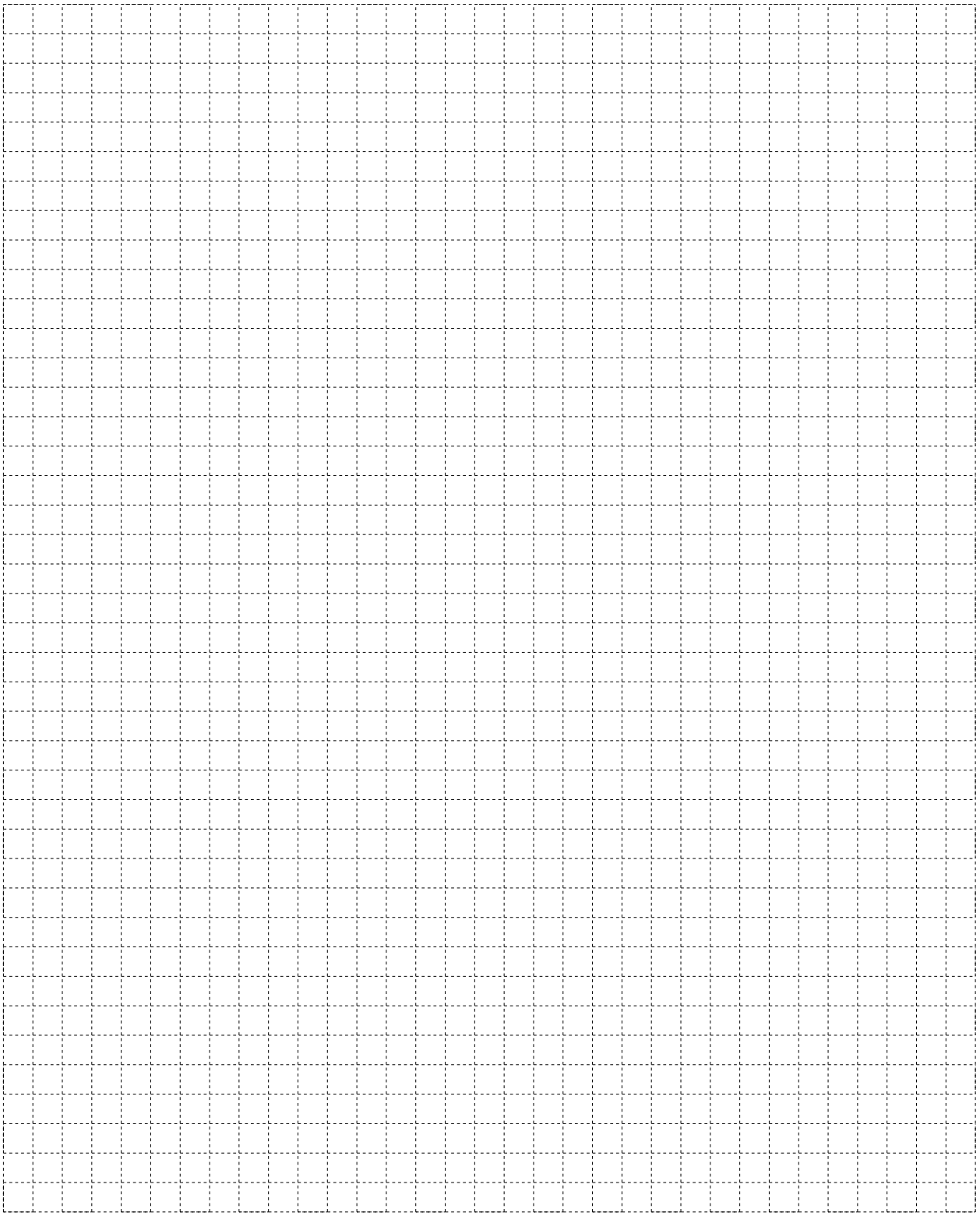
Mengen $L_i^M(f_j)$ und $Prim(f_j)$ an, entweder als Menge von Monomen oder unter Verwendung der in der Vorlesung benutzten abkürzenden Schreibweise (in der Form „`01-1`“).

d) Zeichnen Sie die den Primimplikanten von f_1 bzw. f_2 entsprechenden Teilwürfel in den jeweiligen Hypercube ein.



Ihre Lösung zu Aufgabe 6:

Ihre Lösung zu Aufgabe 6 (Fortsetzung):



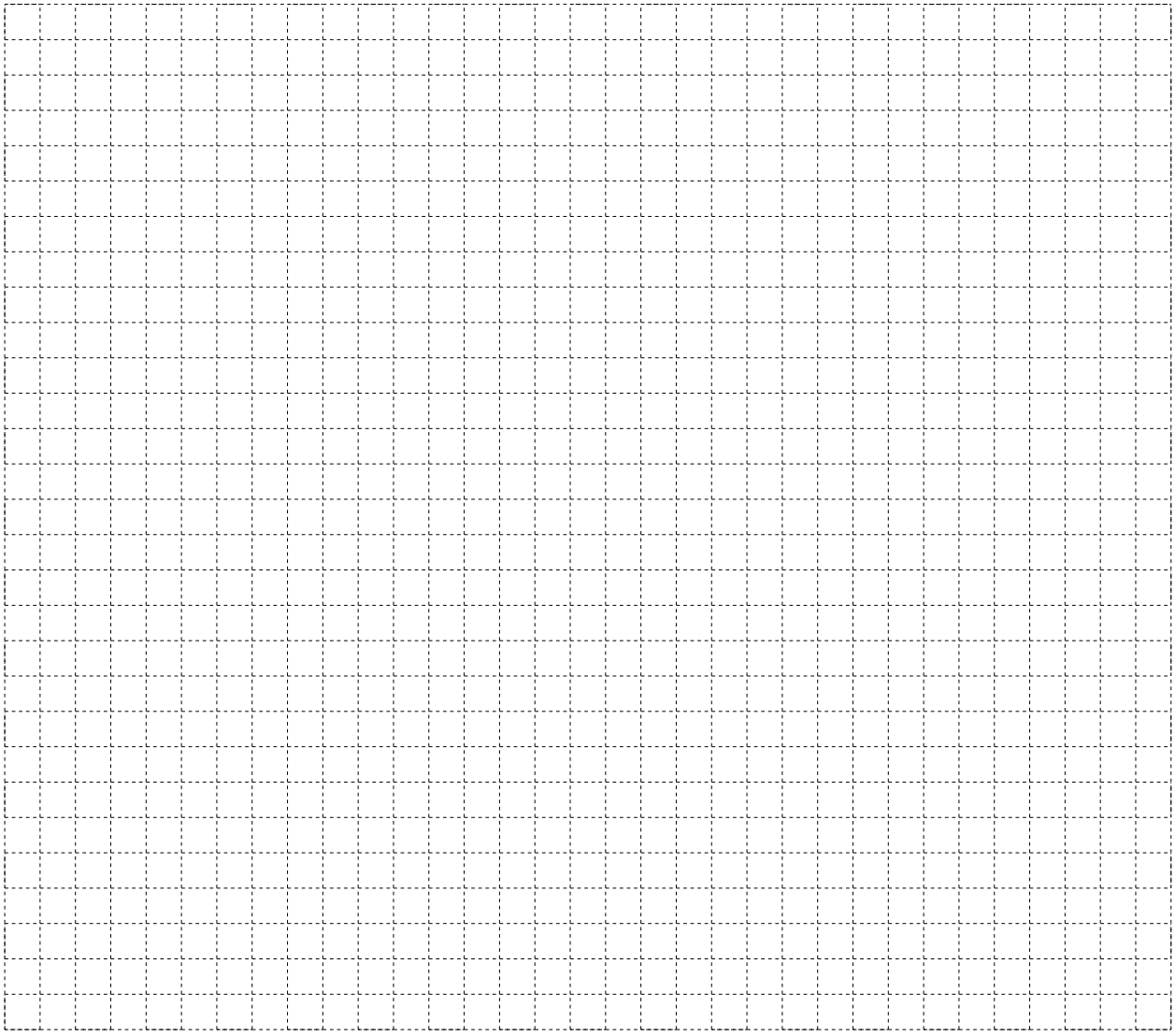
Aufgabe 7 (8 Punkte)

Zeichnen Sie den 4-Bit-Carry-Ripple-Addierer CR_4 über der Bibliothek $BIB = \{and_2, or_2, xor_2, not, mux_2\} \cup \{0, 1\}$ mit $mux_2 : \mathbb{B}^3 \Rightarrow \mathbb{B}$, $mux_2(s, a, b) = \begin{cases} a, & \text{falls } s = 1 \\ b, & \text{falls} s = 0 \end{cases}$; verwenden Sie dabei keine hierarchischen Teilschaltkreise.

Kennzeichnen Sie den längsten Pfad in ihrem Schaltkreis und geben Sie dessen Tiefe an.

Bestimmen Sie für jedes Gatter des ermittelten Schaltkreises den Wert des Gatterausganges für die Belegung
$b_3 = 1$, $b_2 = 1$, $b_1 = 0$, $b_0 = 1$, $a_3 = 0$, $a_2 = 1$, $a_1 = 0$, $a_0 = 1$, $c_{-1} = 1$.

Ihre Lösung zu Aufgabe 7:



Ihre Lösung zu Aufgabe 7 (Fortsetzung):

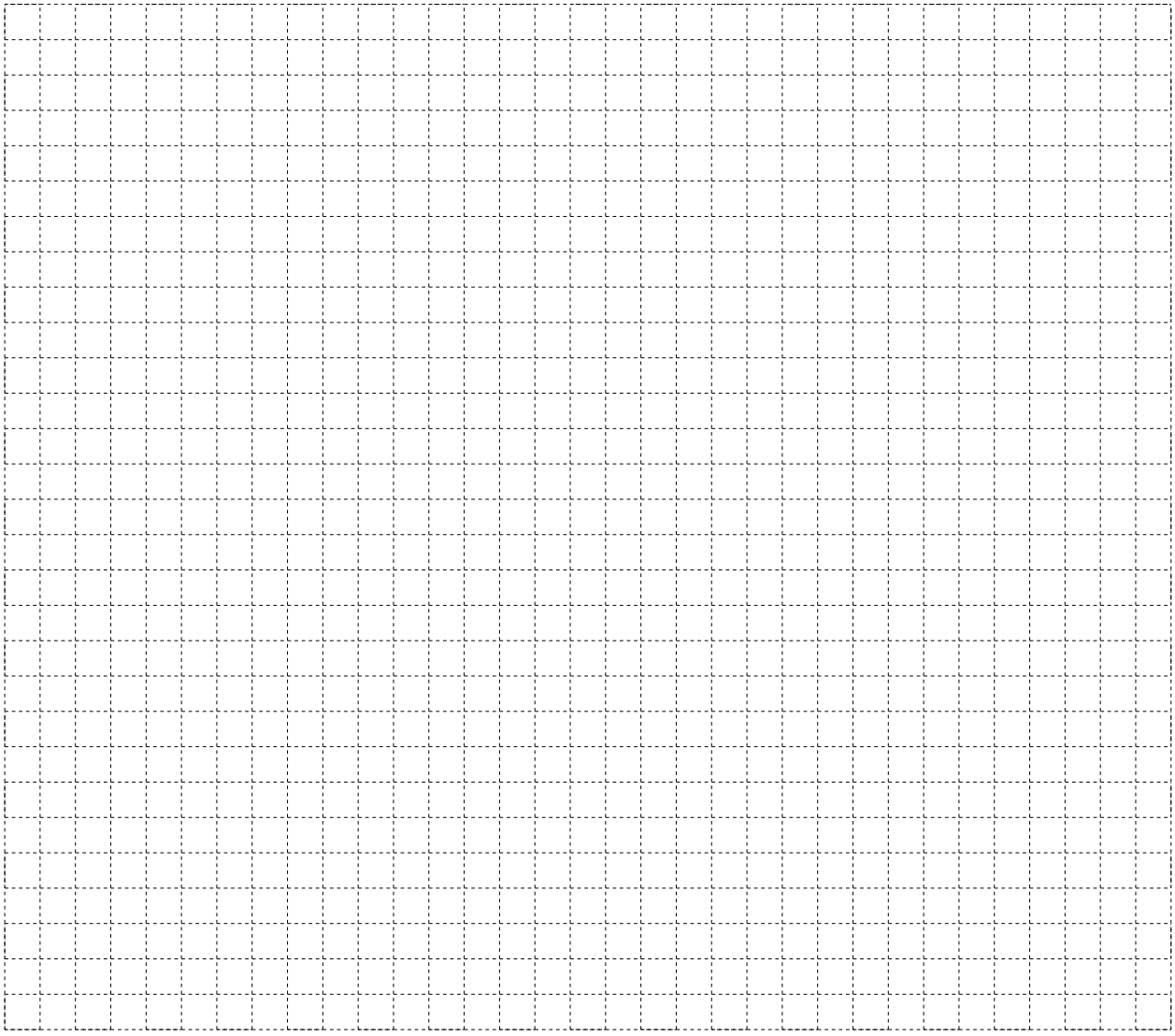
Aufgabe 8 (8 Punkte)

Konstruieren Sie einen Mealy-Automaten , der eine Binärzahl inkrementiert. Gehen Sie dabei davon aus, dass die Binärzahl mit dem niederwertigsten Bit zuerst gelesen wird (d. h. von rechts nach links). Links vom höchstwertigen Bit ist eine Markierung (#) angebracht, die das Ende der Eingabe kennzeichnet. Sie soll nach dem Inkrementieren wieder als letztes Zeichen ausgegeben werden.

Beispiel: Aus #00111 soll #01000 werden.

Zeichnen Sie das Zustandsdiagramm.

Ihre Lösung zu Aufgabe 8:



Ihre Lösung zu Aufgabe 8 (Fortsetzung):

Aufgabe 9 (3 + 5 Punkte)

In Abbildung 3 ist das idealisierte Timing-Diagramm für einen Rechner mit verkürzter *Fetch*- und *Execute*-Phase angegeben. Die Kontrollogik des Rechners wird realisiert wie in Abb. 2 angegeben. Es sollen die Clock-Enable-Signale für die Register I und IN1 erzeugt werden.

Geben Sie die Boolsche Ausdrücke für folgende Signale der Kontrollogik an:

a) Clock-Enable-Signal $Icken_{pre}$ zum Speichern neuer Befehle im Instruktionsregister I.

b) Clock-Enable-Signal $IN1cken_{pre}$ zum Speichern neuer Daten im 1. Indexregister $IN1$.

Hinweis: Gehen Sie analog zur Vorlesung vor, d.h. es wird eine Kontrollogik verwendet, wie sie in der Vorlesung vorgestellt wurde. Beachten Sie bei welchen Befehlen des RE-TI Rechners die obigen Signale jeweils aktiv werden. Die Befehlsübersicht incl. Kodierung der Register befindet sich am Ende der Klausur in Tabelle 1 und 2.

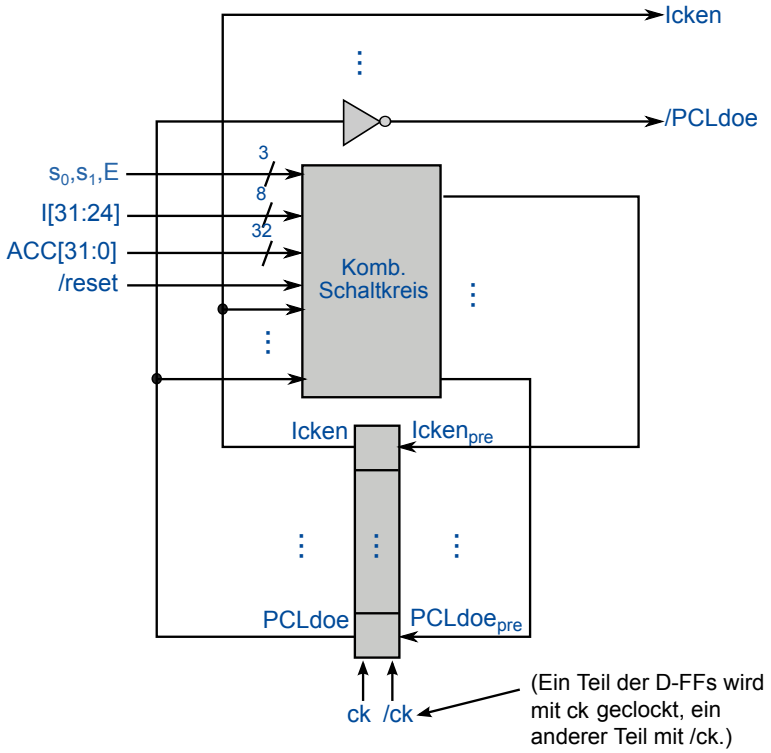


Abbildung 2: Kontrollogik

Ihre Lösung zu Aufgabe 9 (Fortsetzung):

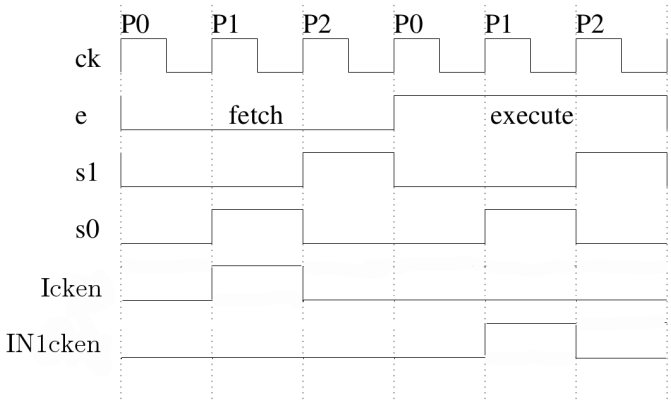
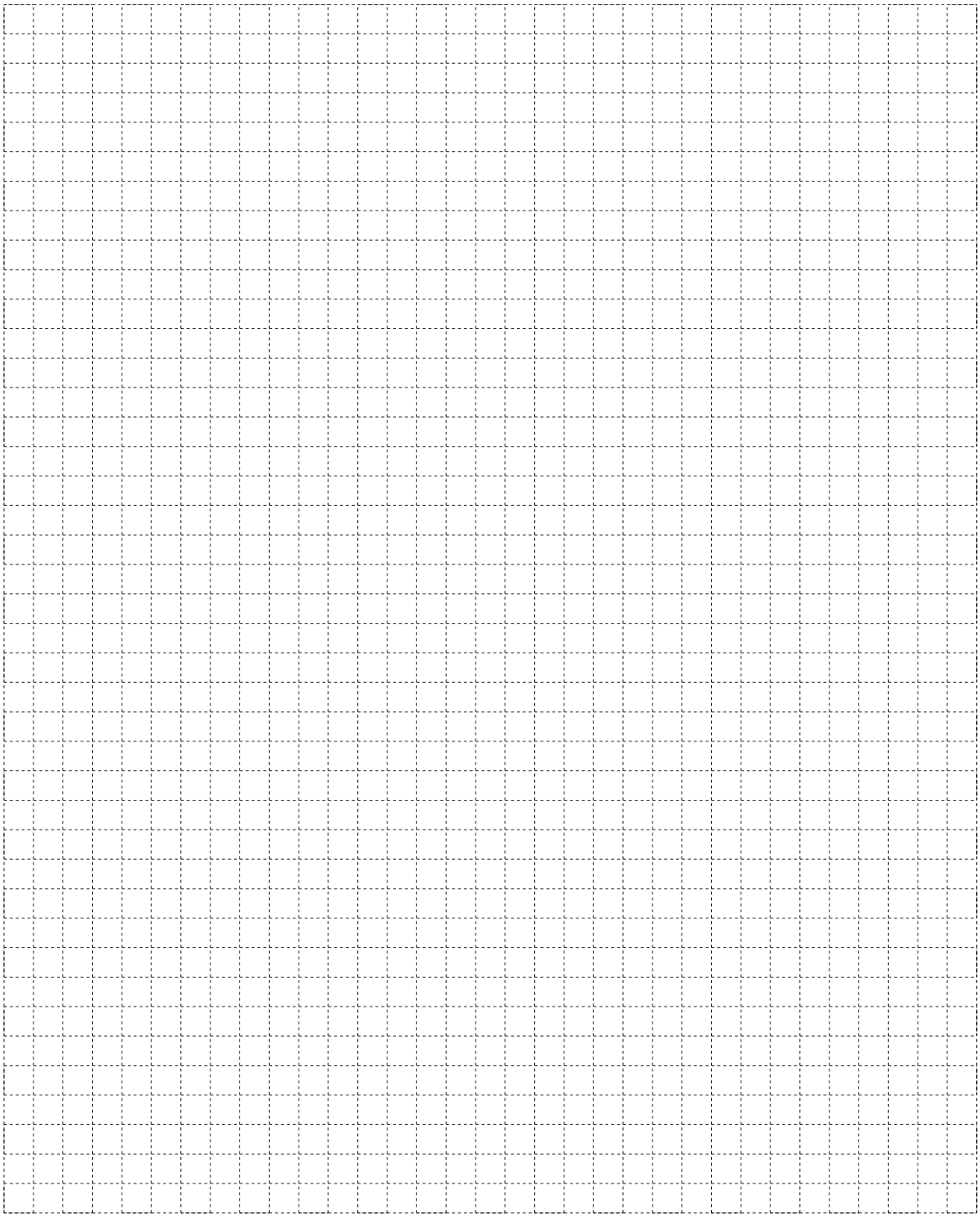


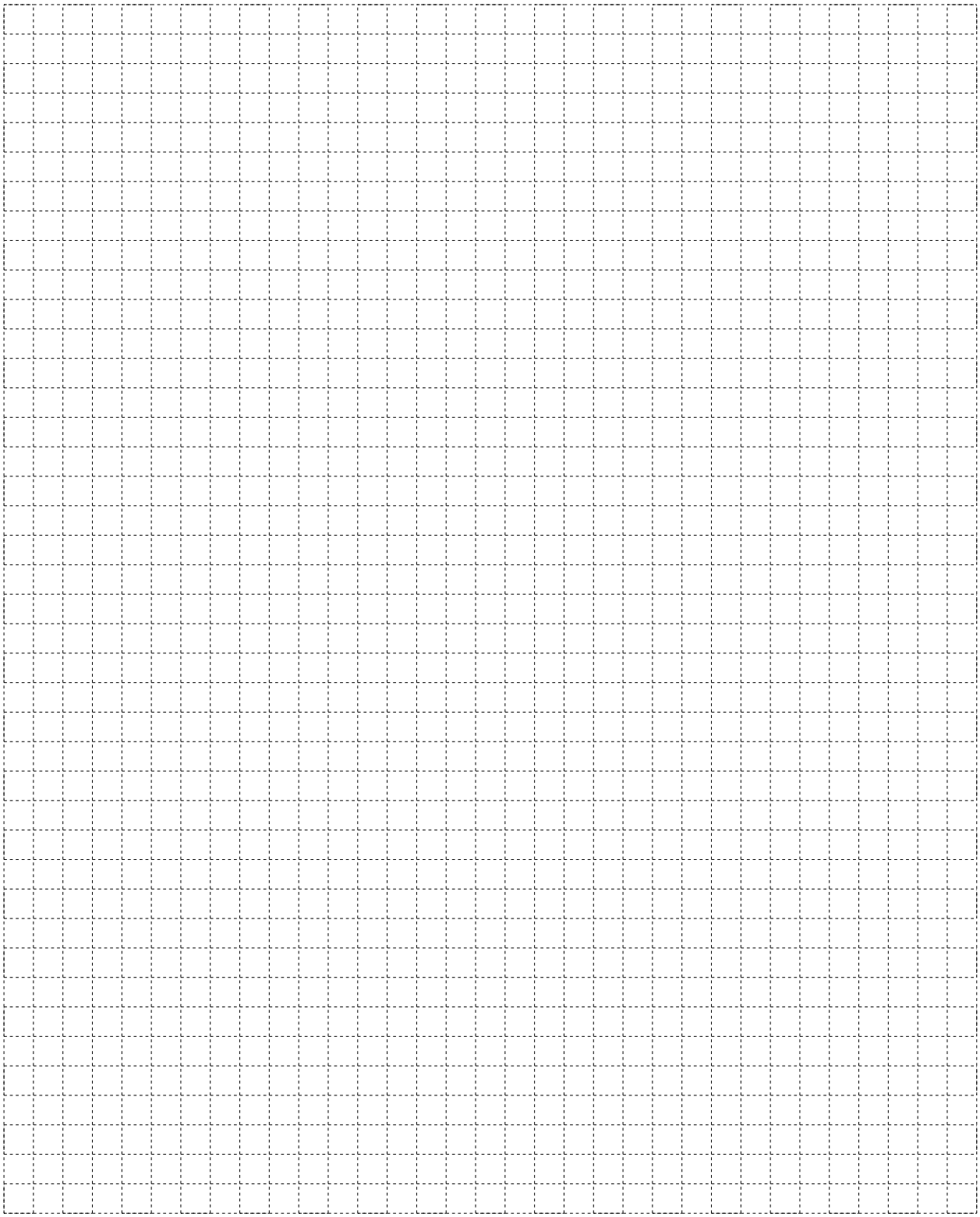
Abbildung 3: Idealisiertes Timing-Diagramm

Ihre Lösung zu Aufgabe 9 (Fortsetzung):



Zusatzseiten für alle Aufgaben – Seite 1

Zusatzseiten für alle Aufgaben – Seite 2



Load Befehle $I[25,24] = D$			
$I[31,28]$	Befehl	Wirkung	
0100	LOAD D i	$D := M(\langle i \rangle)$	
0101	LOADIN1 D i	$D := M(\langle IN1 \rangle + [i])$	$\langle PC \rangle := \langle PC \rangle + 1$, falls $D \neq PC$
0110	LOADIN2 D i	$D := M(\langle IN2 \rangle + [i])$	
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	$M(\langle i \rangle) := ACC$	
1001	STOREIN1 i	$M(\langle IN1 \rangle + [i]) := ACC$	$\langle PC \rangle := \langle PC \rangle + 1$
1010	STOREIN2 i	$M(\langle IN2 \rangle + [i]) := ACC$	
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	$[D] := [D] - [i]$	
000011	ADDI D i	$[D] := [D] + [i]$	
000100	OPLUSI D i	$D := D \oplus 0^8 i$	$\langle PC \rangle := \langle PC \rangle + 1$, falls $D \neq PC$
000101	ORI D i	$D := D \vee 0^8 i$	
000110	ANDI D i	$D := D \wedge 0^8 i$	
001010	SUB D i	$[D] := [D] - [M(\langle i \rangle)]$	
001011	ADD D i	$[D] := [D] + [M(\langle i \rangle)]$	
001100	OPLUS D i	$D := D \oplus M(\langle i \rangle)$	$\langle PC \rangle := \langle PC \rangle + 1$, falls $D \neq PC$
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	$\langle PC \rangle := \begin{cases} \langle PC \rangle + [i], & \text{falls } [ACC]\ c\ 0 \quad (c \in \{<, \leq, =, \geq, >\}) \\ \langle PC \rangle + 1 & \text{sonst} \end{cases}$
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 1: Befehlstabelle der ReTI

S, D	**Register**
0 0	PC
0 1	$IN1$
1 0	$IN2$
1 1	ACC

Tabelle 2: Kodierung S,D von ReTI