



## **Prüfungsklausur Digitaltechnik und Rechnersysteme**

**für die Bachelor-Prüfung im Wintersemester 2024/2025**

**Bearbeitungszeit:** 90 Minuten  
**Maximalpunktzahl:** 90

### **Zugelassene Hilfsmittel**

- Handgeschriebene Formelsammlung max. ein DIN A4 Blatt, beidseitig
- Referenzblatt »MIPS32 Befehlssatz (Auszug) und Register« (im Anhang der Klausur)
- nicht-programmierbarer Taschenrechner

### **Hinweise für alle Aufgaben**

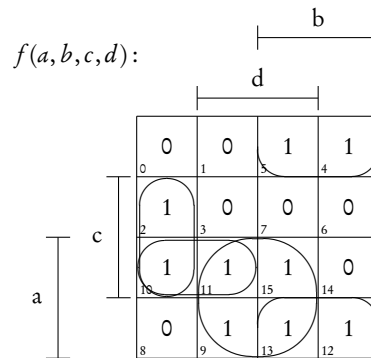
- Tragen Sie zunächst Ihren Namen und Ihre Matrikelnummer auf dem Antwortbogen ein. Kreuzen Sie die Ziffer Ihrer Matrikelnummer an.
- Führen Sie alle Rechnungen ausschließlich auf dem dafür vorgesehenen Antwortbogen aus. Falls dir Platz nicht ausreicht gibt es am Ende des Antwortbogens ein Zusatzblatt. Auf dieses dann in der Aufgabe entsprechend verweisen.
- Lösungen ohne nachvollziehbarem Lösungsweg werden mit null Punkten bewertet.
- Antworten/Lösungen können nur bewertet werden, wenn sie mit dokumentenechten, dunklen Stiften (Kugelschreiber, Füller) geschrieben sind (**keine Farben!**).
- Schreiben Sie leserlich.
- Nach 85 Minuten verlässt niemand mehr den Platz.
- Klausur und Antwortbogen werden am Ende eingesammelt.

### Aufgabe 1: Verständnisfragen (12 Punkte)

Alle Aufgabenteile können unabhängig voneinander gelöst werden!

1.1 Wie lautet der Dezimalwert der binären Festkommazahl  $101,101_2$ ? (1 Punkt)

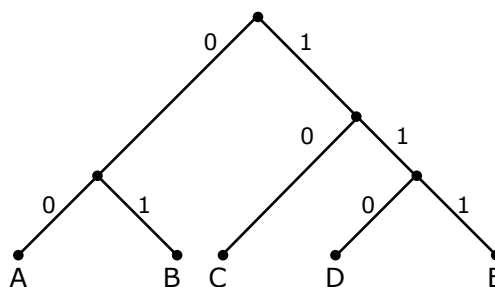
1.2 Gegeben ist folgendes KV-Diagramm: (6 Punkte)



Bestimmen Sie den Typ der eingekringelten Primimplikanten (PIs) und kreuzen Sie diesen in der Tabelle an (KPI: Kernprimimplikant, API: Absolut eliminierbarer Primimplikant, REPI: Relativ eliminierbarer Primimplikant).

Indizes Primimplikant	KPI	API	REPI
2,10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10,11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11,15,9,13	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
12,13	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4,12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4,5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

1.3 Gegeben ist der Codierungsbaum eines Codes mit variabler Länge (ohne Wahrscheinlichkeiten):



Geben Sie die binäre Nachricht für die Symbolfolge EBBCAD an. (2 Punkte)

1.4 Gegeben sind die Zahlen  $98_{10} = 01100010_2$  und  $69_{10} = 01000101_2$ . Führen Sie die Rechnung  $98 - 69$  im Zweierkomplement mit einer Wortbreite von 8 Bit durch. (3 Punkte)

## Aufgabe 2: Boolesche Algebra (12 Punkte)

2.1 – 2.10 Welche der gezeigten Booleschen Gleichungen auf dem Lösungsblatt sind korrekt (Mehrfachauswahl möglich)? Kreuzen Sie die Lösung auf dem Lösungsblatt an. (10 Punkte)

Achtung: Falsch gesetzte Kreuze führen zu Punktabzug! Kreuzen Sie nur Antworten an bei denen Sie sich sicher sind!

- |                          |   |
|--------------------------|---|
| <input type="checkbox"/> | $\overline{ab + \bar{c}\bar{d}} = \overline{ab} \cdot cd$     |
| <input type="checkbox"/> | $\overline{ab + \bar{c}\bar{d}} = \overline{ab}(c + d)$       |
| <input type="checkbox"/> | $\overline{ab + \bar{c}\bar{d}} = \overline{ab}c + d$         |
| <input type="checkbox"/> | $\overline{ab + \bar{c}\bar{d}} = (a + b)(c + d)$             |
| <input type="checkbox"/> | $\overline{ab + \bar{c}\bar{d}} = (\bar{a} + \bar{b})(c + d)$ |

- |                          |  |
|--------------------------|--|
| <input type="checkbox"/> | $\bar{a}bc + cd + \bar{a}b + \bar{c}\bar{d} = \bar{a}bc + \bar{a}b$                        |
| <input type="checkbox"/> | $\bar{a}bc + cd + \bar{a}b + \bar{c}\bar{d} = cd + \bar{a}b + \bar{c}\bar{d}$              |
| <input type="checkbox"/> | $\bar{a}bc + cd + \bar{a}b + \bar{c}\bar{d} = \bar{a}bc + \bar{a}b + \bar{c}\bar{d}$       |
| <input type="checkbox"/> | $\bar{a}bc + cd + \bar{a}b + \bar{c}\bar{d} = c(\bar{a}b + d) + \bar{a}b + \bar{c}\bar{d}$ |
| <input type="checkbox"/> | $\bar{a}bc + cd + \bar{a}b + \bar{c}\bar{d} = \bar{a}bc + cd + \bar{a}b$                   |

2.11 Vereinfachen Sie folgenden Booleschen Ausdruck (und retten Sie damit Ihren Freund Alex!) (2 Punkte):

$$BA(CD + \bar{C}\bar{A}\bar{B} + CABD)(\bar{A} + \bar{B})$$

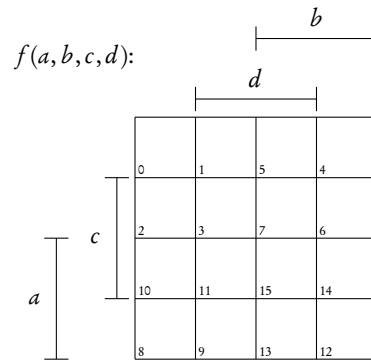
## Aufgabe 3: KV-Minimierung (14 Punkte)

Gegeben ist die Funktion

$$f(a, b, c, d) = \bar{a}\bar{b}\bar{c}\bar{d} + \bar{a}\bar{b}cd + \bar{a}b\bar{c}\bar{d} + a\bar{b}\bar{c}\bar{d} + \bar{a}bcd + a\bar{b}cd + ab\bar{c}\bar{d} + abcd$$

3.1 Ermitteln Sie die Wahrheitstabelle für  $f(a, b, c, d)$  und minimieren Sie die Funktion mit Hilfe des gegebenen KV-Diagramms. Geben Sie die Funktion sowohl in minimaler **Disjunktiver Normalform (DNF)** als auch in minimaler **Konjunktiver Normalform (KNF)** an. (10 Punkte)

$a$	$b$	$c$	$d$	$f$
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	



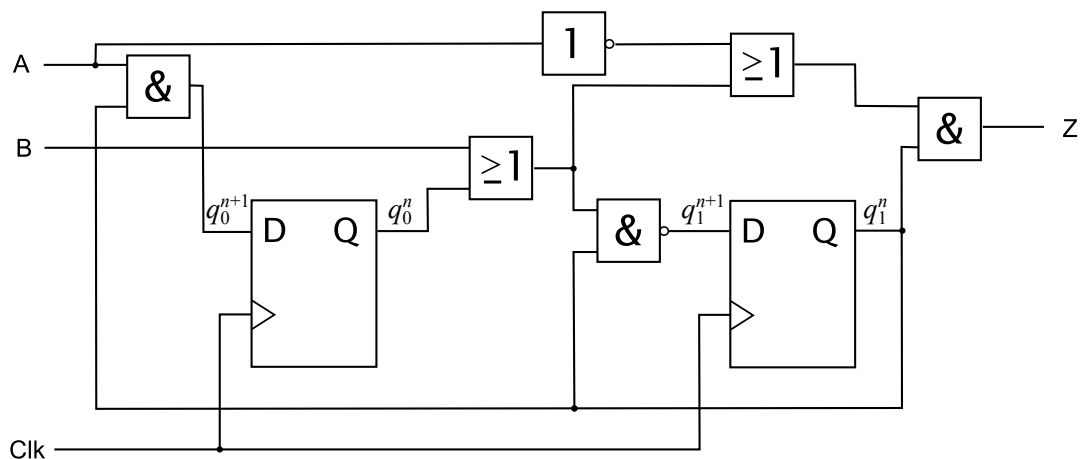
3.2 Welchen abgeleiteten Operator realisiert die Schaltung? Geben Sie den Namen des Operators bzw. Gatters an. (2 Punkte)

3.3 Zeichnen Sie das Schaltbild der DNF. (2 Punkte)

#### Aufgabe 4: Automaten-Analyse synchroner Automat (21 Punkte)

Die Aufgabenteile 4.6 und 4.7 sind unabhängig von den Teilen 4.1 bis 4.5 lösbar!

Gegeben ist folgende Schaltung eines Automaten:



4.1 Geben Sie die Zustandsübergangsfunktionen  $q_0^{n+1}$  und  $q_1^{n+1}$  sowie die Ausgangsfunktion  $z$  in Abhängigkeit der Eingangsvariablen  $A$  und  $B$  sowie aktuellen Zustände  $q_0^n$  und  $q_1^n$  in Disjunktiver Normalform (DNF) an. (3 Punkte)

4.2 Geben Sie die Zustandsübergangs- und Ausgangstabelle an. (3 Punkte)

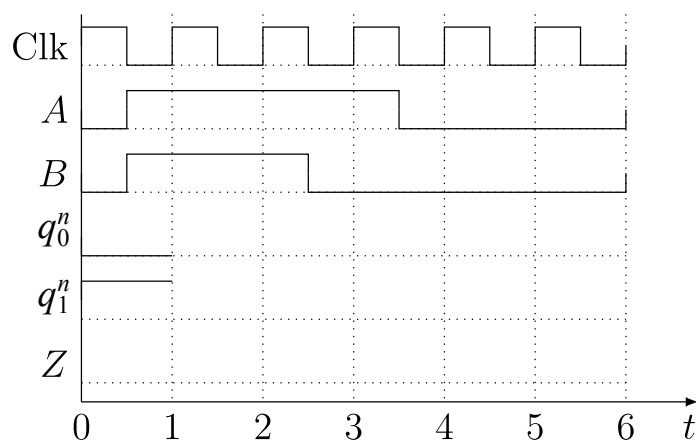
$q_1^n$	$q_0^n$	$A$	$B$	$q_1^{n+1}$	$q_0^{n+1}$	$z$
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1			
1	1	1	0			
1	1	1	1			

4.3 Welche der gegebenen Aussagen trifft zu? (1 Punkt)

- ☐ Es handelt sich um einen Moore-Automat, da der Zustand von der Eingabe abhängt.
- ☐ Es handelt sich um einen Moore-Automat, da die Ausgabe von der Eingabe abhängt.
- ☐ Es handelt sich um einen Mealy-Automat, da der Zustand von der Eingabe abhängt.
- ☐ Es handelt sich um einen Mealy-Automat, da die Ausgabe von der Eingabe abhängt.

4.4 Zeichnen Sie das Zustandsübergangsdiagramm des Automaten und geben Sie die verwendete Notation an. (6 Punkte)

4.5 Ergänzen Sie das Timing-Diagramm für den Automaten, welcher sich zum Zeitpunkt  $t = 0$  im Zustand  $q_0^n = 0$ ,  $q_1^n = 1$  befindet. (2 Punkte)



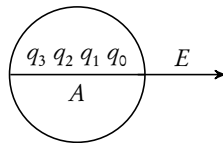
4.6 Alle Gatter haben je eine Durchlaufzeit von 5 ns. Bestimmen Sie die maximalen Durchlaufzeiten durch die kombinatorischen Schaltungsteile für  $q_0^{n+1}$ ,  $q_0^{n+1}$  und  $z$  und tragen Sie die Zeiten in das Schaltbild ein. Ermitteln Sie den (die) kritischen Pfad(e) und zeichnen Sie diese deutlich in die Schaltung ein. Wie groß ist die maximale Durchlaufzeit ( $t_{PD,max}$ )? (4 Punkte)

4.7 Wie hoch ist die maximale Taktfrequenz  $f_{max}$  mit der die Gesamtschaltung fehlerfrei betrieben werden kann wenn auch die Ein- und Ausgänge taktsynchron verarbeitet werden? Alle D-Flipflops haben hierbei eine Durchlaufzeit von 7 ns und eine Setup-Zeit von 3 ns. (2 Punkte)

### Aufgabe 5: Automatenentwurf (12 Punkte)

Es soll ein synchroner Countdown-Zähler entwickelt werden der von  $10_{10}$  bis  $0_{10}$  rückwärts zählt. Bei Eingabe  $E = 0$  steht der Zähler auf 10, bei Eingabe  $E = 1$  zählt dieser mit jedem Takt um eins nach unten. Geht der Eingang während des Zählvorgangs auf  $E = 0$  wird der Zählvorgang abgebrochen und der Zähler wird auf 10 zurückgesetzt. Erreicht der Zähler den Wert 0, soll der Ausgang  $A$  auf 1 gesetzt werden solange die Eingabe  $E = 1$  ist, ansonsten soll der Ausgang immer 0 sein. Die Zustandskodierung ( $q_3 q_2 q_1 q_0$ ) soll der binär codierten Zahl entsprechen.

5.1 Zeichnen Sie das Zustandsdiagramm. Verwenden Sie die nachfolgende Notation.



### Aufgabe 6: Minimal-Computer (9 Punkte)

Gegeben ist der aus der Vorlesung bekannte erweiterte Minimal-Computer welcher die folgenden Operationen unterstützt:

Operation	Bedeutung
add $r_d, r_s, r_t$	$r_d \leftarrow r_s + r_t$
sub $r_d, r_s, r_t$	$r_d \leftarrow r_s - r_t$
mul $r_d, r_s, r_t$	$r_d \leftarrow r_s \times r_t$
and $r_d, r_s, r_t$	$r_d \leftarrow r_s \wedge r_t$
or $r_d, r_s, r_t$	$r_d \leftarrow r_s \vee r_t$
xor $r_d, r_s, r_t$	$r_d \leftarrow r_s \oplus r_t$
ldi $r_d, i$	$r_d \leftarrow i$
jmp $i$	$PC \leftarrow i$

Gegeben ist folgendes Assembler-Programm (links steht Programm-Adresse, rechts das dort gespeicherte Befehlswort):

```

0: ldi r0,1
1: ldi r1,1
2: ldi r2,1
3: mul r0,r0,r1
4: add r1,r1,r2
5: jmp 3

```

6.1 Führen Sie eine Simulation des Programms für die ersten 9 Takte durch, in dem Sie die Ergebnisse nach jedem Takt in die Tabelle eintragen.

PC	r0	r1	r2	r3
0	0	0	0	0

**Aufgabe 7: MIPS (10 Punkte)**

7.1 Welche Bedeutung hat der folgende Binärcode wenn dieser als MIPS-Befehl interpretiert wird. (2 Punkte)

00000000 00100010 00011000 00100010

7.2 Schreiben Sie ein Programm in MIPS Assembler welches den algebraischen Ausdruck

$$y = (x + 1)(x + 2)(x - 1)$$

berechnet. Gehen Sie davon aus, dass die Variable  $x$  in Register \$1 gespeichert ist und das Ergebnis  $y$  in Register \$2 gespeichert werden soll. (8 Punkte)

## MIPS32 Befehlssatz (Auszug) und Register

Befehlssatz des MIPS32 (Auszug)				
Befehl	Format	opcode/funct <sup>1</sup>	Beispiel	Bedeutung
<i>Arithmetic Instructions</i>				
Add	R	00/32	<code>add \$1,\$2,\$3</code>	$\$1 \leftarrow \$2 + \$3$
Subtract	R	00/34	<code>sub \$1,\$2,\$3</code>	$\$1 \leftarrow \$2 - \$3$
Multiply	R	16/02	<code>mul \$1,\$2,\$3</code>	$\$1 \leftarrow \$2 \times \$3$
Add Immediate	I	08/-	<code>addi \$1,\$2,20</code>	$\$1 \leftarrow \$2 + 20$
<i>Logical Instructions</i>				
AND	R	00/36	<code>and \$1,\$2,\$3</code>	$\$1 \leftarrow \$2 \wedge \$3$
OR	R	00/37	<code>or \$1,\$2,\$3</code>	$\$1 \leftarrow \$2 \vee \$3$
XOR	R	00/38	<code>xor \$1,\$2,\$3</code>	$\$1 \leftarrow \$2 \oplus \$3$
NOR	R	00/39	<code>nor \$1,\$2,\$3</code>	$\$1 \leftarrow \overline{\$2 \vee \$3}$
AND Immediate	I	12/-	<code>andi \$1,\$2,20</code>	$\$1 \leftarrow \$2 \wedge 20$
OR Immediate	I	13/-	<code>ori \$1,\$2,20</code>	$\$1 \leftarrow \$2 \vee 20$
XOR Immediate	I	14/-	<code>xori \$1,\$2,20</code>	$\$1 \leftarrow \$2 \oplus 20$
Shift Left Logical	R	00/00	<code>sll \$1,\$2,10</code>	$\$1 \leftarrow \$2 \ll 10$
Shift Right Logical	R	00/02	<code>srl \$1,\$2,10</code>	$\$1 \leftarrow \$2 \gg 10$
<i>Load/Store Instructions</i>				
Load Word	I	35/-	<code>lw \$1,20(\$2)</code>	$\$1 \leftarrow \text{Memory}[\$2 + 20]$
Store Word	I	43/-	<code>sw \$1,20(\$2)</code>	$\text{Memory}[\$2 + 20] \leftarrow \$1$
Load Half	I	33/-	<code>lh \$1,20(\$2)</code>	$(\$1 \leftarrow \text{Memory}[\$2 + 20])_{15:0}$
Store Half	I	41/-	<code>sh \$1,20(\$2)</code>	$(\text{Memory}[\$2 + 20] \leftarrow \$1)_{15:0}$
Load Byte	I	32/-	<code>lb \$1,20(\$2)</code>	$(\$1 \leftarrow \text{Memory}[\$2 + 20])_{7:0}$
Store Byte	I	40/-	<code>sb \$1,20(\$2)</code>	$(\text{Memory}[\$2 + 20] \leftarrow \$1)_{7:0}$
Load Upper Immediate	I	15/-	<code>lui \$1,0x1234</code>	$\$1 \leftarrow 0x1234 \ll 16$
<i>Jump Instructions</i>				
Jump	J	02/-	<code>j 2500</code>	$\text{PC} \leftarrow \text{PC} + 10000$
Jump Register	R	00/08	<code>jr \$1</code>	$\text{PC} \leftarrow \$1$
<i>Branch Instructions</i>				
Branch on Equal	I	04/-	<code>beq \$1,\$2,25</code>	if $(\$1 \equiv \$2)$ $\text{PC} \leftarrow \text{PC} + 100$
Branch on Not Equal	I	05/-	<code>bne \$1,\$2,25</code>	if $(\$1 \neq \$2)$ $\text{PC} \leftarrow \text{PC} + 100$
Branch on Less Than or Equal to Zero	I	06/-	<code>blez \$1,25</code>	if $(\$1 \leq 0)$ $\text{PC} \leftarrow \text{PC} + 100$
Branch on Greater Than Zero	I	07/-	<code>bgtz \$1,25</code>	if $(\$1 > 0)$ $\text{PC} \leftarrow \text{PC} + 100$
Set on Less Than	R	00/42	<code>slt \$1,\$2,\$3</code>	if $(\$2 < \$3)$ $\$1 \leftarrow 1$ else $\$1 \leftarrow 0$
Set on Less Than Immediate	I	10/-	<code>slti \$1,\$2,20</code>	if $(\$2 < 20)$ $\$1 \leftarrow 1$ else $\$1 \leftarrow 0$
<i>Other Instructions</i>				
No Operation	R	00/00	<code>nop</code>	do nothing

<sup>1</sup> Dezimaldarstellung opcode / funct (bei R-Befehlstypen)

### Befehlsformate des MIPS32

R	opcode	rs	rt	rd	shamt	funct
	31	26 25	21 20	16 15	11 10	6 5
I	opcode	rs	rt	immediate		
	31	26 25	21 20	16 15		
J	opcode	address				
	31	26 25				

- **opcode:** Befehlscode, bei R-Befehl immer 0
- **rs:** 1. *source register*, 1. Quellregister
- **rt:** 2. *source register*, 2. Quellregister
- **rd:** *destination register*, Zielregister
- **shamt:** *shift amount*, wird nur für Bitverschiebung benötigt
- **funct:** *function code*, spezifiziert Variante der Operation