

Aufgabe : Modulo-12-Zähler

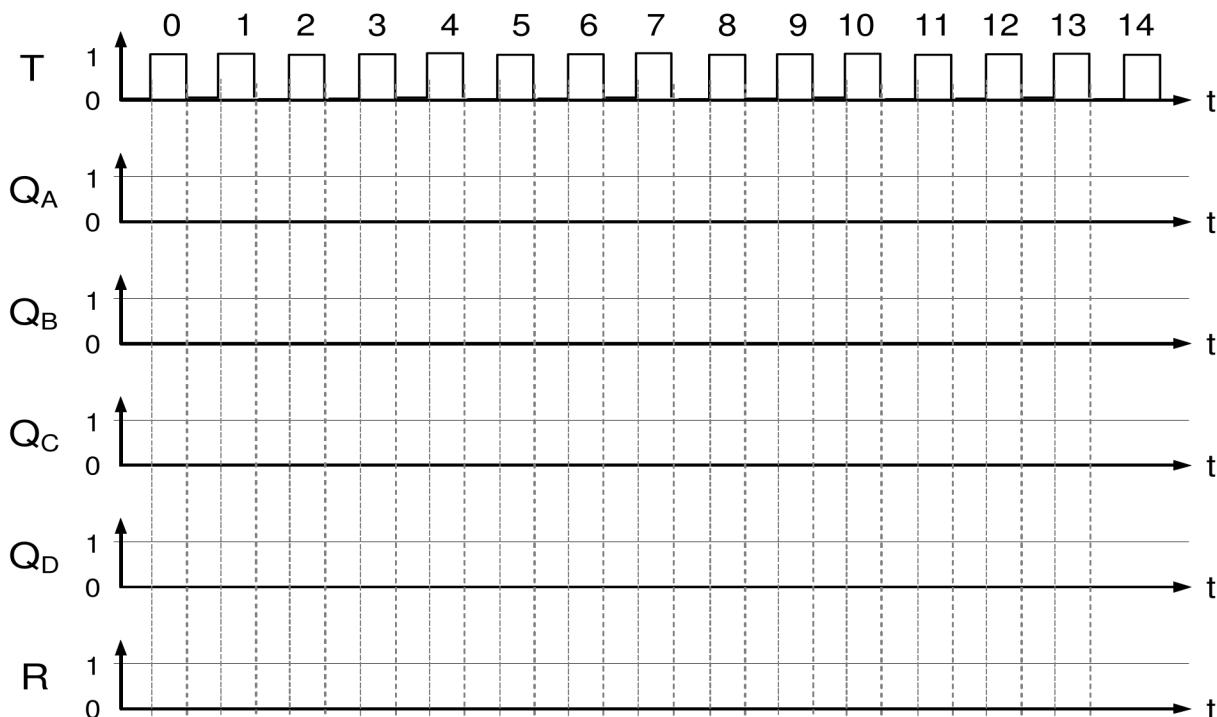
- a) Beschreiben Sie die Zählweise eines Modulo-12-Zählers.
 (Mit welcher Dual-/Dezimalzahl wird der Zähler zurückgesetzt?
 Anzahl der Zählimpulse? Höchster angezeigter Zahlenwert)

.....

- b) Zeichnen Sie die Schaltung eines Modulo-12-Zählers aus Flipflops Ihrer Wahl.



- c) Ergänzen/Zeichnen Sie das Zeitablaufdiagramm zu diesem Zähler.
 (Rückstellung erfolgt mit log.0).

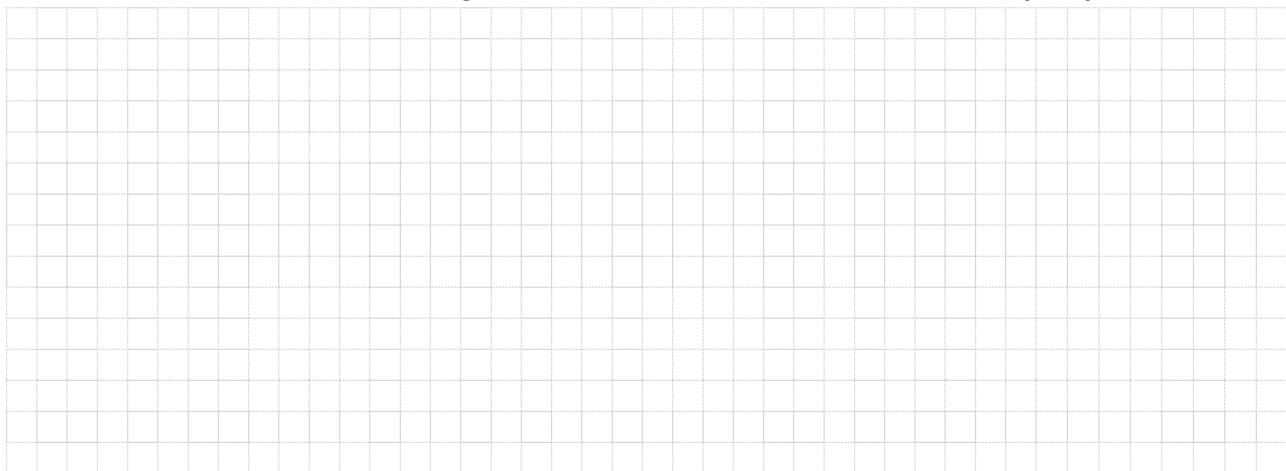


Aufgabe : Modulo-12-Zähler

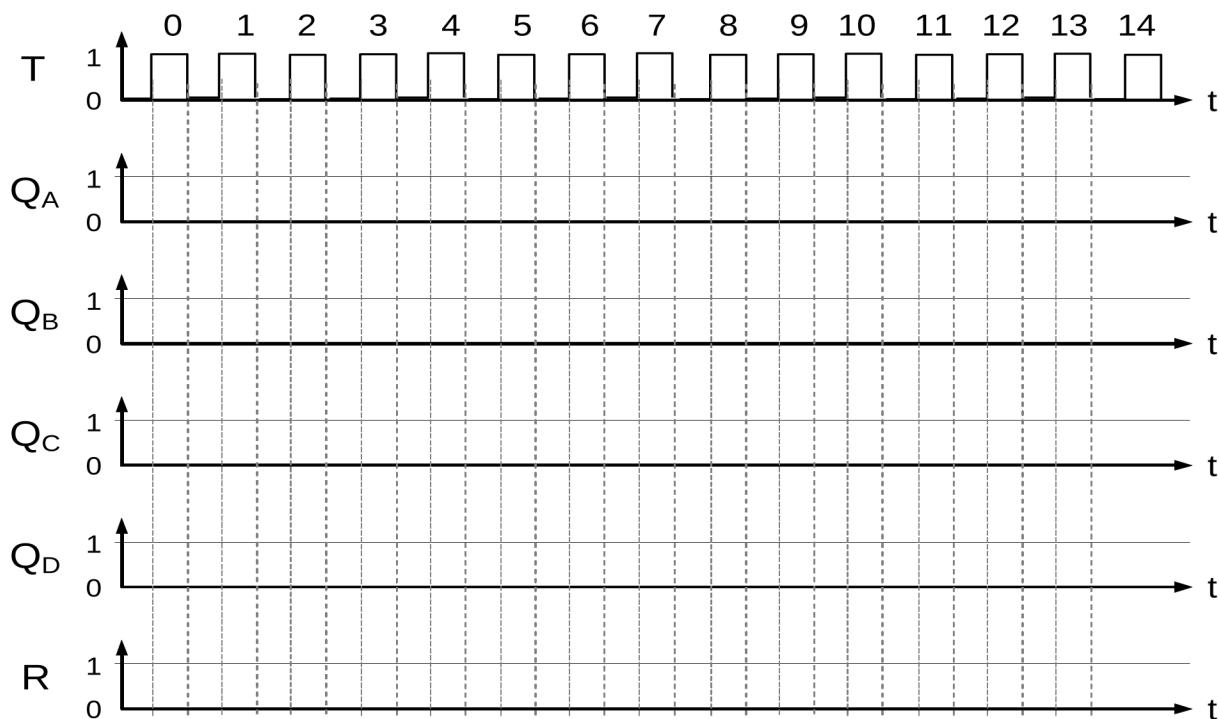
- a) Beschreiben Sie die Zählweise eines Modulo-12-Zählers.
 (Mit welcher Dual-/Dezimalzahl wird der Zähler zurückgesetzt?
 Anzahl der Zählimpulse? Höchster angezeigter Zahlenwert)

.....

- b) Zeichnen Sie die Schaltung eines Modulo-12-Zählers aus Flipflops Ihrer Wahl.



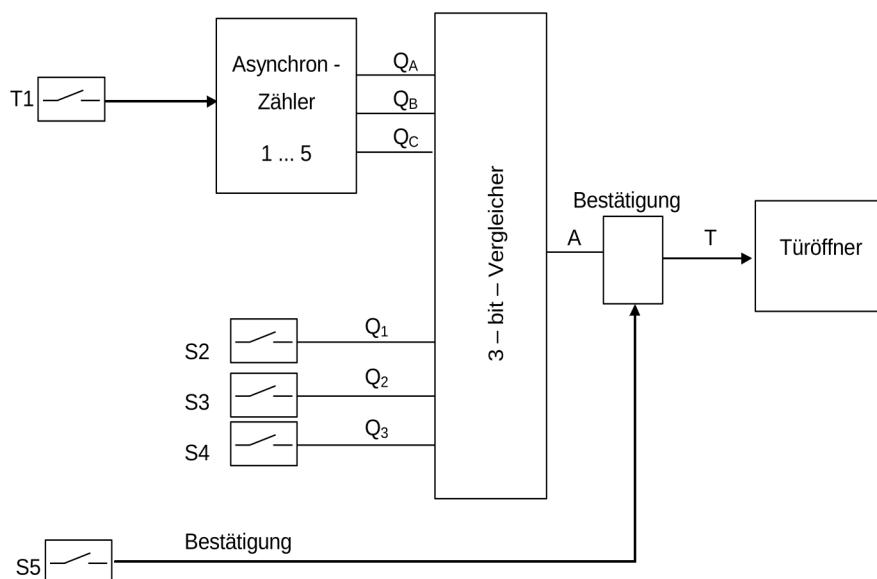
- c) Ergänzen/Zeichnen Sie das Zeitablaufdiagramm zu diesem Zähler.
 (Rückstellung erfolgt mit log.0).



Aufgabe : Türöffner mit Codeschloss (Vereinfachter Auszug)

Das Blockschaltbild zeigt den Aufbau der Anlage. Sie besteht aus folgenden Schaltungsteilen:

- Asynchronzähler zur Voreinstellung der Codenummer
Mit Hilfe des Tasters T1 können mit dem Asynchronzähler die Dezimalzahlen 1... 5 im BCD-Code eingestellt werden.
- Eingabe der Codenummer mit Speicherung
Mit den Schaltern S2 ... S4 wird die Codenummer binär eingegeben.
- 3-bit-Vergleicher mit Bestätigung
Der Vergleicher prüft, ob die eingestellte Dezimalzahl Q_1, Q_2, Q_3 mit der eingegebenen Dezimalzahl Q_A, Q_B, Q_C übereinstimmt.
Mit dem Schalter S5 wird das Ergebnis des Vergleichers weitergegeben, welches dann den Türöffner ansteuert.



1. Entwurf des Asynchronzählers von 1 bis 5

- 1.1 Wie viele Speicherbausteine braucht der Zähler?
- 1.2 Ergänzen Sie auf dem Arbeitsblatt die Schaltung zu einem 1.1 entsprechenden asynchronen Dual-Vorwärtzähler.
- 1.3 Die Schaltung ist nun so zu erweitern, dass der Zähler von "1" bis "5" zählt (ohne Schaltnetz, /S, /R nutzen!).

2. Vergleicher

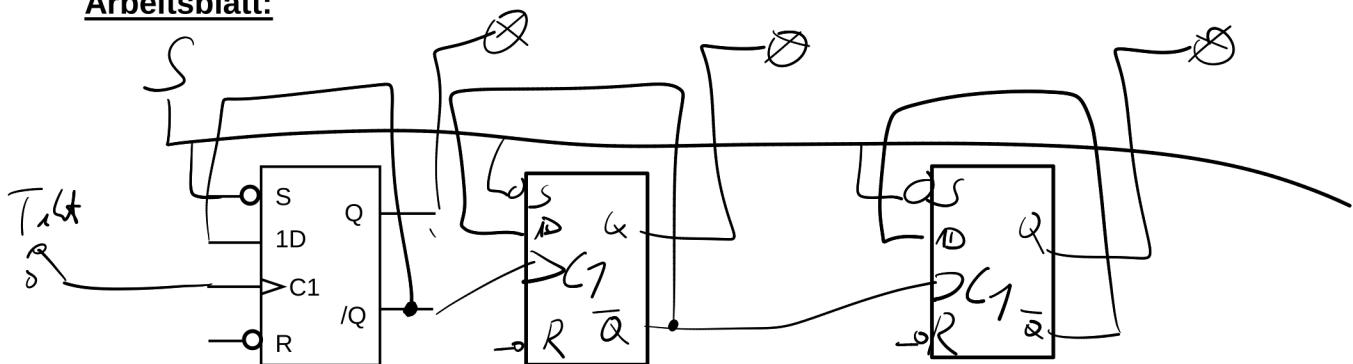
Der Vergleicher soll unter Verwendung von drei Äquivalenz-Gattern (1-Bit-Vergleichern) realisiert werden.

- 2.1 Beschreiben Sie die Funktion einer Äquivalenz. Geben Sie die Funktionsgleichung an.
- 2.2 Ergänzen Sie auf dem Arbeitsblatt (vgl. unten) den Schaltplan des vollständigen 3-bit-Vergleichers.

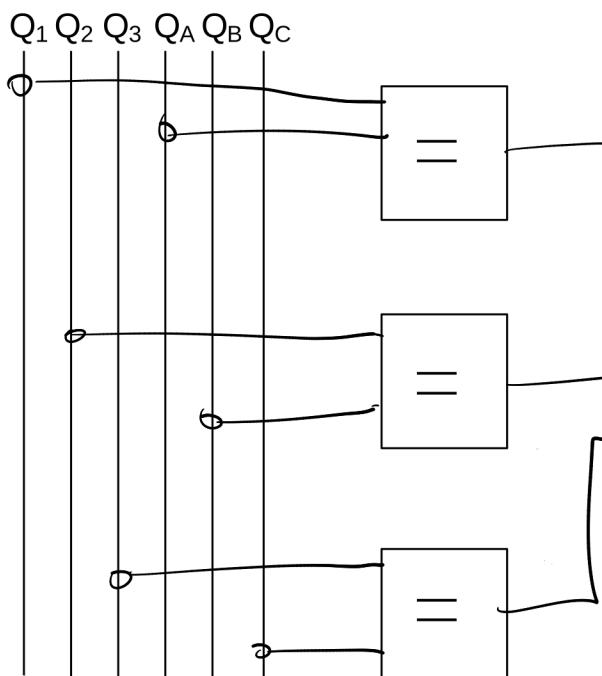
3. Bestätigung

Der Türöffner soll nur dann ein Signal T bekommen, wenn S5 (Bestätigung) kurz betätigt wird. Erweitern Sie auf dem Arbeitsblatt die Schaltung aus 2.2 entsprechend.

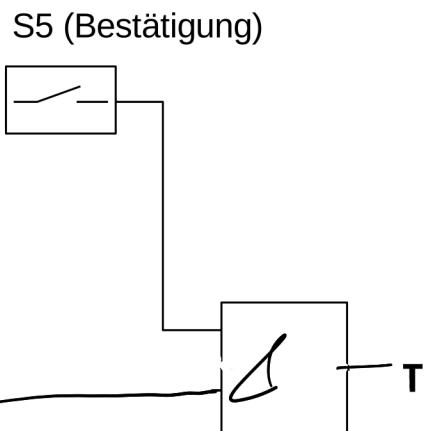
Arbeitsblatt:



Zu 2.2 : 3-bit- Vergleicher



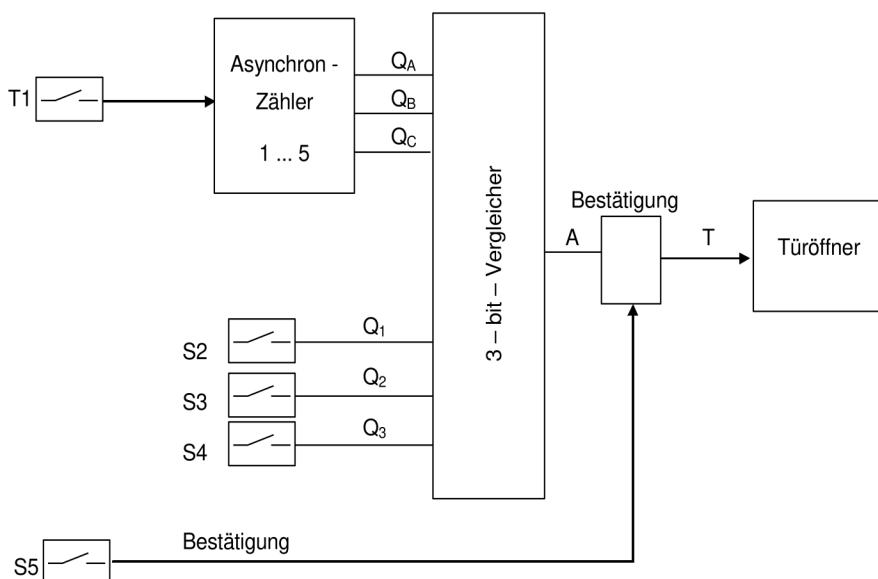
Zu 3 :



Aufgabe : Türöffner mit Codeschloss (Vereinfachter Auszug)

Das Blockschaltbild zeigt den Aufbau der Anlage. Sie besteht aus folgenden Schaltungsteilen:

- Asynchronzähler zur Voreinstellung der Codenummer
Mit Hilfe des Tasters T1 können mit dem Asynchronzähler die Dezimalzahlen 1... 5 im BCD-Code eingestellt werden.
- Eingabe der Codenummer mit Speicherung
Mit den Schaltern S2 ... S4 wird die Codenummer binär eingegeben.
- 3-bit-Vergleicher mit Bestätigung
Der Vergleicher prüft, ob die eingestellte Dezimalzahl Q_1, Q_2, Q_3 mit der eingegebenen Dezimalzahl Q_A, Q_B, Q_C übereinstimmt.
Mit dem Schalter S5 wird das Ergebnis des Vergleichers weitergegeben, welches dann den Türöffner ansteuert.



1. Entwurf des Asynchronzählers von 1 bis 5

- 1.1 Wie viele Speicherbausteine braucht der Zähler?
- 1.2 Ergänzen Sie auf dem Arbeitsblatt die Schaltung zu einem 1.1 entsprechenden asynchronen Dual-Vorwärtzähler.
- 1.3 Die Schaltung ist nun so zu erweitern, dass der Zähler von "1" bis "5" zählt (ohne Schaltnetz, /S, /R nutzen!).

2. Vergleicher

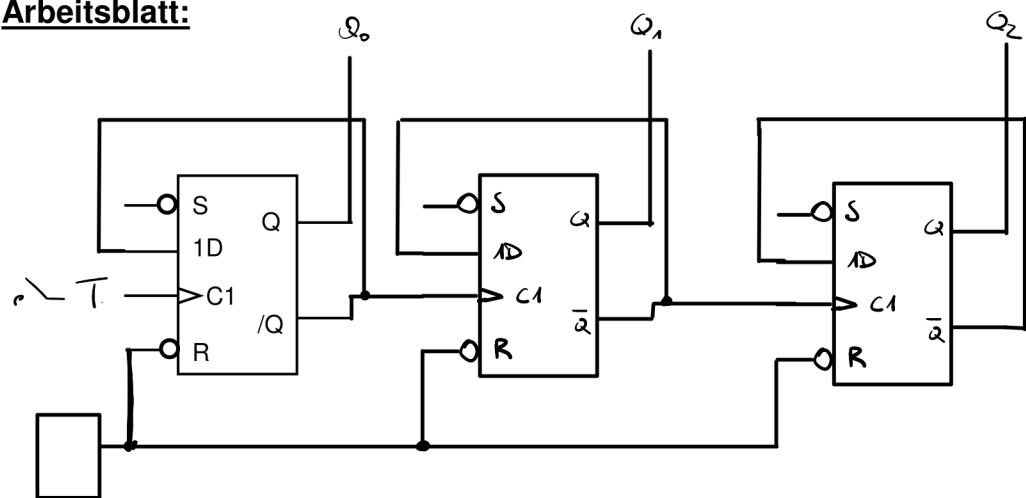
Der Vergleicher soll unter Verwendung von drei Äquivalenz-Gattern (1-Bit-Vergleichern) realisiert werden.

- 2.1 Beschreiben Sie die Funktion einer Äquivalenz. Geben Sie die Funktionsgleichung an.
- 2.2 Ergänzen Sie auf dem Arbeitsblatt (vgl. unten) den Schaltplan des vollständigen 3-bit-Vergleichers.

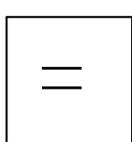
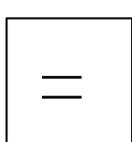
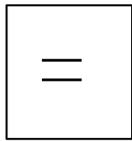
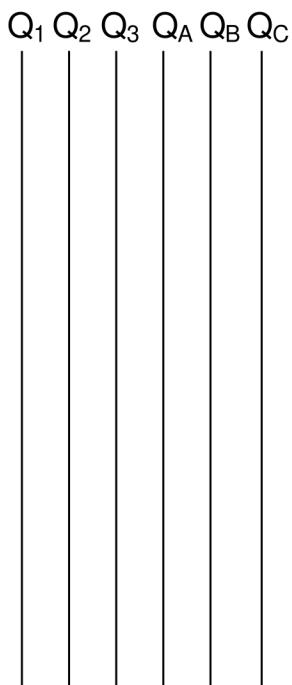
3. Bestätigung

Der Türöffner soll nur dann ein Signal T bekommen, wenn S5 (Bestätigung) kurz betätigt wird. Erweitern Sie auf dem Arbeitsblatt die Schaltung aus 2.2 entsprechend.

Arbeitsblatt:

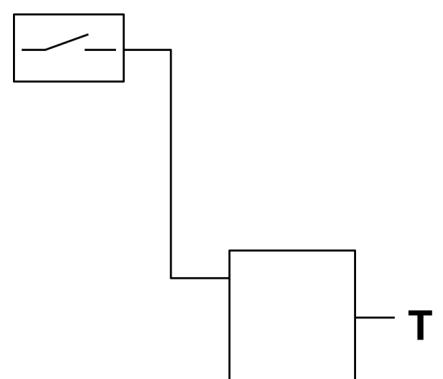


Zu 2.2 : 3-bit- Vergleicher



Zu 3 :

S5 (Bestätigung)

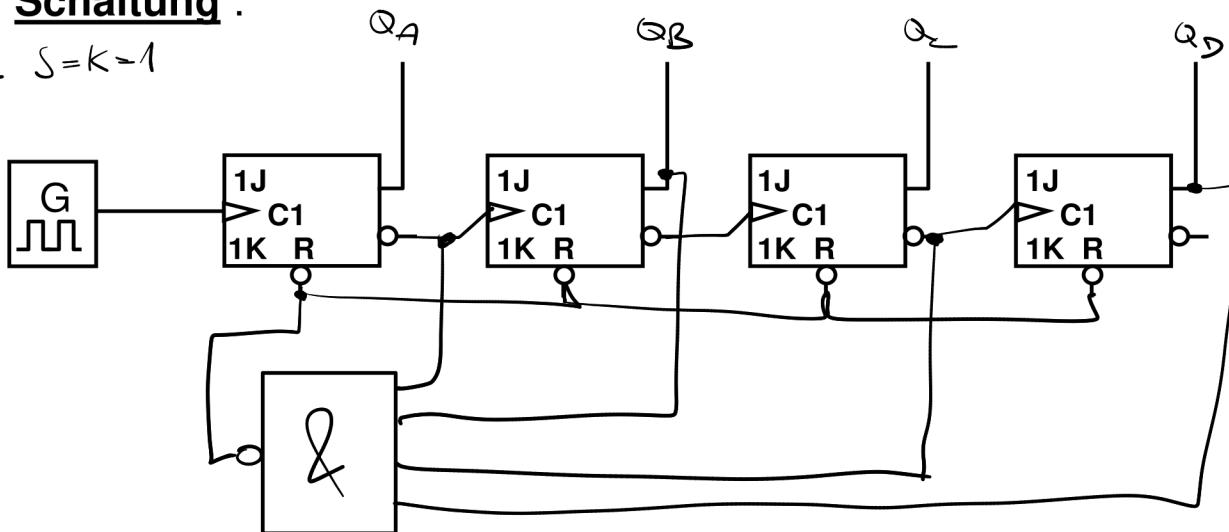


Dekinalzähler

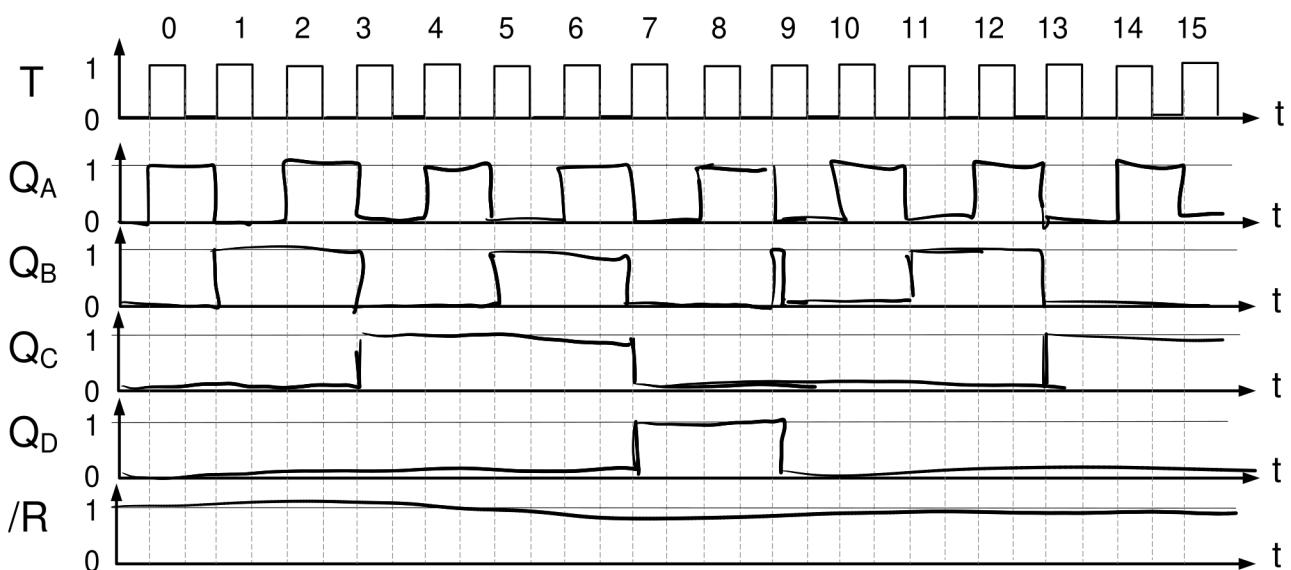
- Werden auch als asynchrone BCD - Vorrührzähler bezeichnet
- BCD-Zähler sind 4-Bit-Dual Zähler
- Der Zähler besteht aus 4 FFs (z.B. JK) und einer Rückstelllogik
- Rückstellung :**
 - nach 10_{dual} Zählschritten wird auf 0 zurückgesetzt
 - beginnt dann von vorn zu zählen
 - ⇒ Pseudotetraden 1010_{dual} ($= 10_{\text{dez}}$) bewirkt den Rückstellimpuls.

Schaltung :

Alle $s = k = 1$



Zeitablaufdiagramm :



Der zeitlich gedehnte Ausschnitt des 10_{dual} Takts zeigt, dass dieser Zähler kurzzeitig den Dualwert 1010_{dual} (dezimal 10_{dez}) ausgibt und erst danach auf 0 zurück gesetzt wird. Mit Beginn des 11_{dual} Takts startet die neue Dekade.

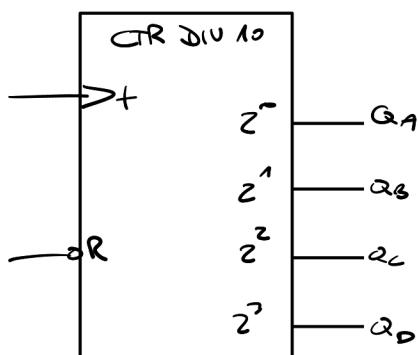
Wertetabelle:

	Q_D	Q_C	Q_B	Q_A	$/R$															
0	0	0	0	0	1															
1	0	0	0	1	1															
2	0	0	1	0	1															
3	0	0	0	1	1															
4	0	1	0	0	1															
5	0	0	1	0	1															
6	0	0	0	1	1															
7	0	1	0	1	1															
8	1	0	0	0	1															
9	1	0	0	1	0															
10	1	0	1	0	0															
11	1	1	0	0	1															
12	1	1	1	0	0															
13	1	1	1	1	0															
14	1	1	1	1	1															
15	1	1	1	1	1															

$$\bar{R} = \overline{Q_A \wedge Q_B \wedge \bar{Q}_C \wedge Q_D} \quad \bar{\bar{R}} = \overline{(\overline{Q_A \wedge Q_B \wedge \bar{Q}_C \wedge Q_D})} \quad R = (Q_A \wedge Q_D)$$

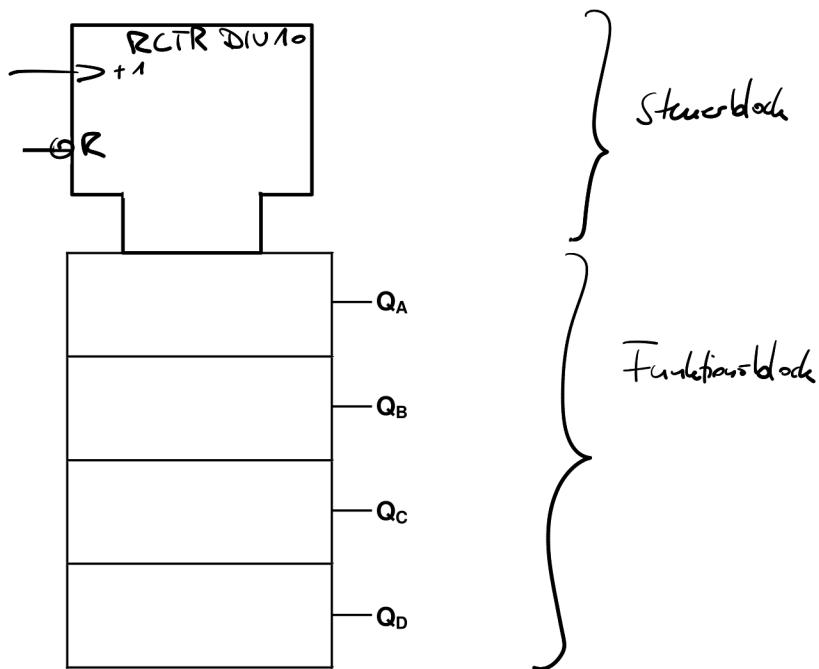
Blockschaltbilder des Dezimalzählers

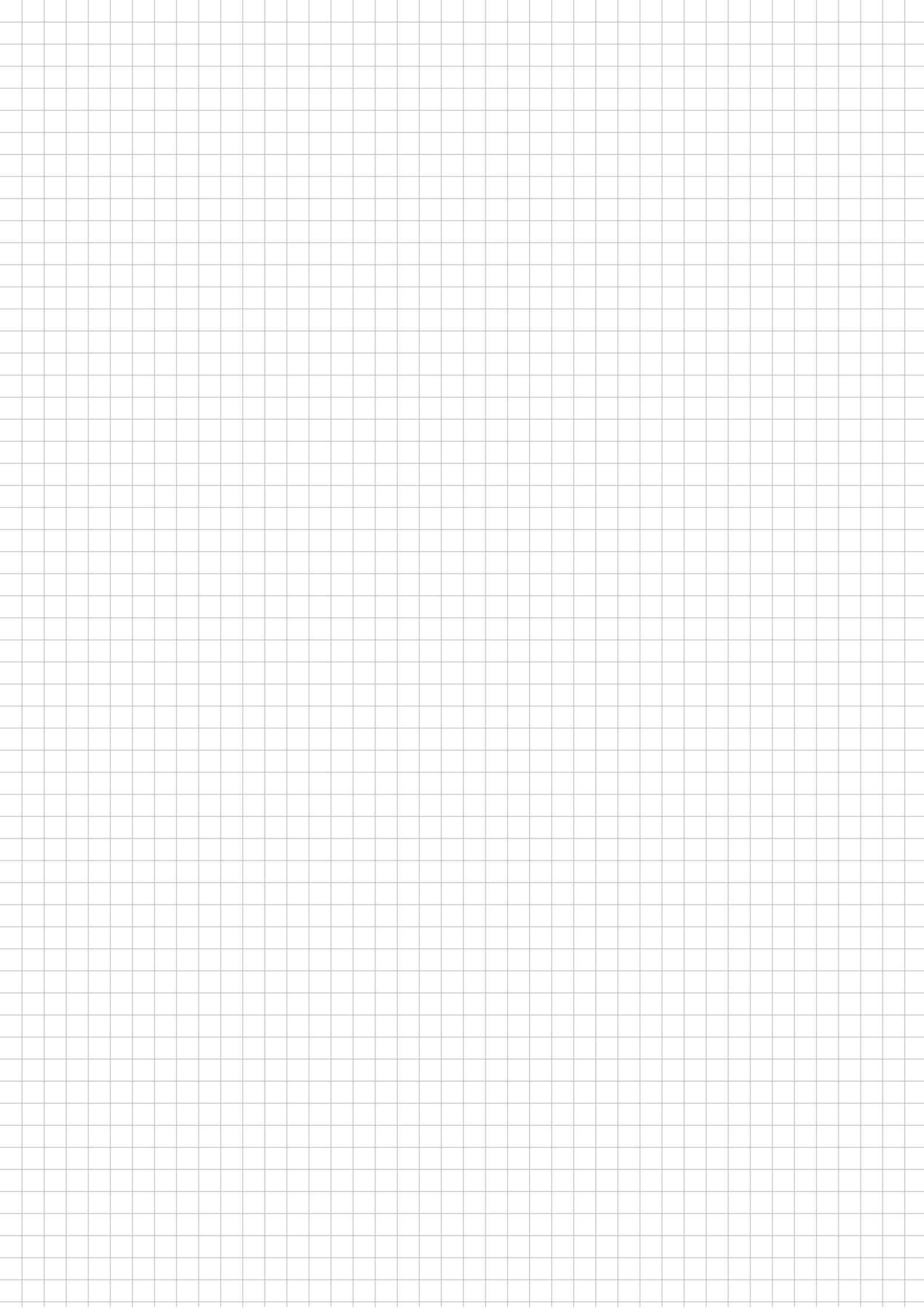
a)



CTR m	Counter $m = \text{number of bits}; \text{cycle length} = 2^m$
CTR DIV m	Counter with cycle length $= m$
RCTR m	Asynchronous counter; cycle length $= 2^m$

b)



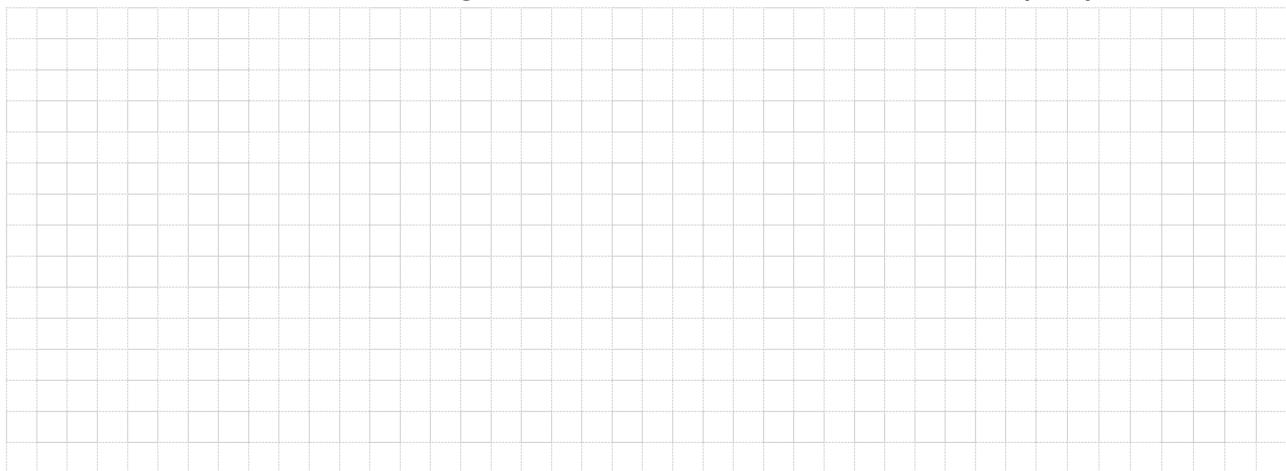


Aufgabe : Modulo-12-Zähler

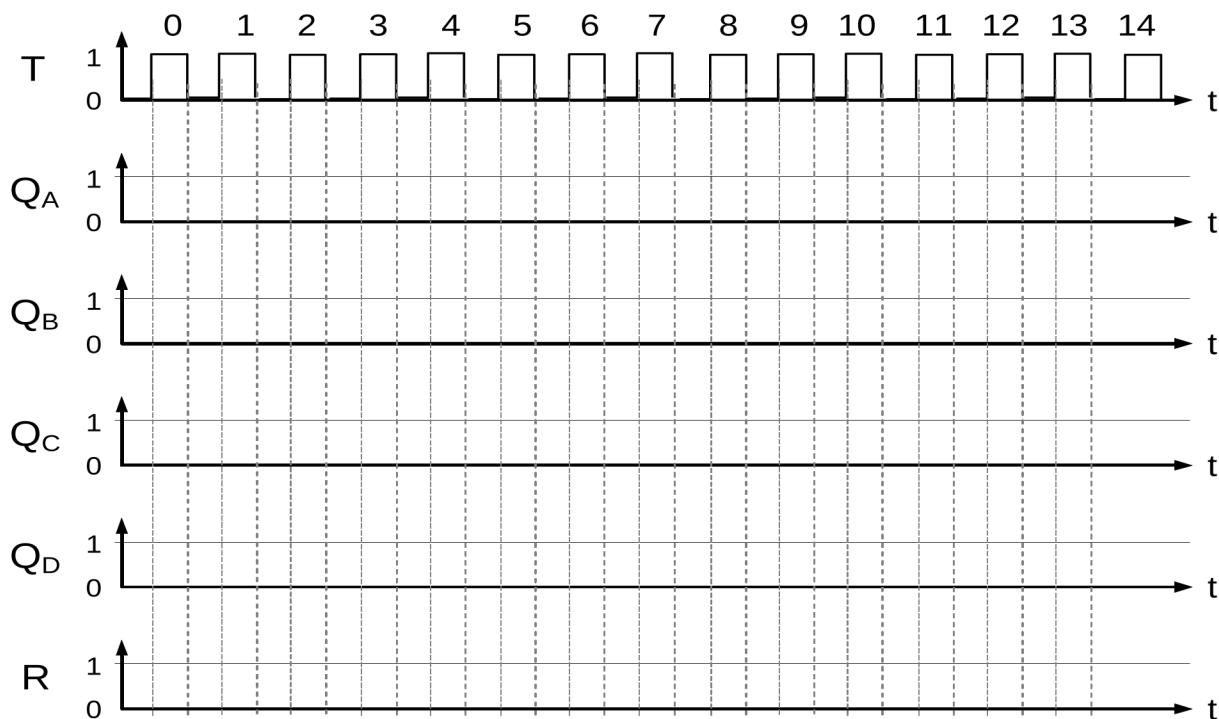
- a) Beschreiben Sie die Zählweise eines Modulo-12-Zählers.
 (Mit welcher Dual-/Dezimalzahl wird der Zähler zurückgesetzt?
 Anzahl der Zählimpulse? Höchster angezeigter Zahlenwert)

.....

- b) Zeichnen Sie die Schaltung eines Modulo-12-Zählers aus Flipflops Ihrer Wahl.



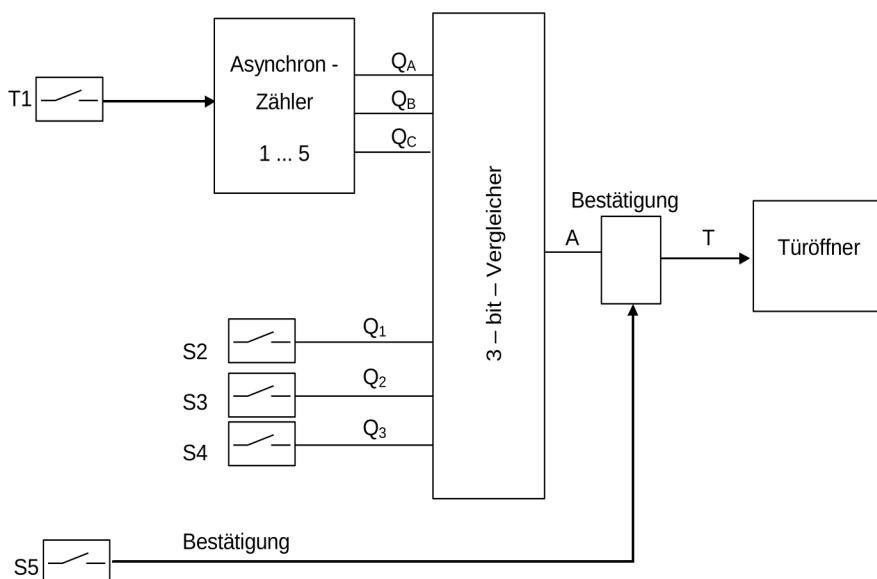
- c) Ergänzen/Zeichnen Sie das Zeitablaufdiagramm zu diesem Zähler.
 (Rückstellung erfolgt mit log.0).



Aufgabe : Türöffner mit Codeschloss (Vereinfachter Auszug)

Das Blockschaltbild zeigt den Aufbau der Anlage. Sie besteht aus folgenden Schaltungsteilen:

- Asynchronzähler zur Voreinstellung der Codenummer
Mit Hilfe des Tasters T1 können mit dem Asynchronzähler die Dezimalzahlen 1... 5 im BCD-Code eingestellt werden.
- Eingabe der Codenummer mit Speicherung
Mit den Schaltern S2 ... S4 wird die Codenummer binär eingegeben.
- 3-bit-Vergleicher mit Bestätigung
Der Vergleicher prüft, ob die eingestellte Dezimalzahl Q_1, Q_2, Q_3 mit der eingegebenen Dezimalzahl Q_A, Q_B, Q_C übereinstimmt.
Mit dem Schalter S5 wird das Ergebnis des Vergleichers weitergegeben, welches dann den Türöffner ansteuert.



1. Entwurf des Asynchronzählers von 1 bis 5

- 1.1 Wie viele Speicherbausteine braucht der Zähler?
- 1.2 Ergänzen Sie auf dem Arbeitsblatt die Schaltung zu einem 1.1 entsprechenden asynchronen Dual-Vorwärtzähler.
- 1.3 Die Schaltung ist nun so zu erweitern, dass der Zähler von "1" bis "5" zählt (ohne Schaltnetz, /S, /R nutzen!).

2. Vergleicher

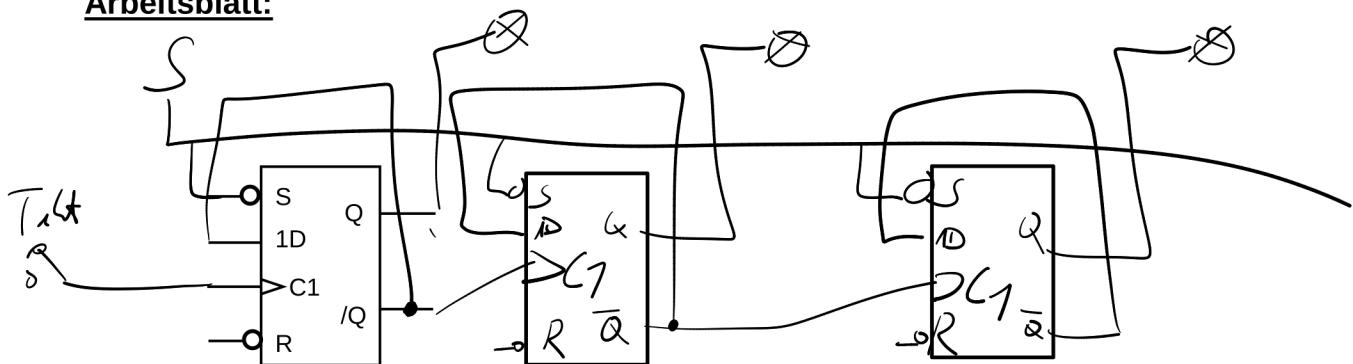
Der Vergleicher soll unter Verwendung von drei Äquivalenz-Gattern (1-Bit-Vergleichern) realisiert werden.

- 2.1 Beschreiben Sie die Funktion einer Äquivalenz. Geben Sie die Funktionsgleichung an.
- 2.2 Ergänzen Sie auf dem Arbeitsblatt (vgl. unten) den Schaltplan des vollständigen 3-bit-Vergleichers.

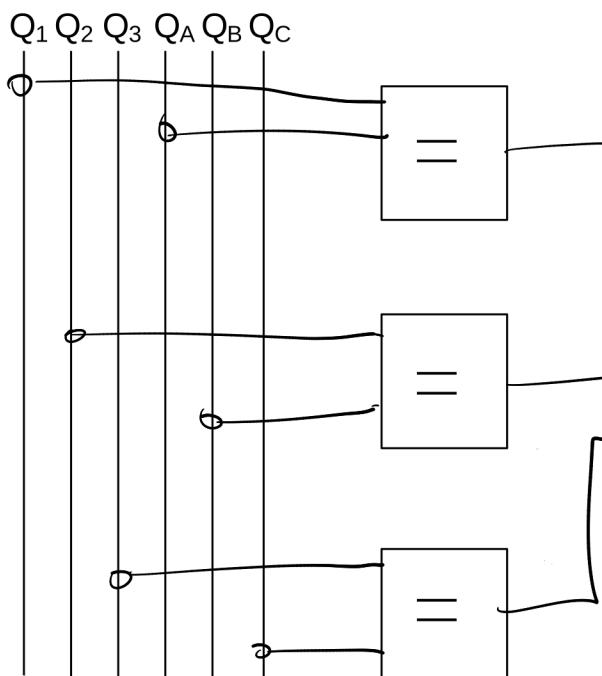
3. Bestätigung

Der Türöffner soll nur dann ein Signal T bekommen, wenn S5 (Bestätigung) kurz betätigt wird. Erweitern Sie auf dem Arbeitsblatt die Schaltung aus 2.2 entsprechend.

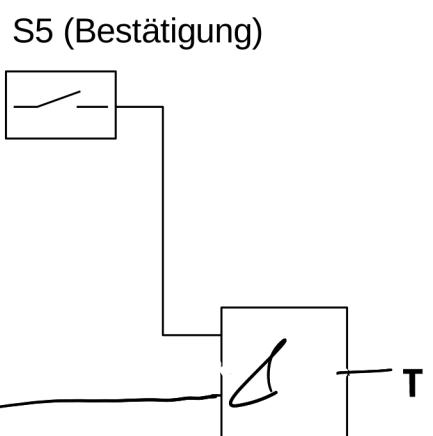
Arbeitsblatt:

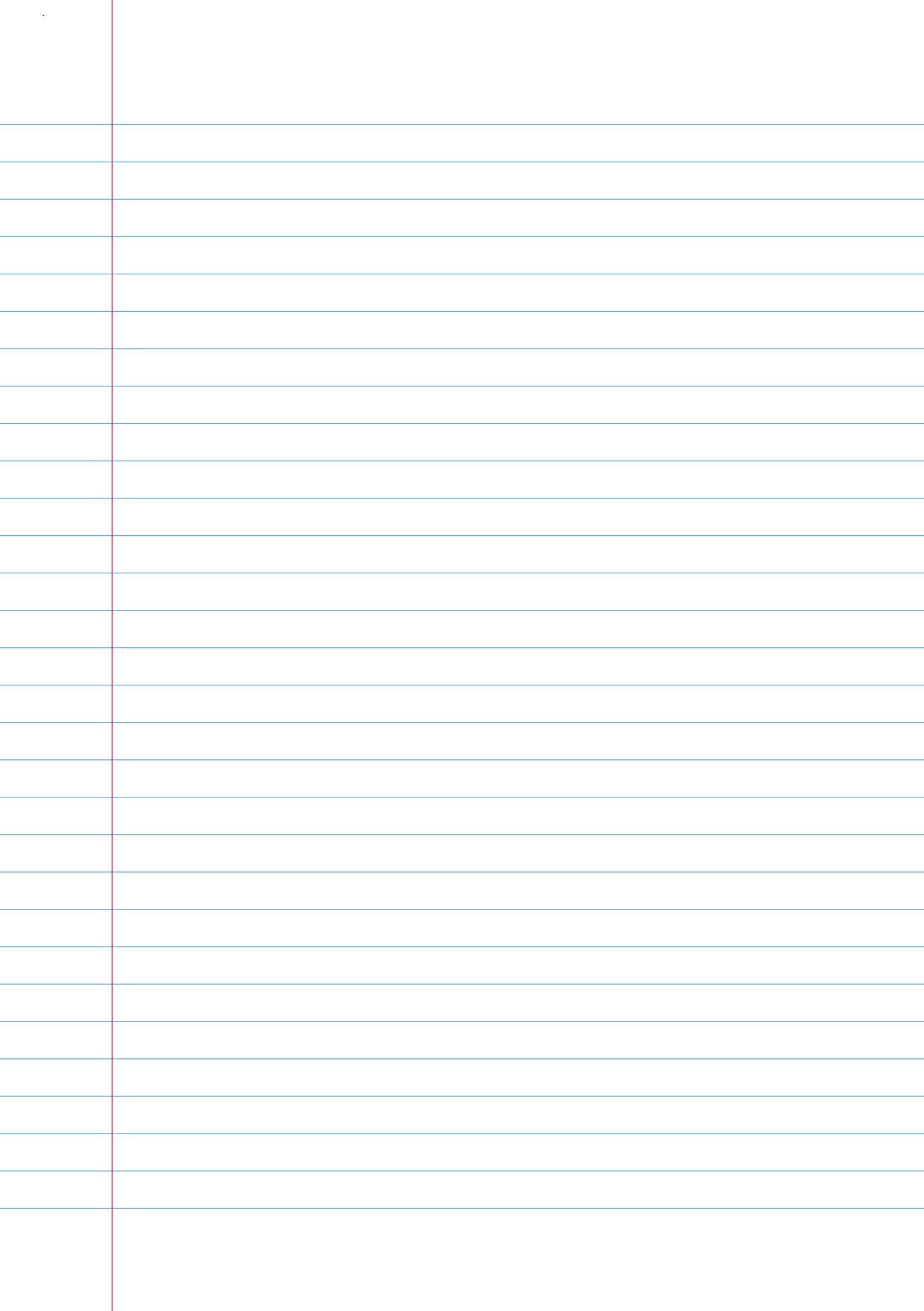


Zu 2.2 : 3-bit- Vergleicher



Zu 3 :





a)

	A	\bar{A}			
B	1_3	1_7	1_6	1_2	\bar{D}
\bar{B}	1_{15}	1_{15}	1_{10}	1_{10}	D
	1_5	1_3	1_{12}	1_7	\bar{D}
	1_1	1_6	1_4	1_3	\bar{D}

$$\alpha = (B \wedge \bar{D}) \vee (\bar{A} \wedge \bar{C}) \vee (C \wedge B \wedge D) \vee (\bar{A} \wedge D \wedge C) \vee \\ (\bar{A} \wedge A \wedge \bar{C}) \vee (A \wedge C \wedge \bar{D})$$

b)

	A	\bar{A}			
B	1_3	1_2	6	1_2	\bar{D}
\bar{B}	1_1	1_5	1_4	1_6	D
	1_3	1_3	1_2	1_8	\bar{D}
	1_1	1_4	1_3	1_3	\bar{D}

$$\beta) (\bar{A} \wedge \bar{C}) \vee (\bar{A} \wedge \bar{D} \wedge \bar{B}) \vee (A \wedge \bar{B} \wedge D) \vee \\ (A \wedge B \wedge \bar{C}) \vee (A \wedge B \wedge \bar{D})$$

c)

	A	\bar{A}			
B	3	7	6	2	\bar{D}
\bar{B}	11	15	14	10	D
	2	13	12	8	\bar{D}
	1	6	4	5	\bar{D}

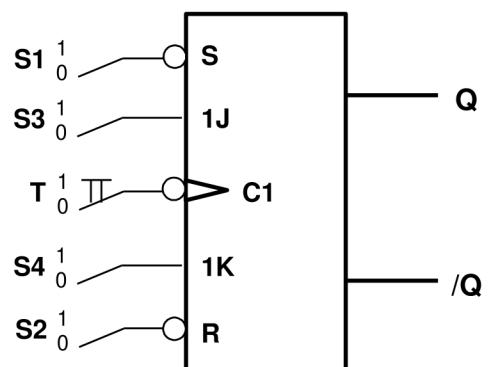
Aufgabe 1 : Grundfunktionen des JK - Flipflop

Ziel: Es soll die Beschaltung und Funktionsweise eines JK-Flipflops erklärt werden.

- a) Beschalten Sie ein JK-Flipflop des IC 7476 entsprechend nebenstehendem Stromlaufplan.

Ermitteln Sie die Ausgangszustände zum Zeitpunkt t_{n+1} (nach dem Takt), ergänzen Sie dazu die Spalten Q_{n+1} , $/Q_{n+1}$ und Bemerkungen in der Wertetabelle.

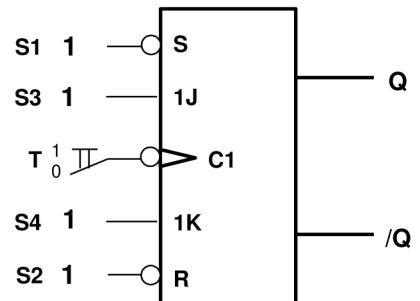
Vorgehen: Nach Spalte t_n jeweilige Zeile einstellen, takten, anschließend neuen Zustand t_{n+1} eintragen usw....



t_n				t_{n+1}		Bemerkungen
S1	S2	S3	S4	Q_{n+1}	$/Q_{n+1}$	
1	1	1	0	1	0	Setzen
1	1	0	0	1	0	Speichern
1	1	0	1	0	1	Rücksetzen
1	1	0	0	0	1	Speichern
1	1	1	0	1	0	Setzen
1	0	0	0	0	1	Taktunab. Rücksetzen
0	1	0	1	1	0	Taktunab. Setzen

- b) Beschalten Sie das JK-Flipflop laut nebenstehendem Stromlaufplan.

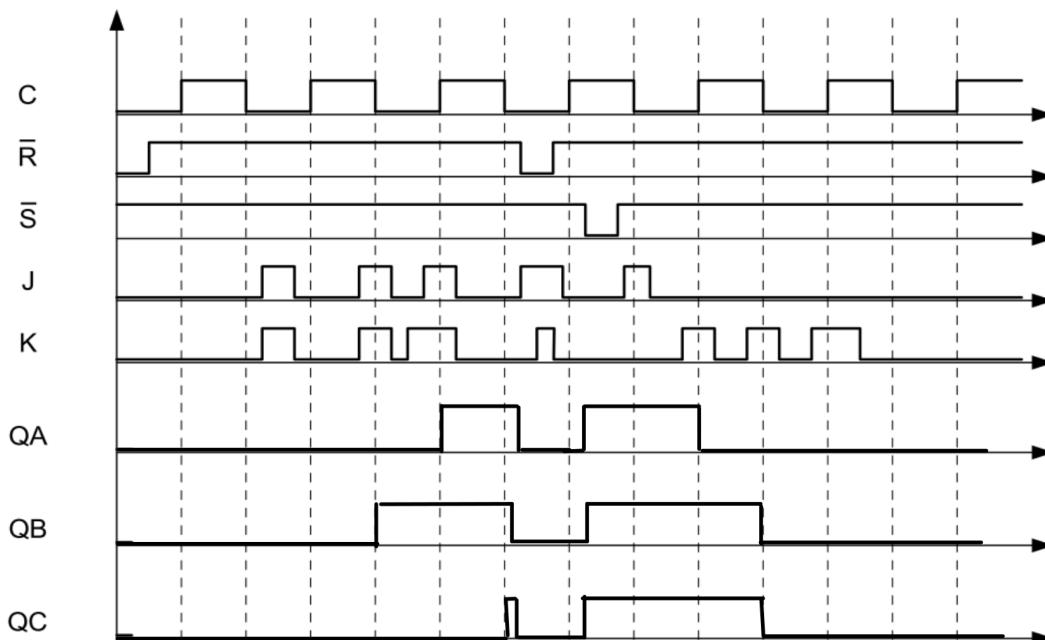
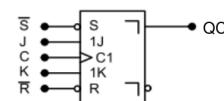
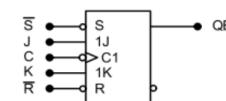
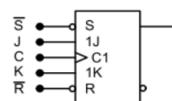
Ermitteln Sie die Ausgangszustände zum Zeitpunkt t_{n+1} , ergänzen Sie die Wertetabelle.



t_n	t_{n+1}		Bemerkungen
	Q_{n+1}	$/Q_{n+1}$	
S1 = S2 = S3 = S4 = 1	1	0	Setzen
	0	1	Rücksetzen
	1	0	Setzen
	0	1	Rücksetzen
	1	0	Setzen
	0	1	Rücksetzen
	1	0	Setzen

} Toggel-mode
Umschaltmodus

- c) Zeichnen Sie das jeweilige Ausgangssignal des Flipflops. Setzen Sie dabei für den Anfangszustand jeweils $Q = 0$ voraus.

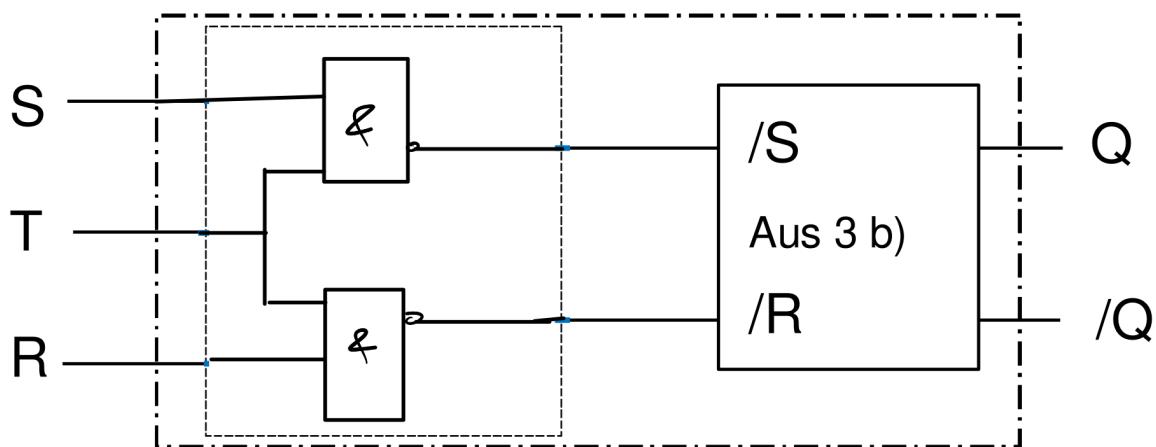


Aufgabe 4: Taktzustandgesteuerte RS - Kippstufe

Ziel: Aufbau und Funktionsweise taktabhängiger Speicherschaltungen.

- a) Ein \overline{RS} -FF nach Aufgabe 3 b) kann durch zwei zusätzliche NAND-Verknüpfungen zu einem Flipflop mit Takteingang (engl.: clock) erweitert werden.

1. Ergänzen Sie den gegebenen Schaltplan entsprechend (ähnlich INF/T) und erweitern Sie Ihre Schaltung aus voriger Aufgabe.
2. Ergänzen Sie die Wertetabelle durch zeilenweises Vorgehen.
3. Zu welchen Zeitpunkten kann das FF nur Informationen übernehmen ?
4. Vervollständigen Sie das folgende Zeitablauf-Diagramm.



Wertetabelle:

Situation vor
Taktsignal

Situation nach
Taktsignal

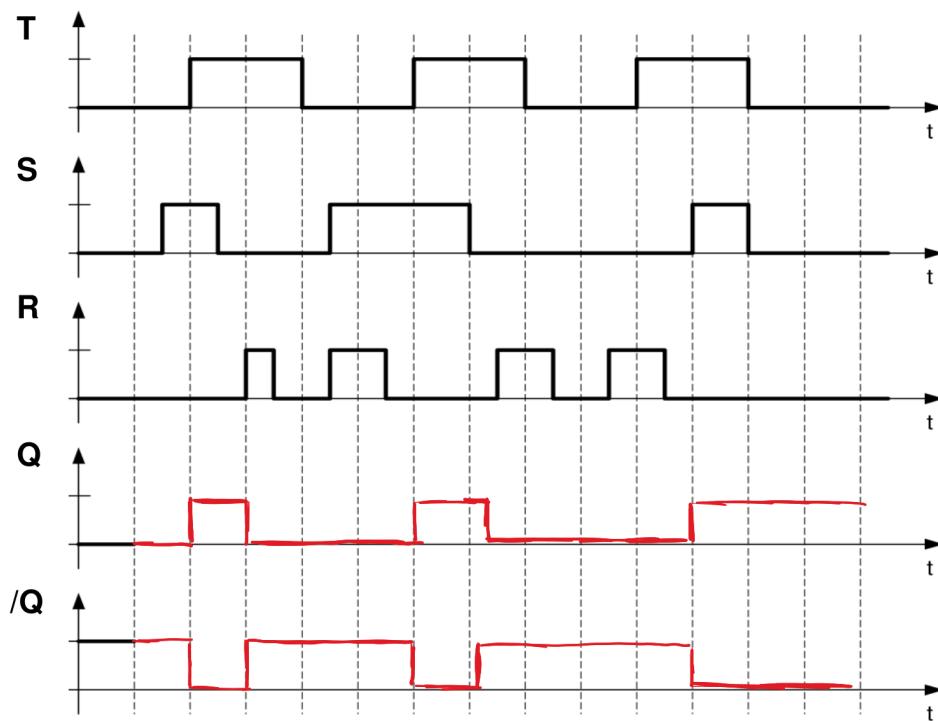
	t_n				t_{n+1}				
T	S	R	Q_n	$/Q_n$	Q_{n+1}	$/Q_{n+1}$	Bemerkungen		
0	0	0	0	1	0	1	Spicken		
1	1	0	0	1	1	0	Set		
1	0	1	1	0	0	1	Reset		
0	1	0	0	1	0	1	Spicken		
1	1	0	0	1	1	0	Set		
1	0	1	1	0	0	1	Reset		
1	0	0	0	1	0	1	Spicken		
0	1	1	0	1	0	1	Spicken		
1	1	0	0	1	1	0	Set		
0	1	1	0	1	0	1	Spicken		

Vorgehen: Q_n bzw. $/Q_n$ eintragen, neue Kombination A, B einstellen, T $\overline{\square}$ ausführen, Q_{n+1} bzw. $/Q_{n+1}$ eintragen, Bemerkung durch Vergleich $Q_n \leftrightarrow Q_{n+1}$ schreiben, Q_n darunter eintragen,

$\overline{T} = 0 = \text{spicken}$

nicht
speckbar

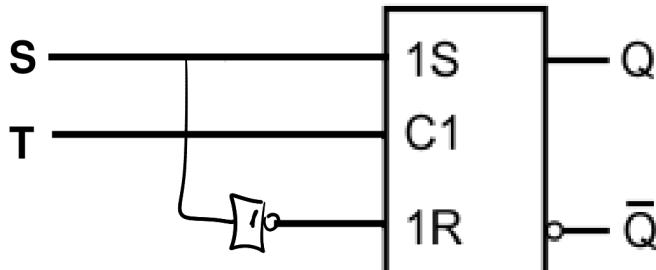
Zeitblaufdiagramm:



Aufgabe 5:

Ziel: Die vorige Schaltung soll nun so zu einem weiteren Flipflop erweitert werden, dass der nicht speicherbare Zustand ausgeschlossen wird.

- a) Ergänzen Sie die Schaltung so, dass bei $S = 1$ der Eingang $R = 0$ wird und umgekehrt.



- b) Bauen Sie die Schaltung auf und vervollständigen Sie die Wertetabelle.

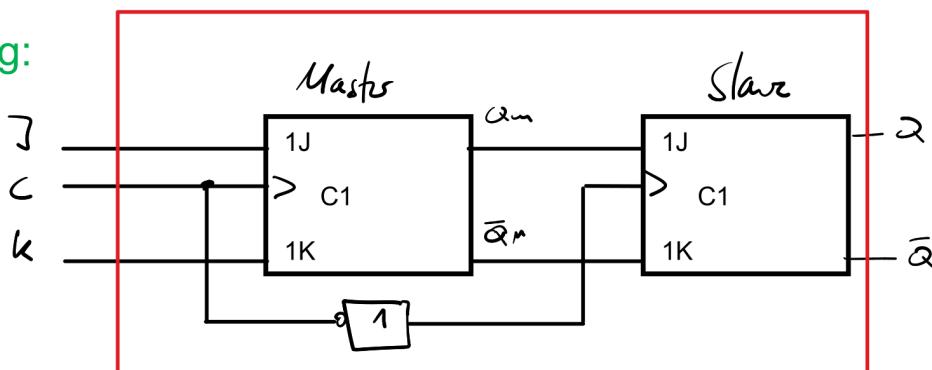
Wertetabelle :

	t_n	t_{n+1}		
T	S	Q_{n+1}	$/Q_{n+1}$	Bemerkungen
1	0	0	1	Reset
0	1	0	1	Speichern
1	1	1	0	Set
1	0	0	1	Reset
0	0	0	1	Speichern
0	1	0	1	Speichern
1	1	1	0	Set

Das JK-MS-Flipflop

Zweiflankenget. Flipflop

Schaltung:

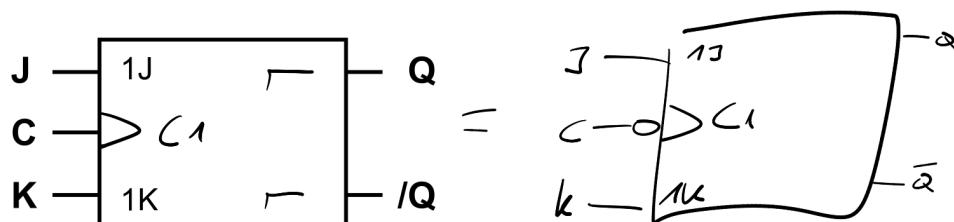


Funktion:

1. $C = \top$: Master übernimmt Eingangssignal
Slave blockiert

2. $C = \perp$: Slave übernimmt Eingangssignal $Q_m(\rightarrow 0)$
Master blockiert.

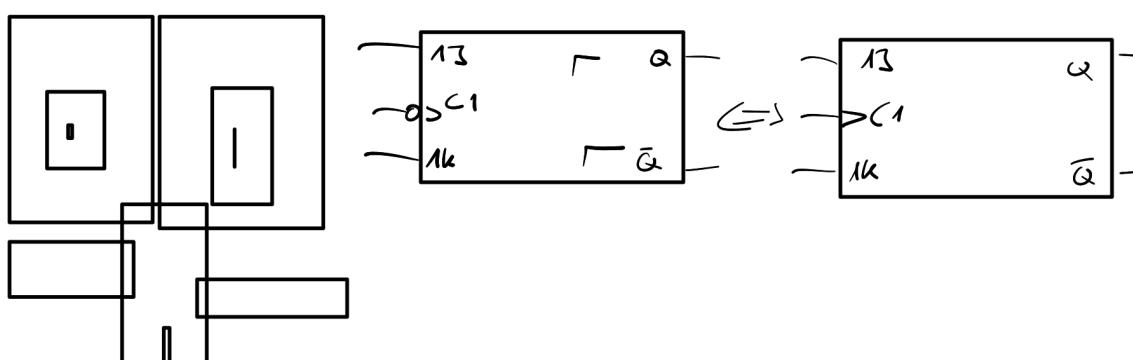
Symbol:



— Jozug genau um $\frac{1}{2}$ Takt

Negativ-taktflanken gesteuertes Flipflop JK-MS-FF

Positiv-taktflanken gesteuerte JK-MS-FF

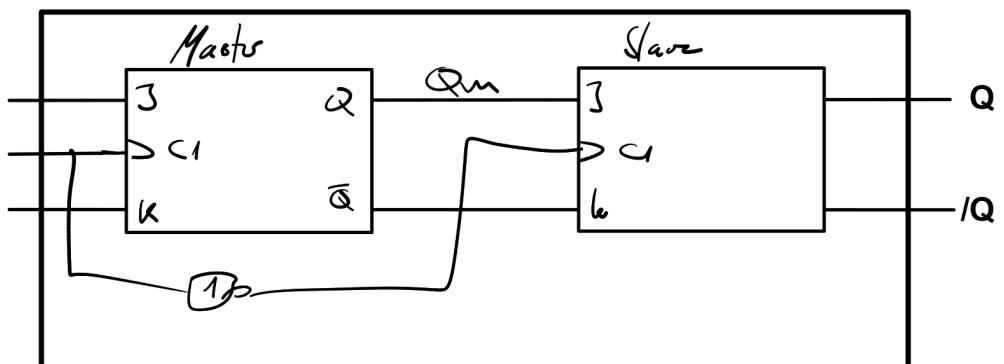


Aufgabe 1: Das JK - Master - Slave - Flipflop

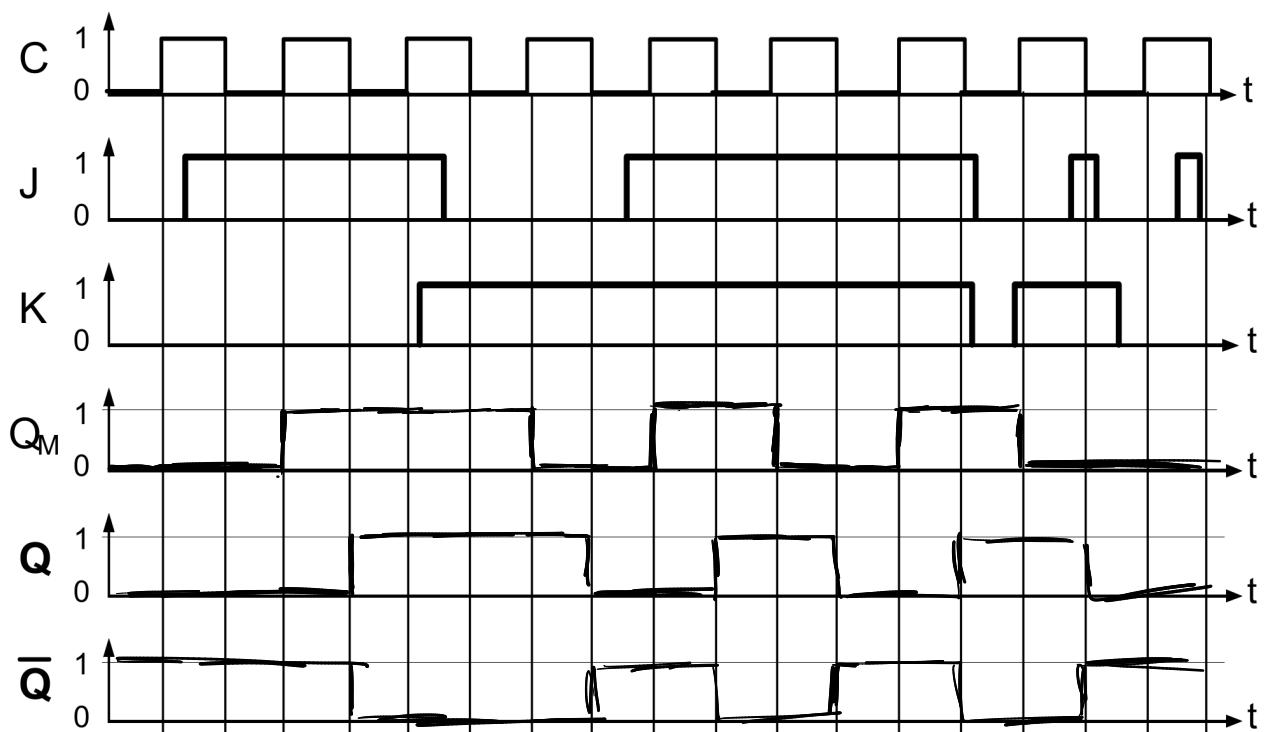
zweiflankengesteuertes JK-MS-Flip-Flop *Jump Kill Master Slave FlipFlop*

Vervollständigen Sie die Schaltung und das Zeitablaufdiagramm eines aus zwei positiv taktflankengesteuerten JK-Flipflops bestehenden JK-Master-Slave-Flipflops.

Schaltung :



Zeitablaufdiagramm :



Link:

<https://studyflix.de/informatik/jk-master-slave-flipflop-994>

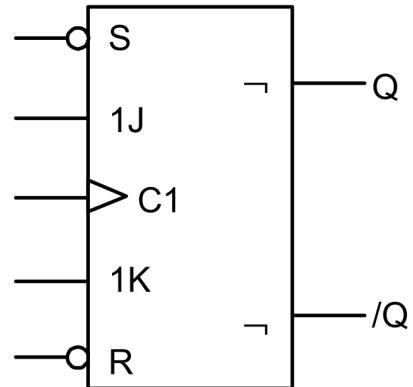


Aufgabe 2: Flipflop

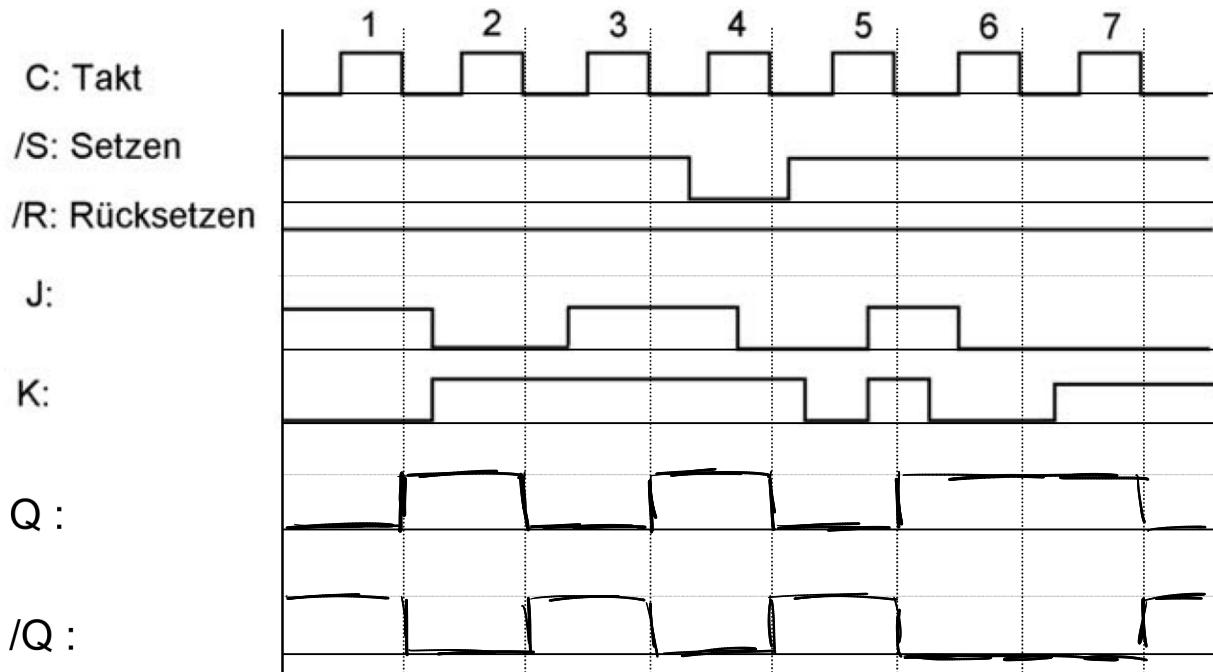
Gegeben sei das nebenstehende Schaltsymbol eines bestimmten Flipflops.

- a) Beschreiben Sie die Eigenschaften, die sich aus dem Schaltsymbol ergeben.

- \triangleq zweiflankengesteuertes JK-Master-Slave-Flipflop (negativ-flankengesteuert)
- \rightarrow Master-Slave-Architektur d.h. C : halber Takt verzögert
- S und R sind lowaktive (takt-)unabhängige Eingänge höherer Priorität
- $C1$ ist steuernder Eingang, Ziffer 1 gibt an, welche Eingänge von C abhängig sind.
- $1J, 1K$ sind die gesteuerten Eingänge



- b) Ergänzen Sie im gegebenen Zeitablaufdiagramm den Signal-Verlauf an den Ausgängen Q und $/Q$.



Aufgabe 1: Multiple Choice

Ein Register ...

- ... wird zur Addition von Dualzahlen verwendet
- ... ist aus Flipflops aufgebaut
- ... ist eine Kombination aus Volladdierern
- ... dient nur zur kurzzeitigen Speicherung von Informationen

Richtige Antwort(en) ankreuzen.



Anschauen!: Video: IFDO 10 -Was ist ein FlipFlop?
<https://www.youtube.com/watch?v=SnBsA1hKTiE>

Aufgabe 2: SR – Speicherschaltungen, Basis-Flipflop

Beantworten Sie folgende Fragen mit Hilfe Ihres Inf/T-Aufschriebes:

- a) Welche 3 definierten Funktionen besitzt ein Basis-Flipflop?

Set (setzen) S Reset (Rücksetzen) R Speichern

- b) Nennen Sie den Ausgangszustand, bei dem ein Flipflop „gesetzt“ bzw. „rückgesetzt“ ist.

Ausgang Q = 1 Ausgang Q = 0 / $\bar{Q} = 1$

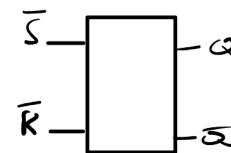
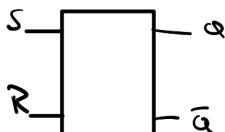
- c) 1. Unterscheiden Sie „highaktives“ und „lowaktives“ RS-Flipflop.

Highaktiv: S=1 R=0 Q=1 $\bar{Q}=0$ Lowaktiv: S=0 R=1 Q=0 $\bar{Q}=1$

2. Welche Eingangssignale liegen bei „Speichern“ an den Eingängen S, R jeweils an?

S = R = 0 Highaktiv S = R = 1 Lowaktiv

3. Zeichnen Sie die Schaltsymbole eines high- und eines lowaktiven SR-Flipflops (Basis-FF).



OWH

Aufgabe 3: RS - Speicherschaltungen aus Gattern

Ziel: Aufbau und Funktionsweise einfacher Speicherschaltungen aus NOR und NAND.

- a) Bauen Sie ein **RS- Flipflop aus NOR - Gattern** auf (Schaltung siehe INF/T) und ergänzen Sie die nachstehende Wertetabelle durch zeilenweises Einstellen der vorgegebenen Eingangssignale Ihrer Schaltung.

Wertetabelle :

Zeile	Ein-gang 1	Ein-gang 2	Ausgang 1	Ausgang 2	
	S	R	Q_{n+1}	$/Q_{n+1}$	Bemerkungen
0	0	1	0	1	Reset
1	1	0	1	0	Set
2	0	0	\bar{Q}_n	\bar{Q}_n	Speicherbar
3	0	1	0	1	Reset
4	0	0	1	0	Set
5	1	1	0	0	Nicht speicherbar

- b) 1. Zeichnen Sie ein **RS-Flipflop aus 2 NAND-Gattern** (lowaktives RS-Flipflop; Schaltung siehe INF/T) und bauen Sie dieses auf.

Hinweis: Dieser Aufbau wird in den folgenden Aufgaben weiterhin benötigt!

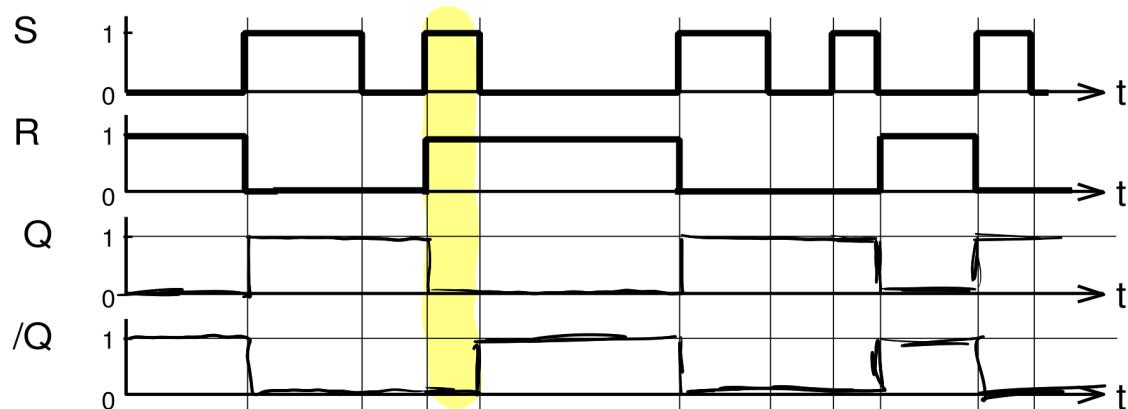
2. Ergänzen Sie die Wertetabelle wie vor beschrieben (siehe auch Inf/T-Ordner).

Wertetabelle :

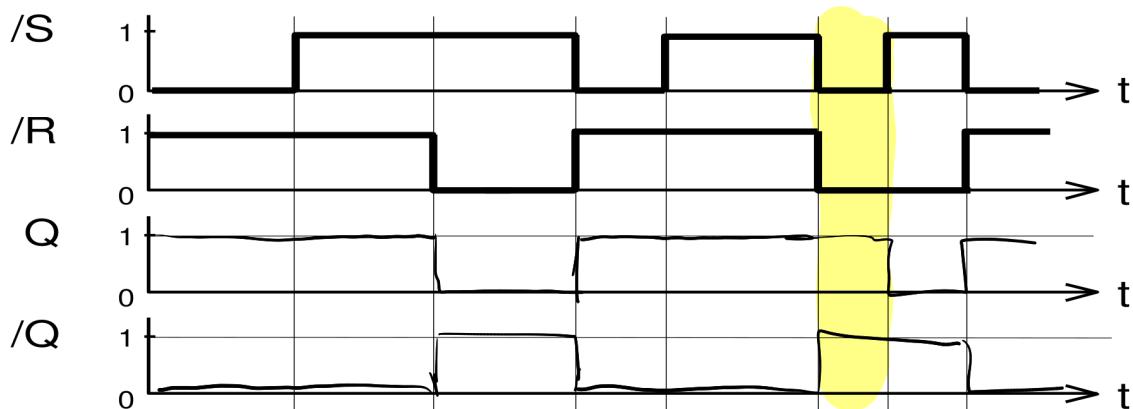
Zeile	Ein-gang 1	Ein-gang 2	Ausgang 1	Ausgang 2	
	/S	/R	Q_{n+1}	$/Q_{n+1}$	Bemerkungen
0	0	1	1	0	Set
1	1	0	0	1	Reset
2	1	1	\bar{Q}_n	Q_n	Speichern
3	0	1	1	0	Set
4	1	1	0	1	Reset
5	0	0	\bar{Q}_n	Q_n	Nicht speicher

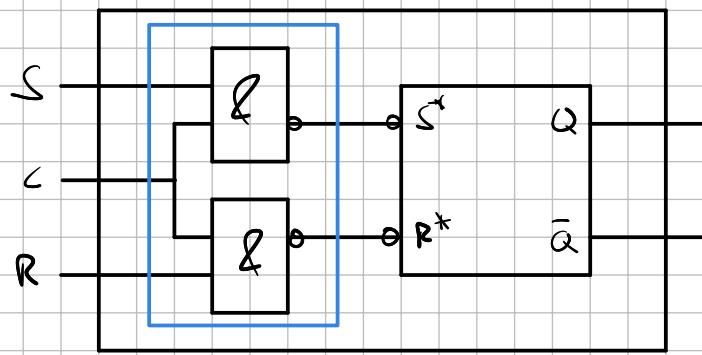
- c) Ergänzen Sie das Signal-Zeit-Diagramm des highaktiven und des lowaktiven RS-Flip-Flops.

Highaktiv:



Lowaktiv:



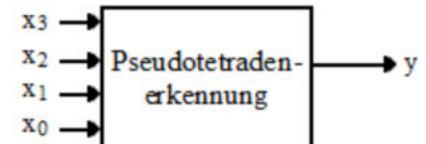


Aufgabe 1: Codes, Pseudotetradenerkennung

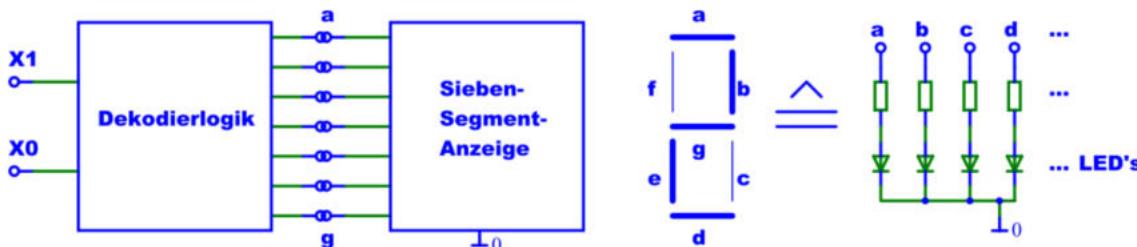
- Was ist ein Code?
- Nennen Sie 3 verschiedene Arten von Codes mit jeweils einem Beispiel!
- Zur welcher Klasse von Codes gehört der ASCII?
- Erklären Sie die Bezeichnung „BCD-Code“ (Bedeutung, Stellenzahl, Nutz- und Pseudotetraden, Wertigkeit etc.).

Bei Vorliegen einer Pseudotetraden eines 8-4-2-1-Codes soll am Ausgang der Schaltung eine "1" erzeugt werden.

- Bestimmen Sie die minimierte Gleichung und zeichnen Sie die Schaltung.



Aufgabe 2: Siebensegment-Dekoder



Mit einer Siebensegmentanzeige und einer Dekodierlogik sollen 2-Bit-Binärzahlen als Dezimalzahlen (0-3) angezeigt werden. Die Anzeigeeinheit besteht aus einer Anordnung von sieben Leuchtdioden, die jeweils bei einer logischen 1 aufleuchten.

- Ermitteln Sie die vereinfachten Schaltfunktionen für jedes Segment.
- Zeichnen Sie den Schaltplan der Segmente a und f unter der ausschließlichen Verwendung von NAND-Gattern.

Aufgabe 3: 1-aus-10-Decoder

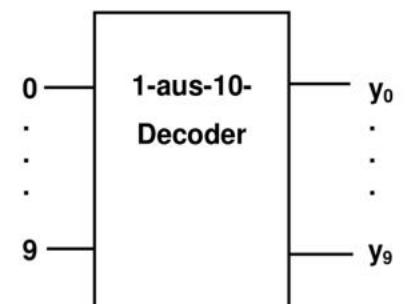
Ein Decoder, der im BCD-Code dargestellte Dezimalziffern 0-9 in den 1-aus-10-Code umsetzt, soll als Schaltnetz realisiert werden.

Vorteil: einfache Fehlererkennung

Nachteil: großer Aufwand

Anwendung: zur Anzeige, numerische Tastatur

- Beschreiben Sie den 1-aus-10-Code.
- Wie viele Ein- und Ausgänge sind nötig?
Benennen Sie diese.
- Geben Sie die zugehörige Funktionstabelle an.
- Realisieren Sie den Decoder als Schaltnetz.



Wertigkeit	9	8	7	6	5	4	3	2	1	0
Dezimalziffern										
0										
1										
2										
3										
4										
5										
6										
7										
8										
9										

<https://www.itwissen.info/1-aus-n-Code-1-out-of-n-code.html>

<https://de.wikipedia.org/wiki/1-aus-n-Code>

① a) Ein Code ist eine Abbildungsvorschrift, die dazu dient Informationen anders darzustellen oder zusammenzufügen.

- b)
- ASCII Code : 'A' = 01000001
 - Binär Code : 0101 = 5
 - Morsecode Code : ... - - - ... = Sos

c) Textcodierung / Alphanumerisch

d) ein System, bei dem jede dezimale Ziffer von 0 bis 9 jeweils auf vier Bits dargestellt wird.

x_0	x_1	a	b	c	d	e	f	g	
0	0	1	1	1	1	1	1	0	0
0	1	0	1	1	0	0	0	0	1
1	0	1	1	0	1	1	0	1	2
1	1	1	1	1	1	0	0	1	3

a	x_0	\bar{x}_0	b	x_0	\bar{x}_0	c	x_0	\bar{x}_0	d	x_0	\bar{x}_0
x_1	1	1	x_1	1	1	x_1	1	0	x_1	1	1
\bar{x}_1	0	1	\bar{x}_1	1	1	\bar{x}_1	1	1	\bar{x}_1	0	1

e	x_0	\bar{x}_0	f	x_0	\bar{x}_0	g	x_0	\bar{x}_0
x_1	0	1	x_1	1	0	x_1	1	1
\bar{x}_1	0	1	\bar{x}_1	0	1	\bar{x}_1	0	0

$$a = x_1 \ 1 \ \bar{x}_0$$

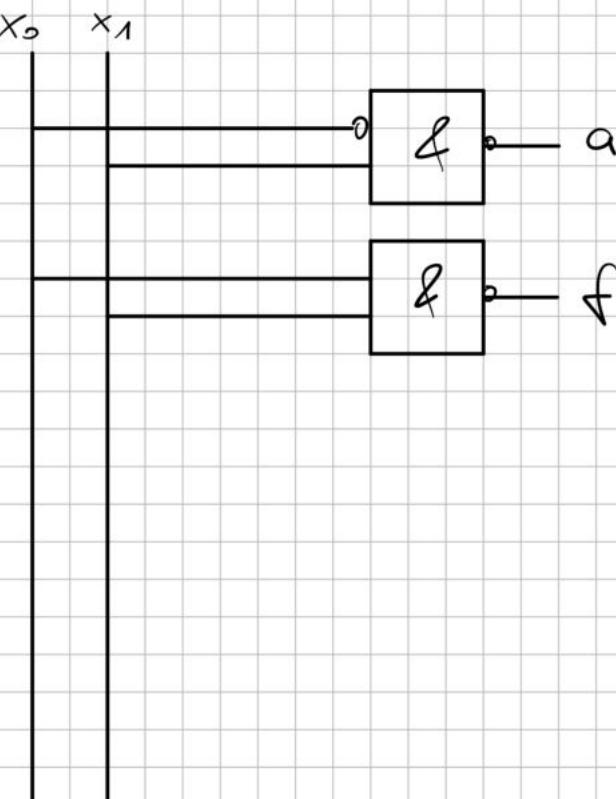
$$b = 1$$

$$c = x_0 \ 1 \ \bar{x}_1$$

$$d = x_0 \ 1 \ x_1$$

$$e = x_0$$

$$f = \bar{x}_0 \ 1 \bar{x}_1 \quad g = x_1$$



3

a)

Im 1-aus-10-Code verschiebt sich das jeweilige Einser-Bit mit steigender Dezimalzahl in dem zehnstelligen Datenwort vom Least Significant Bit (LSB) für den Dezimalwert 0: 0000000001, zum Most Significant Bit (MSB) für die Ziffer 9: 1000000000.

b) 10 Eingänge 0 - 9

10 August 20 - 20

Aufgabe 4: Codewandler

Das nachfolgende Diagramm 1 zeigt die Abwicklung der nebenstehenden Codierscheibe mit Gray-Code (Bild1).

- Berechnen Sie den Drehwinkel beim Wechsel zur nachfolgenden Stelle.
 - Ist der Gray-Code ein
 - tetradesischer ,
 - einschrittiger ,
 - bewertbarer ,
 - stetiger Code?
 - Tragen Sie in das Diagramm 2 die zugehörigen Bitkombinationen ein. Auch hier gilt: eingeschwärzt = 1.
- Es soll im Folgenden ein Codewandler vom Gray-Code in den „normalen“ Dualcode entwickelt werden. Hier allerdings nur für die Elemente d_3 und d_0 .
- Ermitteln Sie für den o.g. Codewandler die vollständige Schaltfunktion (DNF) für das Bit d_3 .
 - Ermitteln Sie für den o.g. Codewandler die minimierte Schaltfunktion (DMF) für das Bit d_2
 - Zeichnen Sie die Schaltung für Bit d_2 in Full-Nand-Darstellung.



Bild 1

Gray/BIN	
g_3	d_3
g_2	d_2
g_1	d_1
g_0	d_0

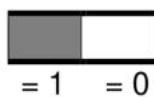
Bild 2

Diagramm 1: Gray-Code

g_3																
g_2																
g_1																
g_0																
Stelle	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Diagramm 2: „Normaler“ Binär-Code

$d_3 = 2^3$																
$d_2 = 2^2$																
$d_1 = 2^1$																
$d_0 = 2^0$																
Stelle	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15



4

a) $360^\circ : 16 = 22.5$

b) tetradesischer, einschaltiger, starker Code

τ	g_3	g_2	g_1	g_0	d_3	d_2	d_1	d_0	d
0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	1
2	0	0	1	1	0	0	1	1	1
3	0	0	1	0	0	0	1	0	1
4	0	1	1	0	0	1	1	1	1
5	0	1	1	1	0	1	1	0	1
6	0	1	0	1	0	1	0	0	1
7	0	1	0	0	0	1	0	1	1
8	1	1	0	0	1	1	1	1	1
9	1	1	0	1	1	1	1	0	1
10	1	1	1	1	1	1	1	0	0
11	1	1	1	1	0	1	1	0	1
12	1	0	1	0	1	0	0	0	0
13	1	0	1	1	1	0	0	1	1
14	1	0	0	1	1	0	1	1	1
15	1	0	0	0	1	0	1	0	1

d)

$$d_2 = g_3$$

1	1	1	1
1	1	1	1

 g_3

e)

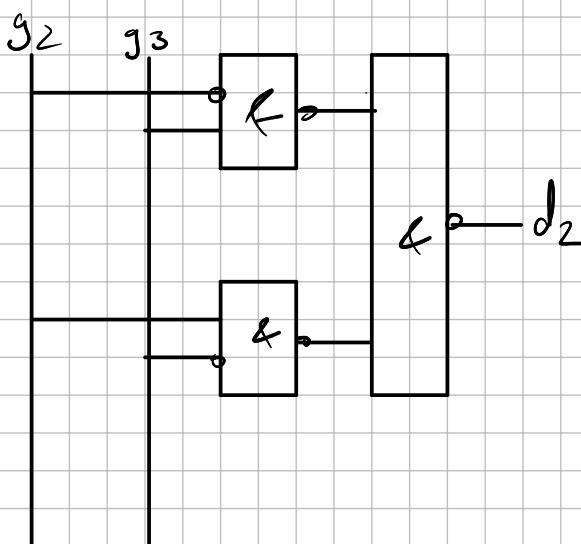
		(1 1)	
1			1
1			1
	1		1
		1 1	

 g_1 g_3 g_2

$$d_2 = \overline{(g_2 \wedge g_3)} \vee \overline{(g_2 \wedge \bar{g}_3)}$$

$$d_2 = \overline{(g_2 \wedge g_3)} \vee \overline{(g_2 \wedge \bar{g}_3)}$$

$$d_2 = \overline{\overline{(g_2 \vee g_3)} \wedge \overline{(g_2 \wedge \bar{g}_3)}}$$



d_0	d_1	d_2	d_3
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

Aufgabe 1: Pseudotetradenerkennung

a) Vgl. Infoblatt;

BCD-Code = Binary Coded Decimal, also dualkodierte Dezimalziffer. Dabei wird jede dezimale Ziffer 0 bis 9 durch jeweils vier Bit dargestellt, d.h. er ist ein Ziffernkode (Tetraden; 0000 bis 1001). Er ist ein bewertbarer/gewichteter Code, jede Bitstelle hat eine Stellen-Wertigkeit. Die Bezeichnung 8-4-2-1-Code steht für die Wertigkeit der einzelnen Bits.

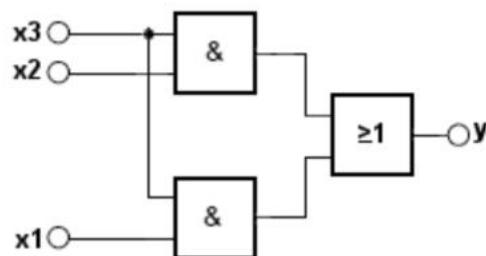
b) Wahrheitstabelle und Logikgleichung:

x_3	x_2	x_1	x_0	y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

x_3	x_2	x_1	x_0	$/x_0$	$/x_3$
0	0	0	0	0	1
1	1	1	1	1	0
0	1	1	1	0	0
0	0	0	0	0	1

$$y = (x_1 \wedge x_3) \vee (x_2 \wedge x_3)$$

Schaltungsdesign:



Aufgabe 2: Siebensegment-Dekoder

a)

Zeile	x_1	x_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	0
1	0	1	1	0	1	1	0	0	0
2	1	0	1	1	1	0	1	1	0
3	1	1	1	1	1	1	0	0	1

a x_0 $/x_0$

x ₁	1	1
$/x_1$		1

$$a = /x_0 \vee x_1$$

c x_0 $/x_0$

x ₁		1
$/x_1$	1	1

$$c = /x_0 \vee /x_1$$

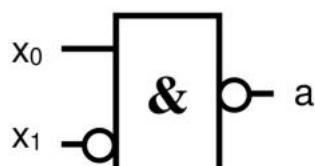
f x_0 $/x_0$

x ₁		
$/x_1$		1

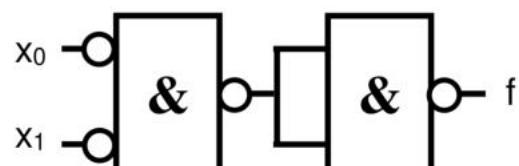
$$f = / /(/x_0 \wedge /x_1)$$

b)

$$a = / (x_0 \wedge /x_1)$$



$$f = //(/x_0 \wedge /x_1)$$



Aufgabe 3: 1-aus-10-Decoder

- a) Eine dezimale Ziffer wird im 1-aus-10-Code durch 10 Bits dargestellt, wobei jeweils nur ein Bit auf 1 gesetzt ist, während die restlichen 9 Bits 0 sind.

Ein **1-aus-10-Decoder** ist eine Schaltung mit 10 Ausgängen und 4 Eingängen. Der jeweils adressierte Ausgang geht dann auf High, wenn die Dualzahl A am Eingang der Nummer des betreffenden Ausgangs y_n entspricht. Die anderen Ausgänge werden dann nicht angesteuert und bleiben auf Low.

Lösungsvorschlag:

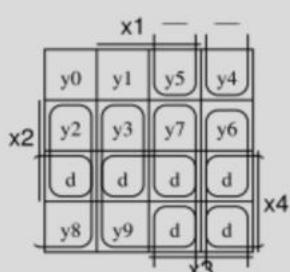
1-aus-10-Code: vier Steuereingänge, 10 Datenausgänge.

Eingangsvariablen: x_1, x_2, x_3, x_4

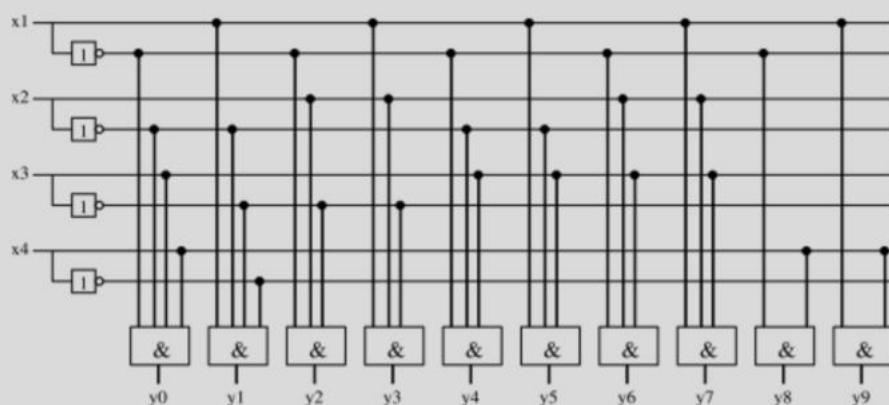
Ausgangsvariablen: $y_0, y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9$

Da mit 4 Eingängen 16 Ausgänge angesteuert werden können, aber nur 10 vorhanden sind, ergeben sich "don't care"-Terme d.

	x_4	x_3	x_2	x_1	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	1	0	0	0	0	0	0	0
3	0	0	1	1	0	0	0	1	0	0	0	0	0	0
4	0	1	0	0	0	0	0	0	1	0	0	0	0	0
5	0	1	0	1	0	0	0	0	0	1	0	0	0	0
6	0	1	1	0	0	0	0	0	0	0	0	1	0	0
7	0	1	1	1	0	0	0	0	0	0	0	0	1	0
8	1	0	0	0	0	0	0	0	0	0	0	0	1	0
9	1	0	0	1	0	0	0	0	0	0	0	0	0	1
10	1	0	1	0	d	d	d	d	d	d	d	d	d	d
11	1	0	1	1	d	d	d	d	d	d	d	d	d	d
12	1	1	0	0	d	d	d	d	d	d	d	d	d	d
13	1	1	0	1	d	d	d	d	d	d	d	d	d	d
14	1	1	1	0	d	d	d	d	d	d	d	d	d	d
15	1	1	1	1	d	d	d	d	d	d	d	d	d	d



$$\begin{aligned}
 y_0 &= \overline{x_1} \overline{x_2} \overline{x_3} \overline{x_4} \\
 y_1 &= x_1 \overline{x_2} \overline{x_3} \overline{x_4} \\
 y_2 &= \overline{x_1} x_2 \overline{x_3} \\
 y_3 &= x_1 x_2 \overline{x_3} \\
 y_4 &= \overline{x_1} \overline{x_2} x_3 \\
 y_5 &= x_1 \overline{x_2} x_3 \\
 y_6 &= \overline{x_1} x_2 x_3 \\
 y_7 &= x_1 x_2 x_3 \\
 y_8 &= \overline{x_1} x_4 \\
 y_9 &= x_1 x_4
 \end{aligned}$$



a)

	A	\bar{A}	
B	1 ₃ 1 ₇ 1 ₆ 1 ₂	1 ₁ 1 ₅ 1 ₁₀ 1 ₁₄	\bar{D}
\bar{B}	1 ₃ 1 ₅ 1 ₁₂ 1 ₇	1 ₁ 1 ₆ 1 ₁₀	D
	1 ₁ 1 ₆	1 ₄ 1 ₅	\bar{D}

$$\alpha = (B \wedge \bar{D}) \vee (\bar{A} \wedge \bar{C}) \vee (C \wedge B \wedge D) \vee (\bar{A} \wedge D \wedge C) \vee \\ (\bar{A} \wedge A \wedge \bar{C}) \vee (A \wedge C \wedge \bar{D})$$

b)

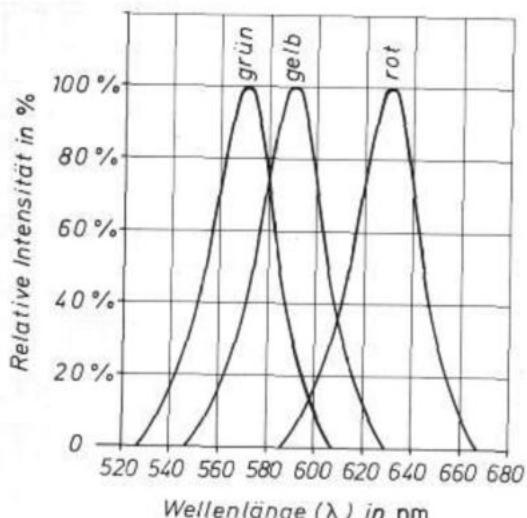
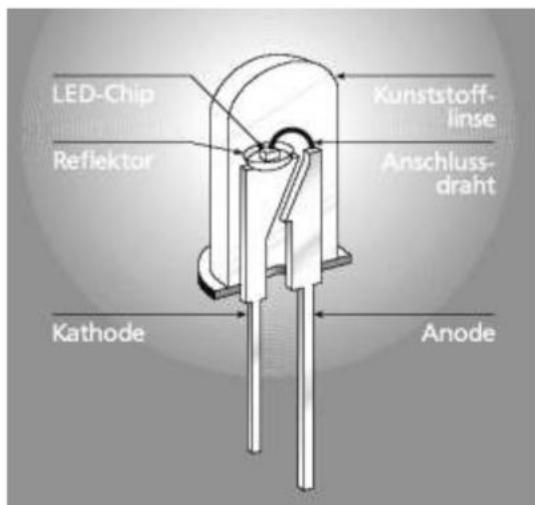
	A	\bar{A}	
B	1 ₃ 1 ₂	6 1 ₂	\bar{D}
\bar{B}	1 ₃ 1 ₅ 1 ₃ 1 ₈	1 ₁ 1 ₄	D
	1 ₁ 1 ₄	1 ₄ 1 ₅	\bar{D}

$$\beta) (\bar{A} \wedge \bar{C}) \vee (\bar{A} \wedge \bar{D} \wedge \bar{B}) \vee (A \wedge \bar{B} \wedge D) \vee \\ (A \wedge B \wedge \bar{C}) \vee (A \wedge B \wedge \bar{D})$$

c)

	A	\bar{A}	
B	3 7 6 2		\bar{D}
\bar{B}	1 ₁ 1 ₅ 1 ₄ 1 ₀		D
	1 1 ₃ 1 ₂ 8		\bar{D}
	1 1 ₄ 4 5		

LED (Datenblattauszüge)



Verteilung der Intensität auf die Wellenlänge

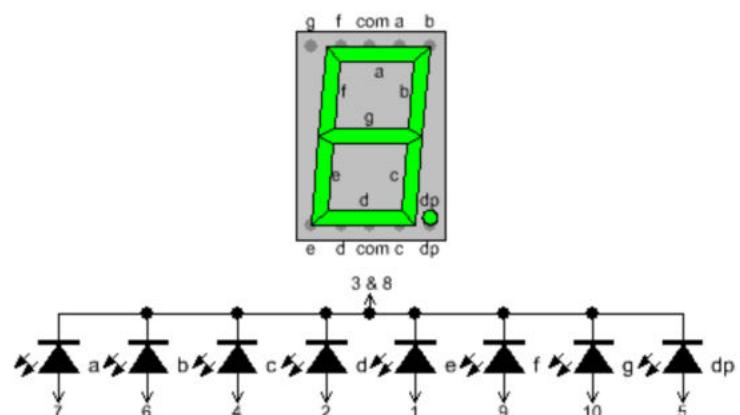
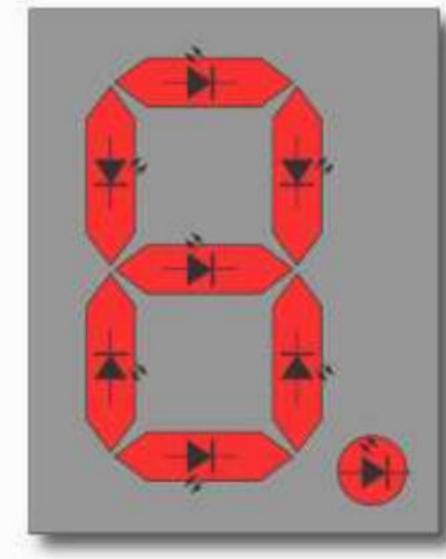
Herstellerangaben einer roten Leuchtdiode:

Maximaler Durchlassstrom $I_F = 100 \text{ mA}$ bei 25°C
Sperrspannung $U_R = 5 \text{ V}$



Sieben-Segment-Anzeige

Die Segmente werden meist aus Leuchtdioden (LEDs) oder Flüssig-Kristallanzeigen (LCDs) gebildet.



Aufgabe : Sieben-Segment-Anzeige

Ziel: Dieser Codeumsetzer soll den 4-Bit BCD-Code in eine direkt lesbare Dezimalziffer umsetzen, die aus sieben einzelnen Segmenten gebildet ist.

Die Anzeige erfolgt nach Tabelle 1.

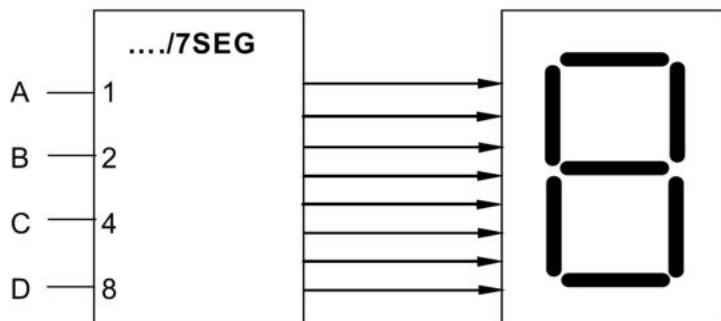
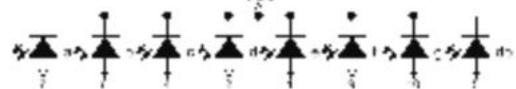
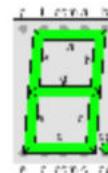
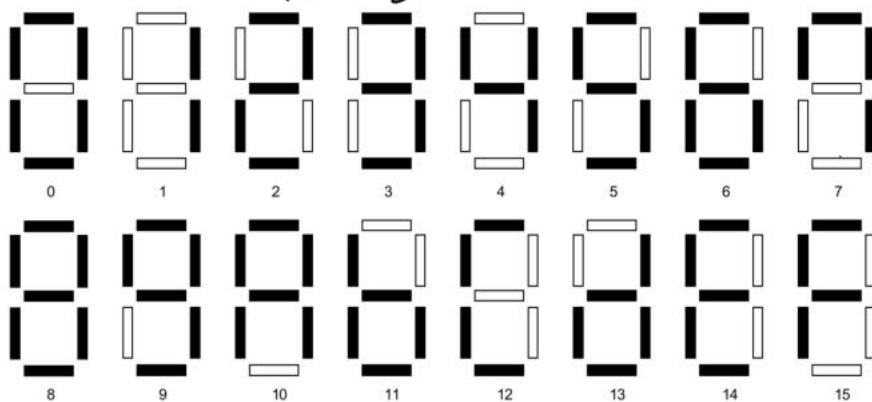
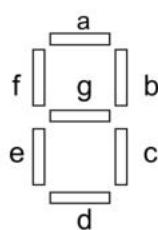


Tabelle 1 : Darstellung der Zahlen im 7-Segmentcode:

a b d e g



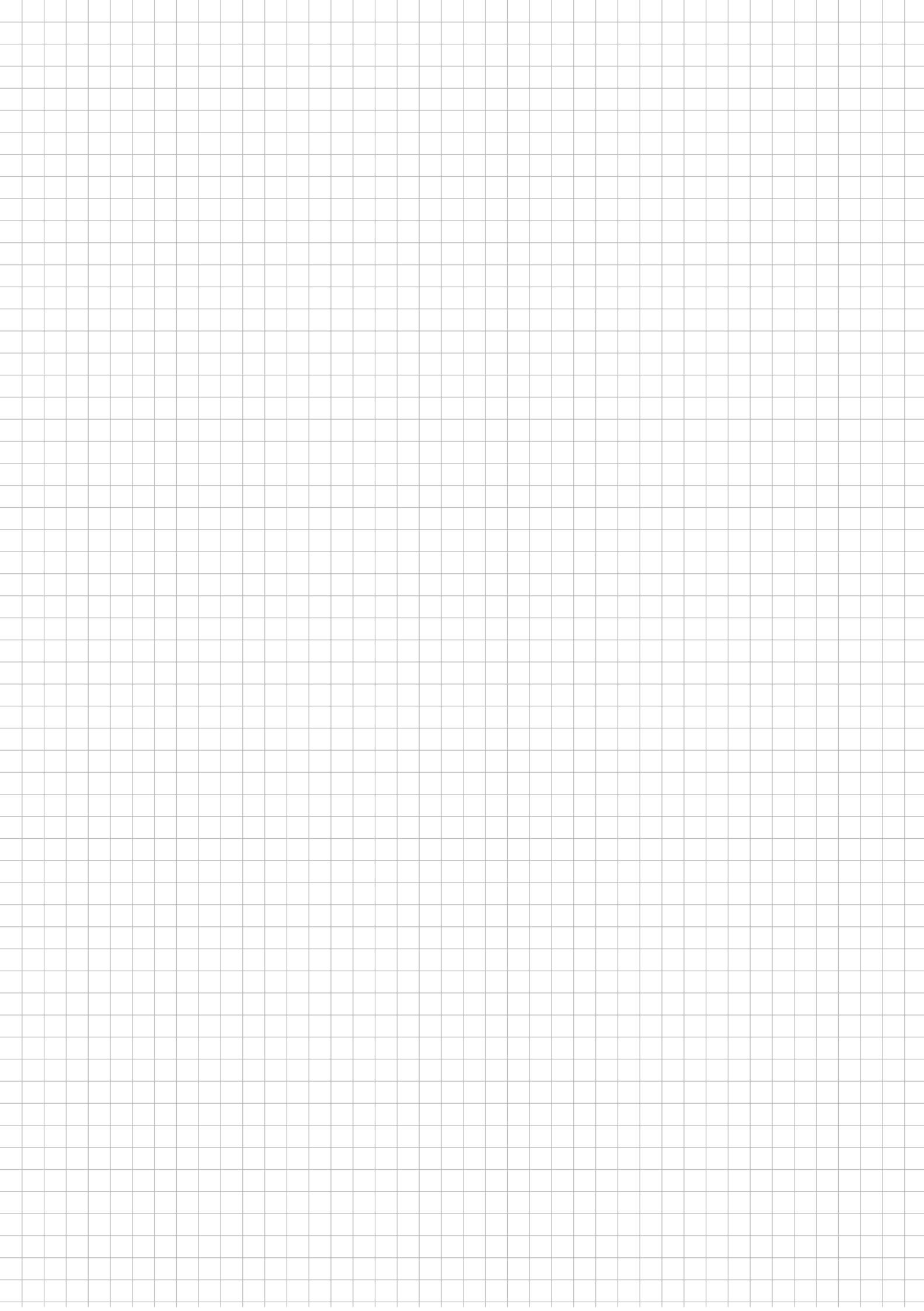
Aufgabenteil:

Im ersten Teil sollen 4-bit-Dualzahlen hexadezimal (0 - F) über die Siebensegmentanzeige ausgegeben werden.

- Welche Segmente leuchten auf, wenn DCBA = 0111 anlegt ? *ab cf*
- Erstellen Sie die vollständige Wertetabelle des Codewandlers und geben Sie die minimalisierten Schaltfunktionen für die Elemente a bis g an.
- Aufbau und Test für die Elemente e und f nach Teilaufgabe b).

Im zweiten Teil sollen die Zustände A - F_{hex} nicht mehr angezeigt werden.

- Welche minimalisierten Schaltfunktionen ergeben sich nun für die Elemente e und f? Aufbau in Full-Nand mit 7400 und Test.
- In integrierter Form gibt es den Anzeigetreiber 7448 mit ähnlichen Steuersignalen (A – F verschieden) für eine 7-Segment-Anzeige. Beschalten Sie dieses IC nach Datenblatt und steuern Sie damit die Elemente der 7-Segment-Anzeige des Digiboards an.



8 4 2 1

b)

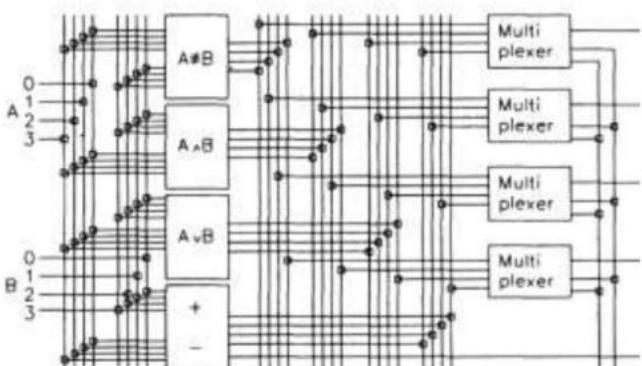
	D	C	B	A	g	f	e	d	c	b	a
0	0	0	0	0	0	1	1	1	1	1	1
1	0	0	0	1	0	0	0	0	1	1	0
2	G	0	1	0	1	0	1	1	0	1	1
3	0	0	1	1	1	0	0	1	1	1	1
4	0	1	0	0	1	1	0	0	1	1	0
5	0	1	0	1	1	1	0	1	1	0	1
6	0	1	1	0	1	1	1	1	1	0	1
7	0	-1	1	1	0	1	0	0	1	1	1
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	0	1	1	1	1
10	1	0	1	0	1	1	1	0	1	1	1
11	1	0	1	1	1	1	1	1	1	0	0
12	1	1	0	0	0	1	1	1	0	0	1
13	1	1	0	1	1	0	1	1	1	1	0
14	1	1	1	0	1	1	1	1	0	0	1
15	1	1	1	1	1	1	1	0	0	0	1

3	4	6	2
11	16	14	10
5	13	12	9
1	8	4	2

Programmierbare Schaltnetze – Arithmetisch-logische-Recheneinheit

Bei programmierbaren Schaltnetzen sollte man hier nicht an Mikroprozessoren denken, welche keine Schaltnetze, sondern sehr komplexe Schaltwerke darstellen, da der Ablauf von einem Takt gesteuert wird und damit der Zustand der Schaltung zeitlich abhängig ist. Ein programmierbares Schaltnetz ist vielmehr eine Schaltung, deren *Eigenschaften durch Steuereingänge veränderbar* sind.

Solche Schaltungen lassen sich beliebig entwerfen. Sie besitzen immer Signaleingänge, einen oder mehrere Ausgänge und einige Steuereingänge, die die Verknüpfungen bestimmen. Ein



Beispiel eines programmierbaren Schaltnetzes zeigt das nebenstehende Bild, eine ALU für vier Bit (Arithmetic Logic Unit). Mit diesem Schaltnetz lassen sich die beiden 4-Bit-Worte am Eingang arithmetisch oder logisch verknüpfen. Der Vorteil eines programmierbaren Schaltnetzes kann auch sein, dass das Netz als Baugruppe fertiggestellt und sogar vergossen werden kann. Die endgültige Funktion kann dann nachträglich durch die Steuereingänge festgelegt werden.

Aufgabe 1: 1-Bit-ALU

Ziel: Es soll die Schaltung einer ALU entwickelt werden, die wahlweise vier Operationen ausführen kann.

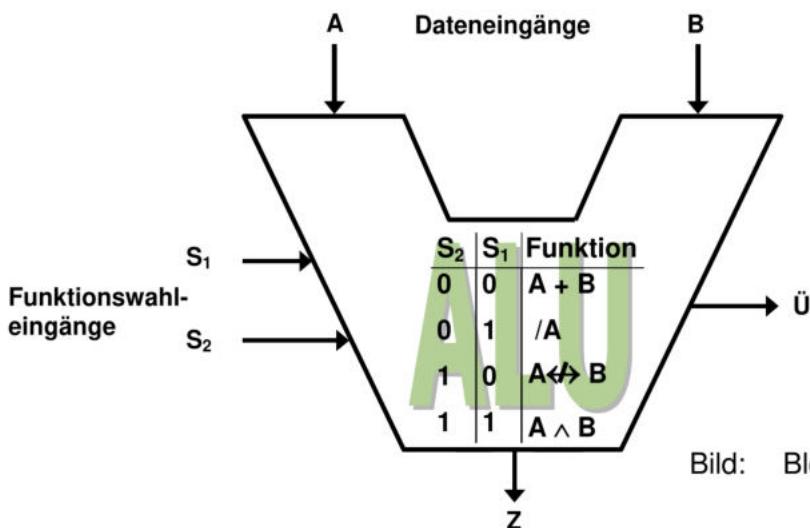


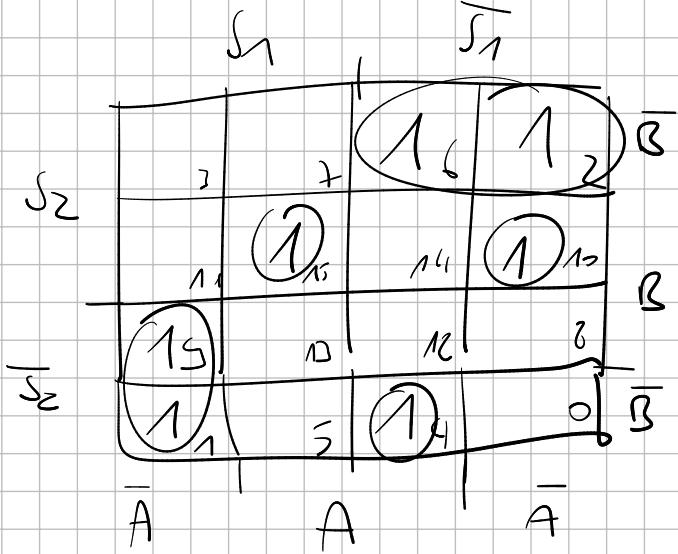
Bild: Blockschaltbild der Schaltung

- ~~Was versteht man unter einem programmierbaren Schaltnetz?~~
- Erklären Sie die Bedeutung von ALU.
Unterscheiden Sie dabei arithmetische von logischen Operationen.
Warum sind bei solchen Bausteinen „Funktionswahleingänge“ erforderlich?
Welche Bedeutung haben die Ausgänge Z und Ü?
- Entwickeln Sie eine einfache 1 Bit - Rechenschaltung (ALU), die wahlweise die im Blockschaltbild angegebenen Operationen ausführt:
 - Benennen Sie die 4 Operationen nach obigem Blockschaltbild der ALU.
 - Entwickeln Sie den Schaltplan.
 - Aufbau und Test Ihrer Schaltung.

- b) - ALU ist das elektronische Rechenwerk des CPU.
- arithmetische Operationen: $+ - \times \div$
- logische Operationen: und, oder, nicht u.s.w...
- damit der Chip darum arbeiten und die Eingänge verarbeiten kann.
- Z : Ergebnis

Ü: Ablauf

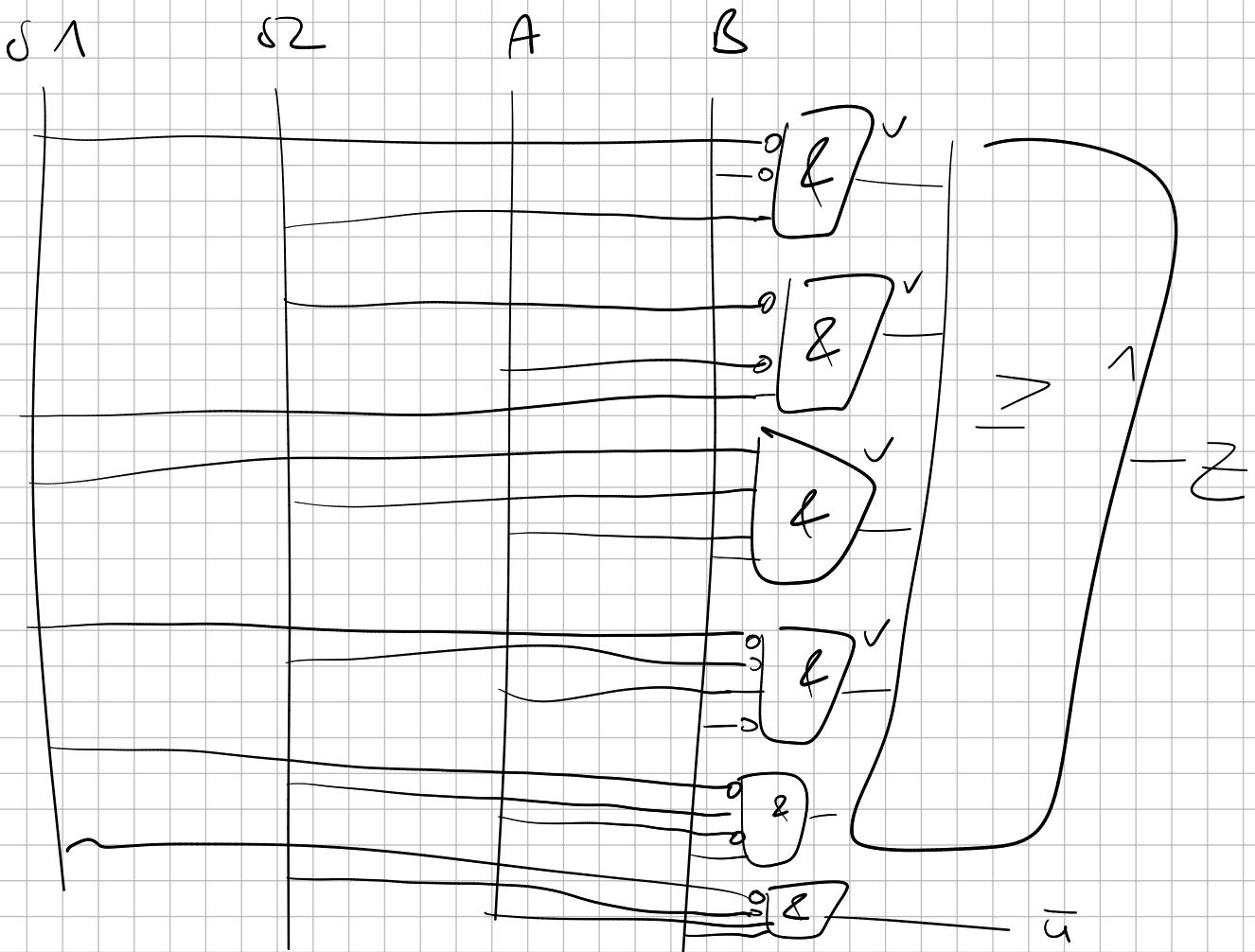
	S1	S2	A	B	Σ	Z	
0	0	0	0	0	0	0	Add. ✓
1	0	0	0	1	0	1 ✓	Add.
2	0	0	1	0	0	1 ✓	Add.
3	0	0	1	1	1	0 ✓	Add.
4	0	1	0	0	0	0 ✓	Inv.
5	0	1	0	1	0	0 ✓	Inv.
6	0	1	1	0	0	1 ✓	Inv.
7	0	1	1	1	0	0 ✓	Inv.
8	1	0	0	0	0	1 ✓	XOR
9	1	0	0	1	0	1 ✓	XOR
10	1	0	1	0	0	0 ✓	XOR
11	1	0	1	1	0	0 ✓	XOR
12	1	1	0	0	0	0 ✓	AND
13	1	1	0	1	0	0 ✓	AND
14	1	1	1	0	0	0 ✓	AND
15	1	1	1	1	0	1 ✓	AND

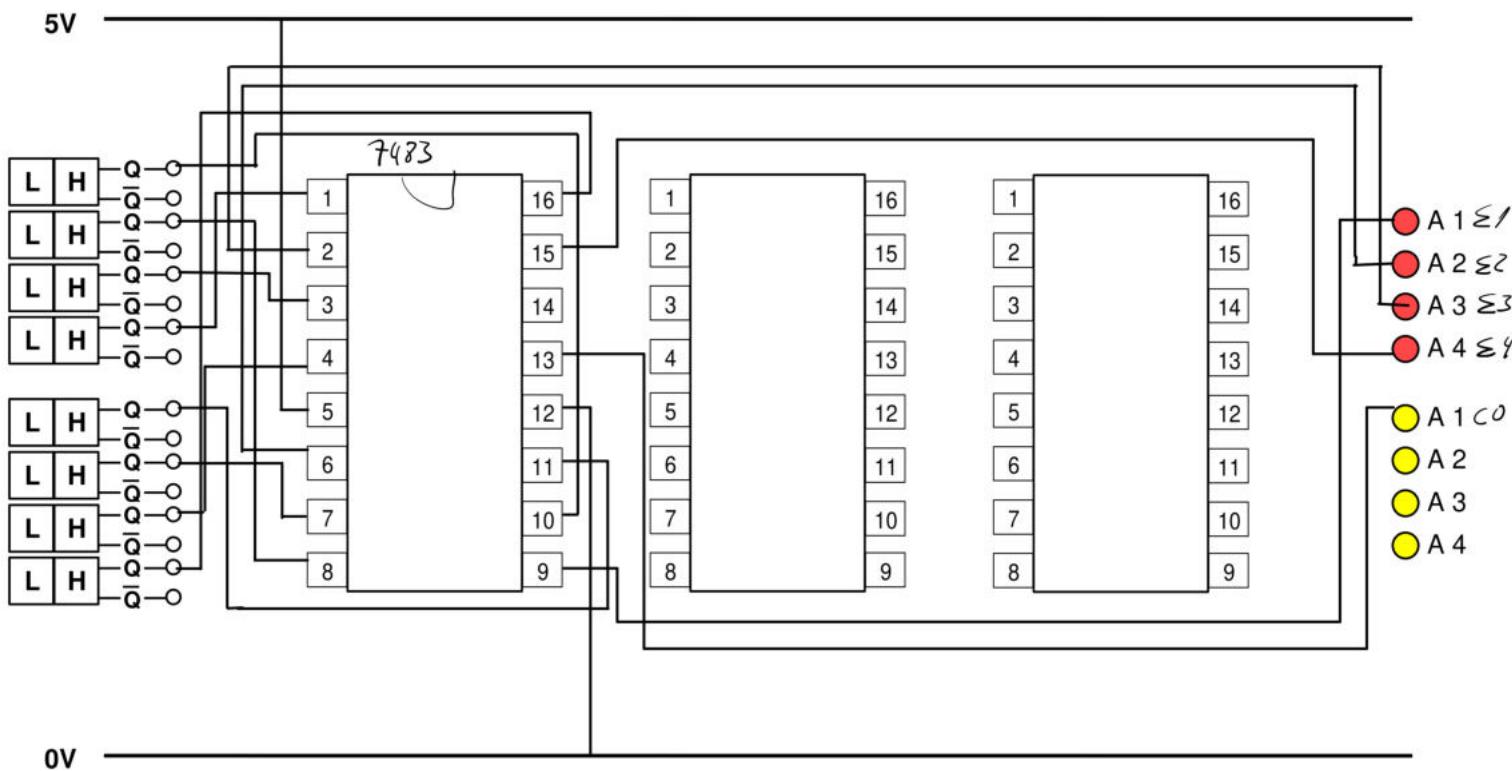


$Z =$

$$\begin{aligned}
 & (\overline{S_1} \wedge \overline{B} \wedge S_2) \vee \\
 & (\overline{S_2} \wedge \overline{A} \wedge S_1) \vee \\
 & (S_1 \wedge S_2 \wedge A \wedge B) \vee \\
 & (\overline{S_1} \wedge \overline{S_2} \wedge A \wedge \overline{B}) \vee \\
 & (\overline{S_1} \wedge S_2 \wedge \overline{A} \wedge B)
 \end{aligned}$$

$$\bar{u} = \overline{S_1} \wedge \overline{S_2} \wedge \overline{A} \wedge \overline{B}$$

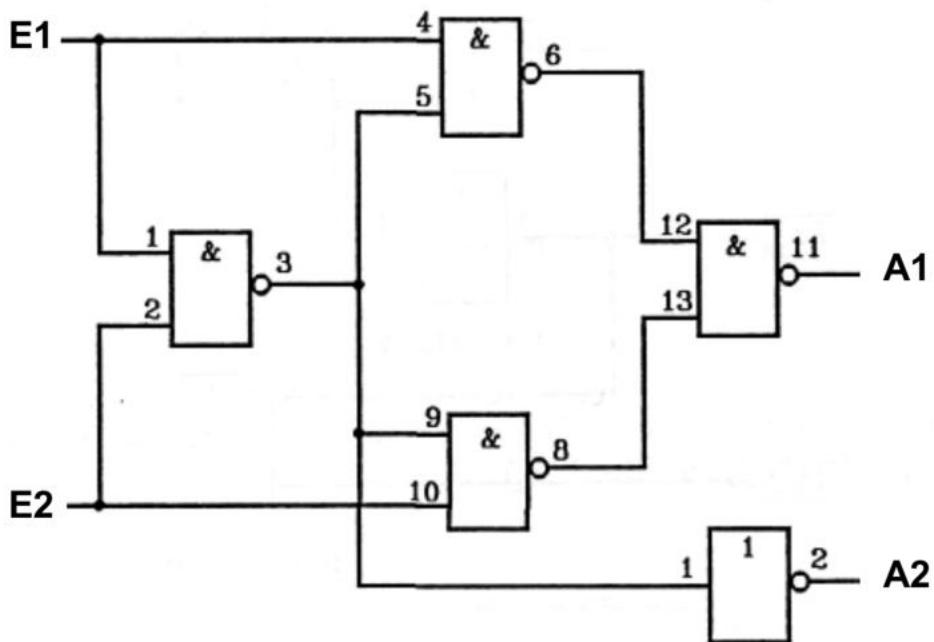




Aufgabe 1 : Halbaddierer

Ziel : Es sollen (Halb-)Addierschaltungen für zwei einstellige Dualzahlen aufgebaut und entworfen werden.

Stromlaufplan:



Aufgabenstellungen :

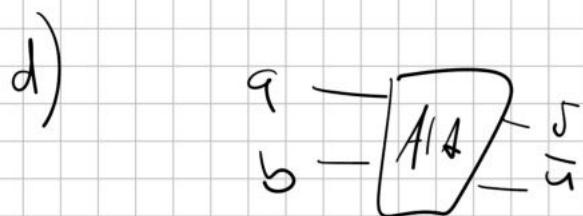
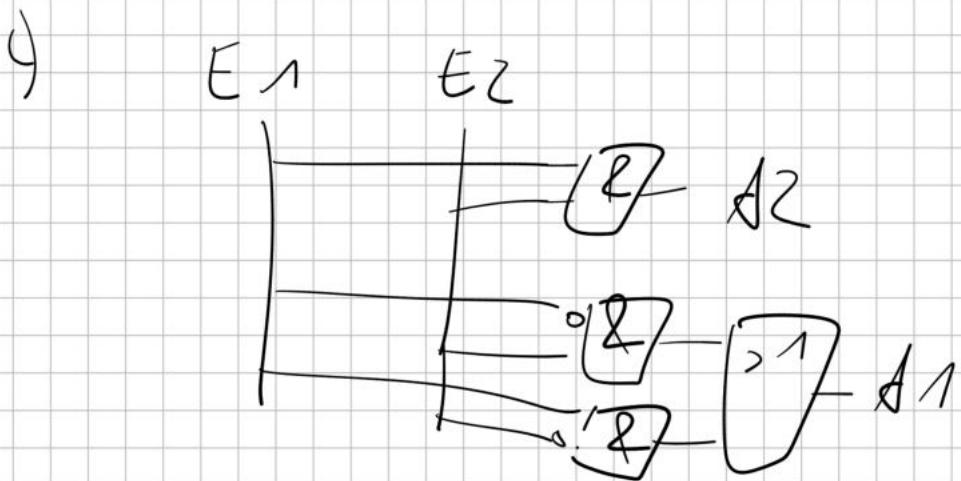
- Bauen Sie die Schaltung nach Schaltplan auf und erstellen Sie die Wertetabelle. Ordnen Sie die Ausgänge nach Summe und Übertrag.
Hinweise zum Aufbau:
Verwenden Sie die Bausteine 74LS00 und 74LS04 und verschalten Sie diese entsprechend dem Stromlaufplan.
- Geben Sie die minimierten Schaltfunktionen für A1 und A2 an.
- Entwerfen Sie für diese Funktionen einen Schaltplan und bauen Sie diese mit den Bausteinen 74LS86 und 74LS08 auf. Prüfen Sie deren richtige Funktion.
- Zeichnen Sie das Blockschaltbild eines Halbaddierers.
- Wie viele Bit lassen sich mit diesem Halbaddierer verarbeiten ?

a)

E_1	E_2	δ_1 / δ_2
0	0	0
0	1	0
1	0	0
1	1	1

b) $A_1 = (\bar{E}_1 \wedge E_2) \vee (E_1 \wedge \bar{E}_2)$

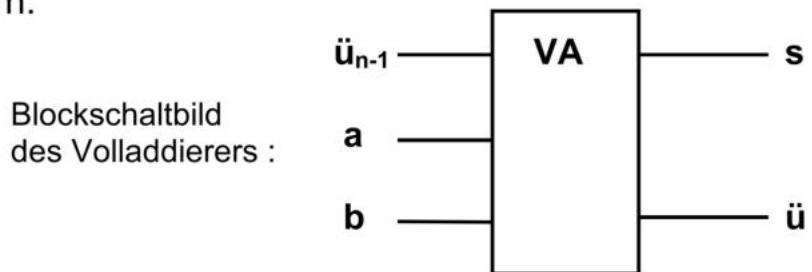
$A_2 = E_1 \wedge E_2$



e) 2 Bit

Aufgabe 2 : Volladdiererschaltung

Ziel : Addition von 3 Binärziffern.



Aufgabenstellungen :

- Erstellen Sie für einen Volladdierer VA (s.o.):
 - die vollständige Wertetabelle ,
 - die Funktionsgleichungen (minimalisieren Sie, soweit möglich) ,
 - ein Schaltnetz für s und ü.
 Bauen Sie auf und testen Sie die Funktion.
- Entwickeln Sie die VA-Schaltung aus zwei Halbaddierern (aus Afg.1) und einem Oder-Gatter, zeichnen Sie den Schaltplan, bauen Sie die Schaltung anschließend auf und testen Sie deren Funktion.
- Wodurch unterscheiden sich Halbaddierer und Volladdierer ?
Wie viele Bit lassen sich mit einem Volladdierer verarbeiten ?

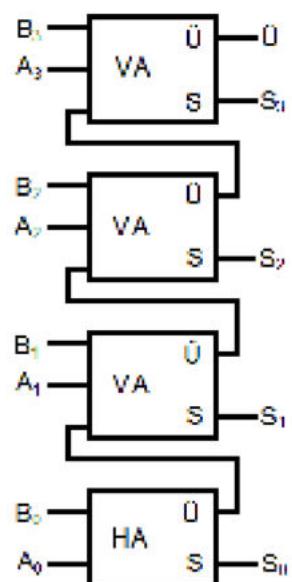
Aufgabe 3 : 4-Bit-Addierer

Der TTL-Baustein 7483 besteht aus 4 Volladdierern, die im IC zu einem 4-Bit-Addierer verschaltet sind.

Merkmale: Übertrags-Eingang C0 (Übertrag vorgehender Stelle)
Übertrags-Ausgang C4 (entstehender Übertrag der 4. Stelle).
Näheres entnehmen Sie bitte dem Datenblatt des TTL-Buches.

Achtung : Beachten Sie die Pin - Nummern der Spannungsversorgung!

- Zeichnen Sie den Verdrahtungsplan, beschaltet als 4-Bit-Addierer mit duality Anzeige des Ergebnisses.
- Aufbau, Test und dessen Dokumentation mit 3-4 selbstgewählten Beispielen.
- 4-Bit-Addierer mit dezimaler Anzeige:
Schalten Sie das IC zusätzlich an die dezimale Siebensegment - Anzeige.



(2)

	\bar{u}_{n-1}	a	b	s	\bar{u}
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

S:

	\bar{u}_{n-1}	$\bar{\bar{u}_{n-1}}$
9	3 1 ₂ 5 1 ₂ 1 ₂	
\bar{a}	1 ₁ 5 1 ₄ 0	

$\frac{1}{5} \quad \frac{1}{5} \quad \frac{1}{5}$

 \bar{u} :

	\bar{u}_{n-1}	$\bar{\bar{u}_{n-1}}$
9	1 ₂ 1 ₃ 1 ₂ 1 ₂ 2	
\bar{a}	1 1 ₅ 4 0	

$\frac{1}{5} \quad \frac{1}{5} \quad \frac{1}{5}$

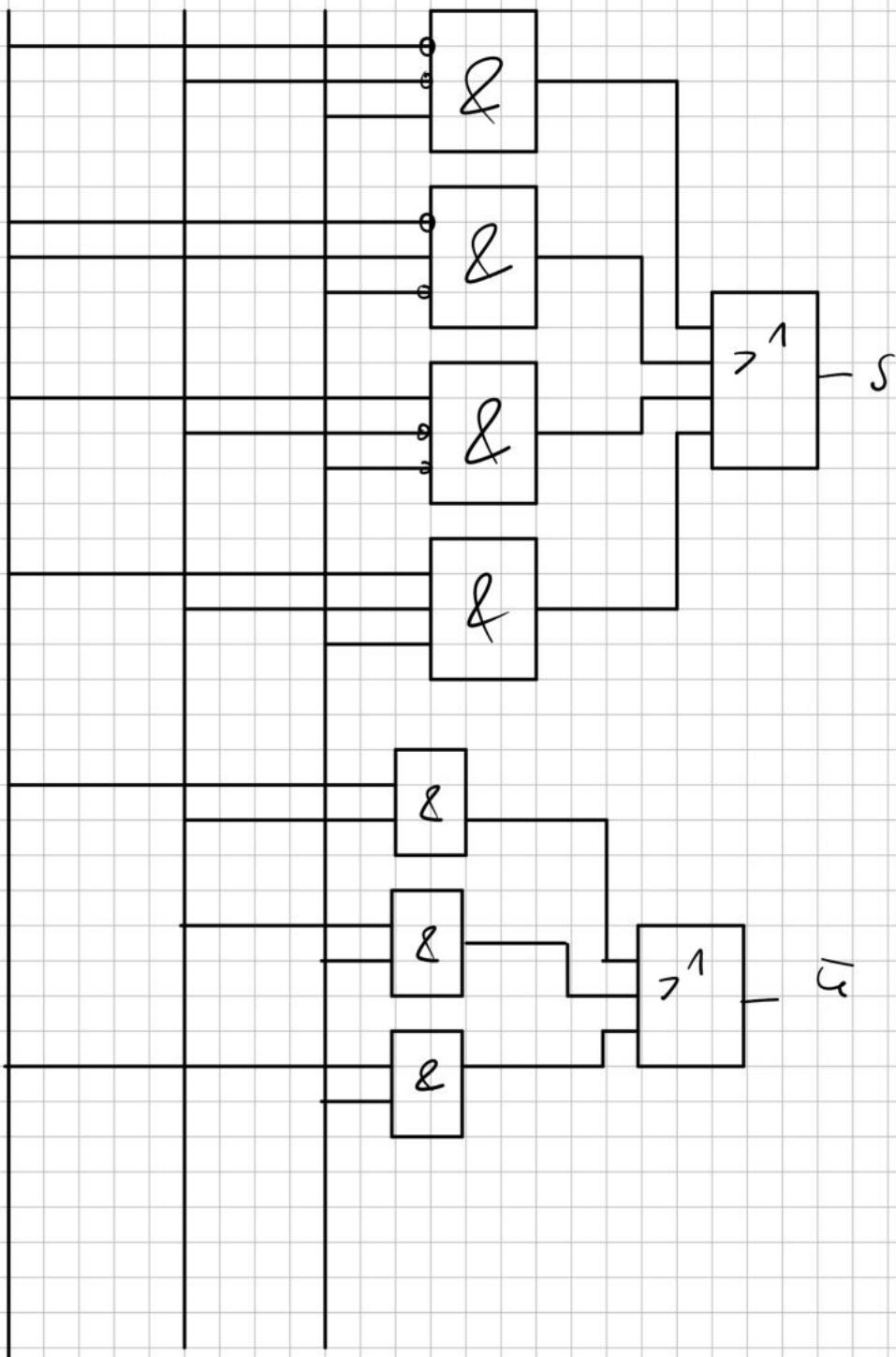
$$\begin{aligned} S &= \left((\bar{u}_{n-1} \wedge \bar{a} \wedge b) \vee \right. \\ &\quad \left(\bar{u}_{n-1} \wedge a \wedge \bar{b} \right) \vee \\ &\quad \left(\bar{u}_{n-1} \wedge \bar{a} \wedge \bar{b} \right) \vee \\ &\quad \left. (\bar{u}_{n-1} \wedge a \wedge b) \right) \end{aligned}$$

$$u = \bar{u}_{n-1} \vee (b \wedge a) \vee (\bar{u}_{n-1} \wedge b)$$

$$\bar{U}_{n-1}$$

a

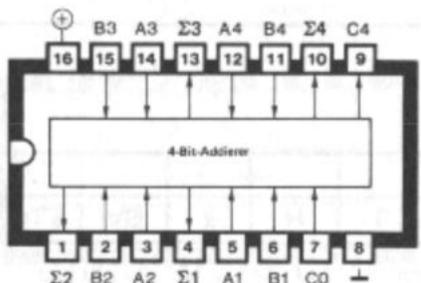
6



Aufgabe ③

4-Bit-Addier

a)


Beschreibung:

Dieser Baustein enthält einen Volladdierer, der die Summe zweier 4-Bit-Binärzahlen mit Übertrag liefert.

Betrieb:

Die Zahl A wird als der eine Eingang verwendet und folgendermaßen gewichtet: A1 = 1, A2 = 2, A3 = 4 und A4 = 8.

Die Zahl B wird als der andere Eingang verwendet und folgendermaßen gewichtet: B1 = 1, B2 = 2, B3 = 4 und B4 = 8.

Die Summe der beiden Zahlen steht dann an den Σ -Ausgängen, ebenfalls gewichtet $\Sigma_1 = 1$, $\Sigma_2 = 2$, $\Sigma_3 = 4$, $\Sigma_4 = 8$.

Wenn das Ergebnis dezimal 15 (binär 1111) überschreitet, erscheint eine 1 am Übertrags-Ausgang (Carry Output) an Anschluß C4.

Der Übertrags-Eingang C0 sollte an Masse gelegt werden, wenn nur 4-Bit-Zahlen verwendet werden.

Wenn es sich um die oberen 4 Bits einer 8-Bit-Zahl handelt, wird der Eingang C0 mit dem Ausgang C4 der vorhergehenden (niedrigstwertigen) Stufe verbunden.

Dieser Baustein ist funktionsmäßig identisch mit dem 7483, er besitzt nur eine andere Anschlußbelegung.

Anmerkung:

Die Eingangsbedingungen an A1, B1, A2, B2 und C0 werden für die Σ_1 , Σ_2 und den internen Übertrag C2 verwendet. Die Werte an C2, A3, B3, A4 und B4 werden dann für Σ_3 , Σ_4 und C4 verwendet.

Eingänge								Ausgänge											
								Wenn C0 = L				Wenn C0 = H							
								Wenn C2 = L		Wenn C2 = H		Wenn C2 = H		Wenn C2 = H					
A1	A3	B1	B3	A2	A4	B2	B4	I1	I3	I2	I4	C0	C4	I1	I3	I2	I4	C2	C4
L	L	L	L	L	L	L	L	L	L	L	L	H	L	L	L	L	L	L	
H	L	L	L	L	L	L	L	H	L	L	L	L	H	L	L	L	L	L	
L	H	L	L	L	L	L	L	H	L	L	L	L	H	L	L	L	L	L	
H	H	L	L	L	L	L	L	L	H	L	L	L	H	L	L	L	L	L	
L	L	H	L	L	L	L	L	L	H	L	L	L	H	L	L	L	L	L	
H	L	H	H	L	L	L	L	H	H	L	L	L	H	L	L	L	L	H	
L	H	H	H	L	L	L	L	H	H	L	L	L	H	L	L	L	L	H	
H	H	H	H	L	L	L	L	H	H	L	L	L	H	L	L	L	L	H	
L	L	H	H	H	H	L	L	L	L	H	H	L	H	L	H	L	H	H	
H	L	H	H	H	H	H	H	L	L	H	H	H	L	H	H	L	H	H	
L	H	H	H	H	H	H	H	L	L	H	H	H	L	H	H	L	H	H	
H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	

Anwendung:

Schnelle Binär-Additionen.

Daten:	Std	AS	F	LS	S
Typ. Additionszeit (für 8 Bits)	23	7	25	15	ns
Stromaufnahme	62	36	19	102	mA

Familien:	Std	ALS	AS	F	H	L	LS	S	
	●	●	●	●			●	●	

4-Bit-Volladdierer

74283

Aufgabe : Förderbandsteuerung

Ein Förderband wird mittels Hauptschalter S4 ein- und ausgeschaltet.

Die Lichtschranke S1 meldet evtl. vorhandene Pakete.

Normalerweise läuft der Motor ($M = 1$).

Ist ein Paket zu klein, wird dies durch die nicht unterbrochene Lichtschranke S2 gemeldet und der Motor M abgeschaltet.

Ist ein Paket zu groß, wird dies durch die unterbrochene Lichtschranke S3 gemeldet und der Motor M abgeschaltet.

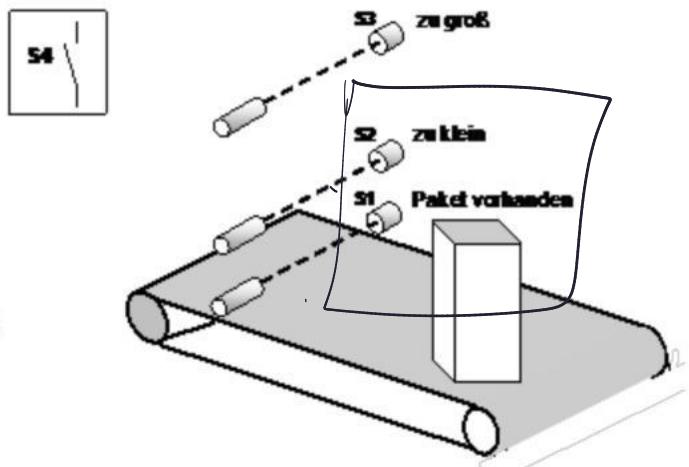


Einweg-Lichtschranke

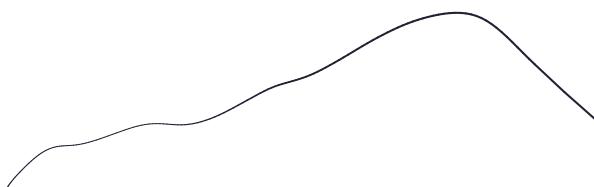


Unterbrechungslichtschranke:
Lichtstrahl unterbrochen $S = 0$

Technologieschema :



- Erstellen Sie die komplette Wertetabelle.
- Geben Sie die Funktionsgleichung (DMF) für den Motor M an.
- Testen Sie die Gleichung durch Schaltungsaufbau.
- Zeichnen Sie den Schaltplan nach b) in 7400-NAND-Technik auf.
- Aufbau und Test der Schaltung.



s_4	s_3	s_2	s_1	M
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

$\brace{M \text{ aus}}$

\rightarrow Palet zu graph

$\brace{\text{nicht m\"ogl.}}$

\rightarrow Palet in ordnung

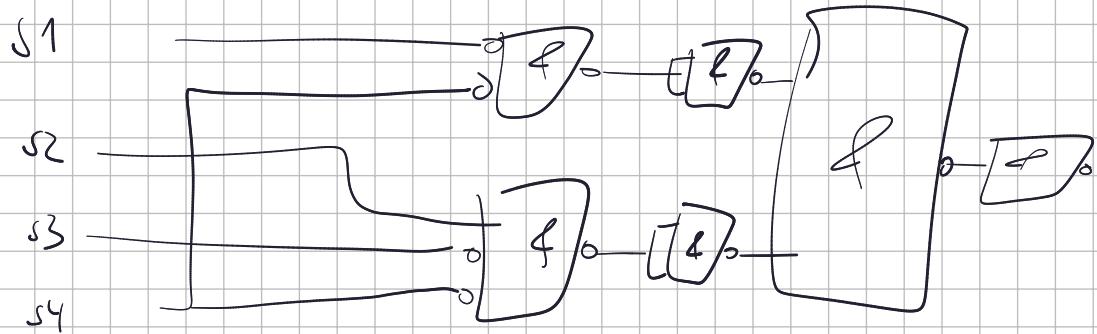
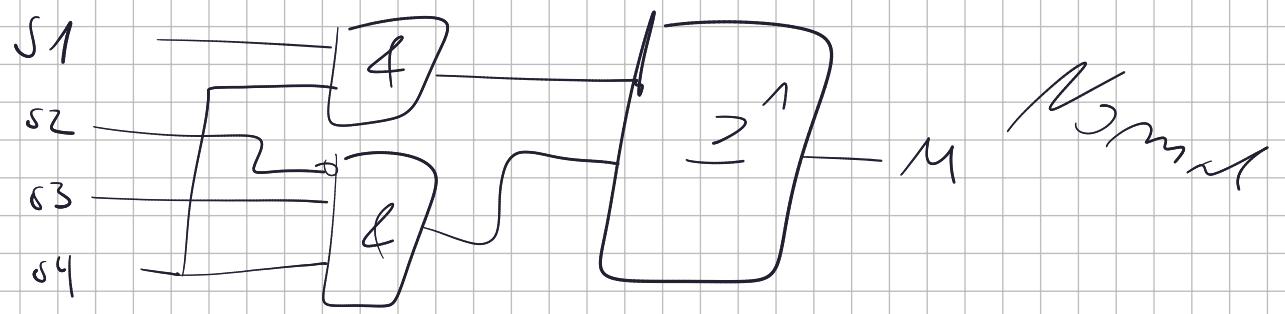
\rightarrow Palet zu klein

\rightarrow Kein Palet vorhanden

s_1	\bar{s}_1			
s_2	3	7	6	2
\bar{s}_2	x_{11}	1_{12}	0_{13}	x_{10}
\bar{s}_2	x_9	x_{13}	1_{12}	0_8
\bar{s}_3	1	5	4	9

$$DNF = (s_1 \wedge s_4) \vee (\bar{s}_2 \wedge s_3 \wedge s_4)$$

$$M = s_4 \cdot 1 [s_1 \vee (\bar{s}_2 \wedge s_3)]$$



Aufgabe 1 : 1 - Bit - Vergleicher

Ein **Komparator / Vergleicher** in der Digitaltechnik vergleicht zwei digitale Werte miteinander. Prüfen sie im einfachsten Fall nur die Gleichheit der Signale, bezeichnet man sie auch als Identitätskomparator. Wird zusätzlich noch zwischen größer oder kleiner unterschieden, dann handelt es sich um universell einsetzbare Magnitude- oder Größenkomparatoren.

Entwerfen Sie einen **1-Bit-Vergleicher**, der angibt, ob Signal a "gleich" Signal b ist.

- Erstellen Sie dazu die Wertetabelle. Wie lautet die DNF? Wie heisst diese Schaltung?
- Formen Sie die Gleichung mathematisch oder grafisch in die Full-NOR-Technik um und bauen Sie die Schaltung auf.

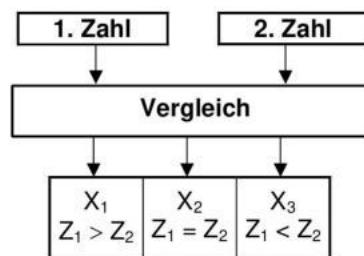
Aufgabe 2 : 2 - Bit - Vergleicher

Nun soll eine Schaltung entstehen, die zwei Dualzahlen Z_1, Z_2 mit jeweils 2 Stellen (= 2 Bit) miteinander vergleichen kann und entsprechend dem Ergebnis einen Ausgang X ansteuert.

1.Zahl : $Z_1 = A ; B$, wobei A den Stellenwert 2^0 , B den Stellenwert 2^1 hat.

2.Zahl : $Z_2 = C ; D$, wobei C den Stellenwert 2^0 , D den Stellenwert 2^1 hat.

Der Ausgang X_1 liefert den Wert 1, wenn die Zahl Z_1 größer als die Zahl Z_2 ist,
 der Ausgang X_2 liefert den Wert 1, wenn die Zahl Z_1 gleich der Zahl Z_2 ist,
 der Ausgang X_3 liefert den Wert 1, wenn die Zahl Z_1 kleiner als die Zahl Z_2 ist.



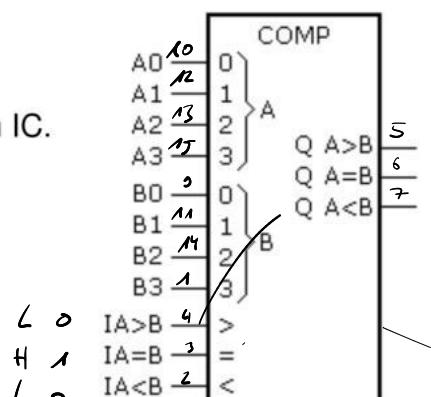
Nr.	2.Zahl		1.Zahl		Ausgänge		
	2^1 D	2^0 C	2^1 B	2^0 A	X_1 $Z_1 > Z_2$	X_2 $Z_1 = Z_2$	X_3 $Z_1 < Z_2$
0	0	0	0	0			
...			
15	1	1	1	1			

- Erstellen Sie die Funktionsgleichungen in minimierter Form (DMF).
- Zeichnen Sie den Logikplan (Schaltplan) aller Ausgänge.
- Bauen Sie die Schaltung für die Ausgänge X_1 und X_3 auf.
- Vereinfachen Sie X_2 , indem Sie die Ausgänge $X_{1,3}$ dafür mitbenutzen.
- Bauen Sie die Schaltung für $X_2 = f(X_1, X_3)$ dazu. Testen Sie die Schaltungen.

Aufgabe 3: Integrierte Vergleicherschaltung

Für diese Vergleicherschaltung gibt es auch schon einen fertigen IC.

- Wie ist dessen TTL-Nummer? 7485
- Beschalten Sie den Baustein und testen Sie.



①

a)	a	b	
	0	0	1
	0	1	0
	1	0	?

 $\leftarrow x \text{ nor}$

UND-Gatter

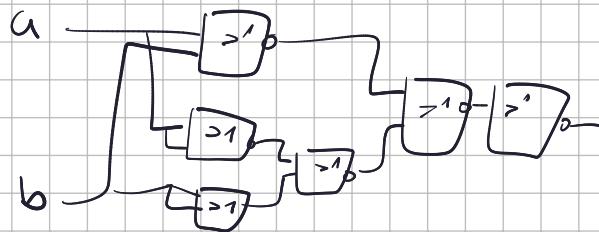
a 1 b

$$b) (\bar{a} \wedge \bar{b}) \vee (a \wedge b)$$

$$\overline{\overline{(\bar{a} \wedge \bar{b})}} \vee \overline{\overline{(a \wedge b)}}$$

$$\overline{\overline{(\bar{a} \wedge \bar{b})}} \vee \overline{\overline{(a \wedge b)}}$$

$$\overline{\overline{(\bar{a} \vee b)}} \vee \overline{\overline{(\bar{a} \vee b)}}$$



$$(B_1 \bar{D}) \vee (\bar{A}_1 B_1 \bar{C}) \vee (A_1 \bar{C}_1 \bar{D})$$

x₁:

	A	\bar{A}	
B	1	1	\bar{D}
\bar{B}	1	1	D
C	1	1	\bar{D}
\bar{C}	1	1	D

$$(A_1 B_1 C_1 \bar{D}) \vee (\bar{A}_1 B_1 D_1 \bar{C}) \vee$$

$$(A_1 B_1 \bar{C}_1 \bar{D}) \vee (\bar{A}_1 \bar{B}_1 C_1 \bar{D})$$

x₂:

	A	\bar{A}	
B	3	7	\bar{D}
\bar{B}	11	15	D
C	3	7	\bar{D}
\bar{C}	11	15	D

$$(\bar{B}_1 \bar{D}) \vee (\bar{A}_1 \bar{B}_1 \bar{C}) \vee$$

$$(\bar{A}_1 C_1 \bar{D})$$

x₃:

	A	\bar{A}	
B	3	7	\bar{D}
\bar{B}	11	15	D
C	3	7	\bar{D}
\bar{C}	11	15	D

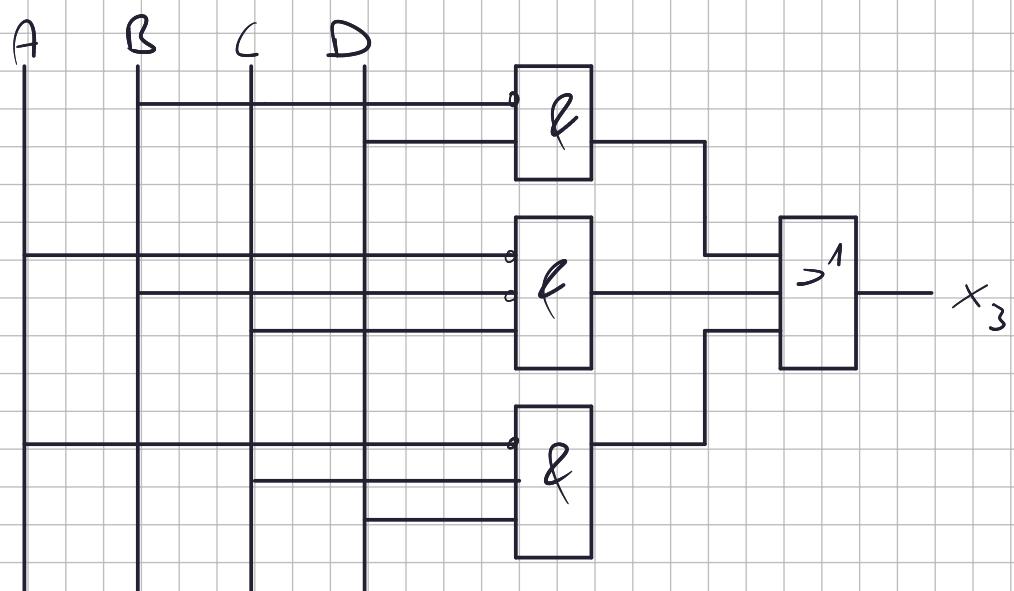
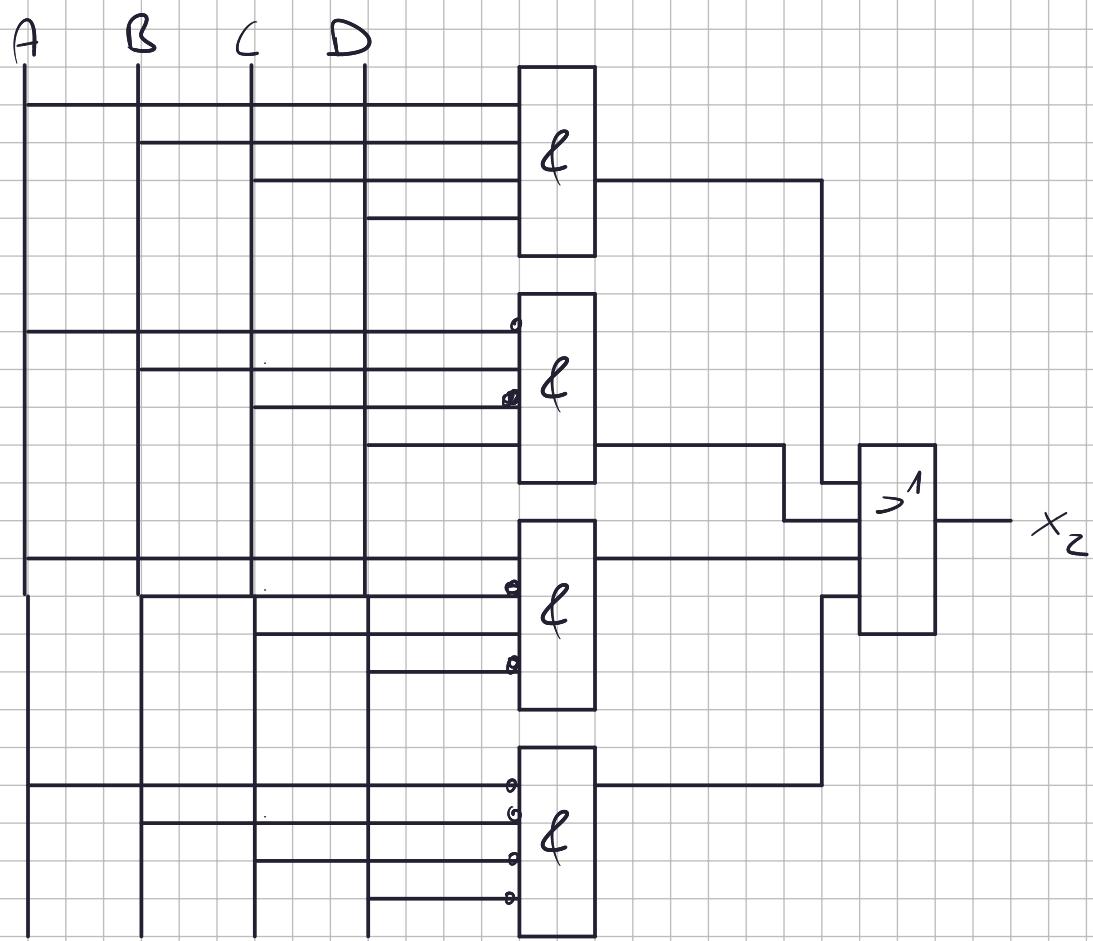
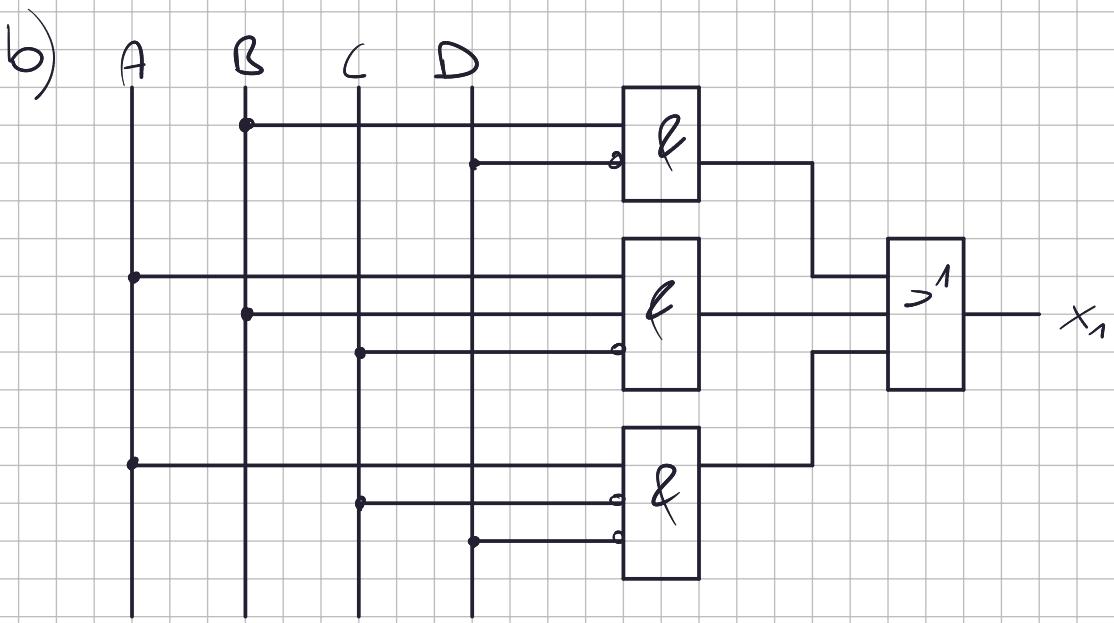
②

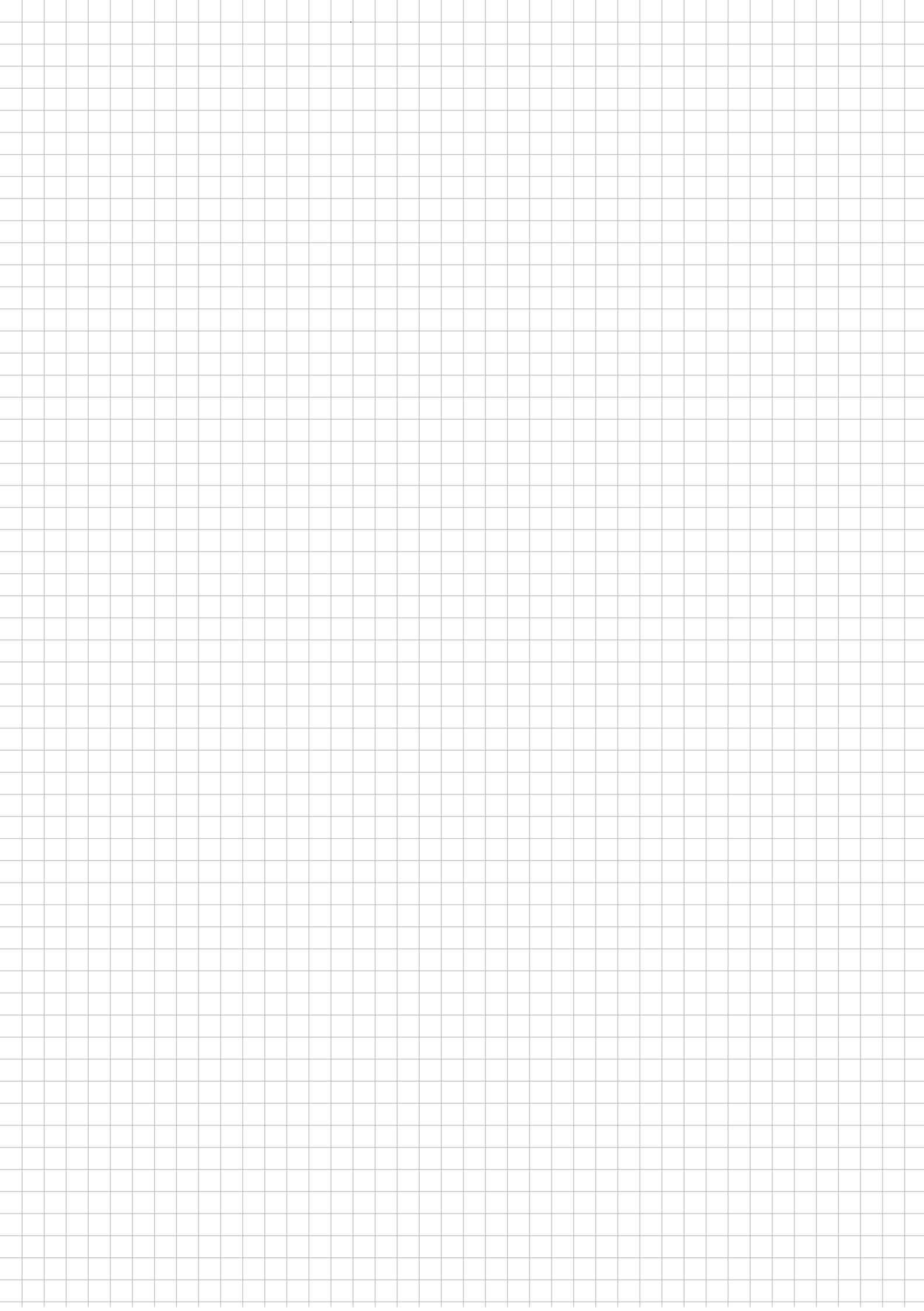
a)

	Z. Zahl				Ausgabe		
Nr.	D	C	B	A	x_1	x_2	x_3
0	0	0	0	0	0	1	0
1	0	0	0	1	1	0	0
2	0	0	1	0	1	0	0
3	0	0	1	1	1	0	0
4	0	1	0	0	0	0	1
5	0	1	0	1	0	1	0
6	0	1	1	0	1	0	0
7	0	1	1	1	1	0	0
8	1	0	0	0	0	0	1
9	1	0	0	1	0	0	1
10	1	0	1	0	0	1	0
11	1	0	1	1	1	0	0
12	1	1	0	0	0	0	1
13	1	1	0	1	0	0	1
14	1	1	1	0	0	0	1
15	1	1	1	1	0	1	0

x_1	x_2	x_3
$\bar{x}_1 \wedge x_2$	$\bar{x}_1 \wedge \bar{x}_2$	$x_1 = x_2$
0	0	1
1	0	0
0	1	0
1	1	X

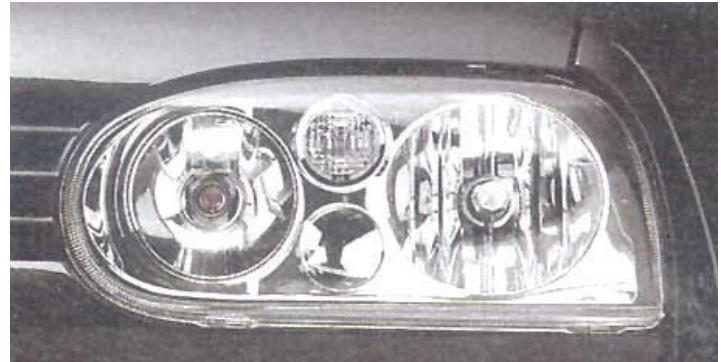
→ Nor bzw. Und mit negierten Eingängen (nach DE Morgan)





Aufgabe: Autobeleuchtung

Für Standlicht, Fahrlicht (Abblendlicht und Fernlicht) und Nebellampen ist eine Logikschaltung zu entwickeln, die den Vorschriften der StVZO (Strassenverkehrszulassungsordnung) genügt. Die Schalter und Eingangsvariablen sowie die Lampen und Ausgangsvariablen werden nach der folgenden Tabelle zugeordnet:



S1	Standlichtschalter	XS	Standlicht
S2	Fahrlichtschalter		
S3	Umschalter: C = 1 Abblendlicht; C = 0 Fernlicht	XA	Abblendlicht
S4	Nebellampenschalter	XF	Fernlicht
		XN	Nebellampen

Die StVZO schreibt vor:

1. Abblend- und Fernlicht dürfen nur eingeschaltet werden können, wenn auch das Standlicht leuchtet ($S1 = 1$).
2. Mit $S2 = 1$ kann das Fahrlicht gewählt werden,
3. mit $S3$ wird zwischen Abblend- und Fernlicht umgeschaltet.
4. Die Nebellampen dürfen nur in Verbindung mit dem Abblendlicht leuchten.

Aufgabenteil:

- a) Ermitteln Sie die minimierten Funktionsgleichungen (DMF). ✓
- b) Zeichnen Sie den gesamten Schaltplan. ✓
- c) Vereinfachen Sie den Schaltplan weiter, indem Sie die Ausgangsvariablen zur Verknüpfung mitverwenden. ✓
- d) Bauen Sie die Schaltung nach c) auf und testen Sie die Funktion. ✓
- e) Zeichnen Sie die Schaltung nach c) in Full-NOR, bauen Sie auf und testen Sie die Funktion.

D)

	S_4	S_3	S_2	S_1	X_S	X_A	X_T	X_N
0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0
2	0	0	1	0	0	0	0	0
3	0	0	1	1	1	0	1	0
4	0	1	0	0	0	0	0	0
5	0	1	0	1	1	0	0	0
6	0	1	1	0	0	0	0	0
7	0	1	1	1	1	1	0	0
8	1	0	0	0	0	0	0	0
9	1	0	0	1	1	0	0	0
10	1	0	1	0	0	0	0	0
11	1	0	1	1	1	0	1	0
12	1	1	0	0	0	0	0	0
13	1	1	0	1	1	0	0	0
14	1	1	1	0	0	0	0	0
15	1	1	1	1	1	1	0	1

$$X_S = \begin{array}{c|cc|c} & S_1 & \bar{S}_1 \\ \hline S_2 & \begin{array}{|c|c|c|} \hline & 1 & 1 \\ \hline & 1 & 1 \\ \hline \end{array} & = & \bar{S}_4 \\ \hline \bar{S}_2 & \begin{array}{|c|c|c|} \hline & 1 & 1 \\ \hline & 1 & 1 \\ \hline \end{array} & = & S_4 \\ \hline \bar{\bar{S}}_2 & \begin{array}{|c|c|c|} \hline & 1 & 1 \\ \hline & 1 & 1 \\ \hline \end{array} & = & \bar{S}_4 \\ \hline \bar{S}_3 & \bar{S}_3 & \bar{S}_3 & \bar{S}_4 \end{array}$$

$$X_S = S_1$$

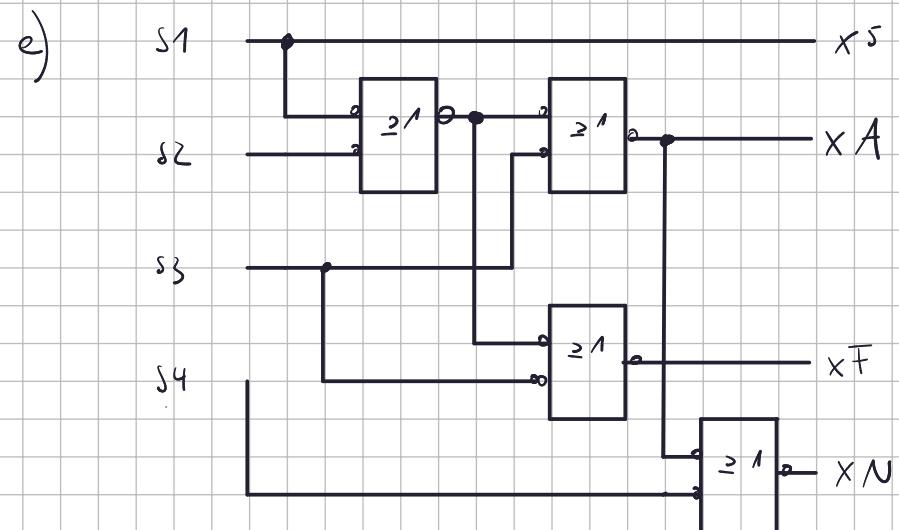
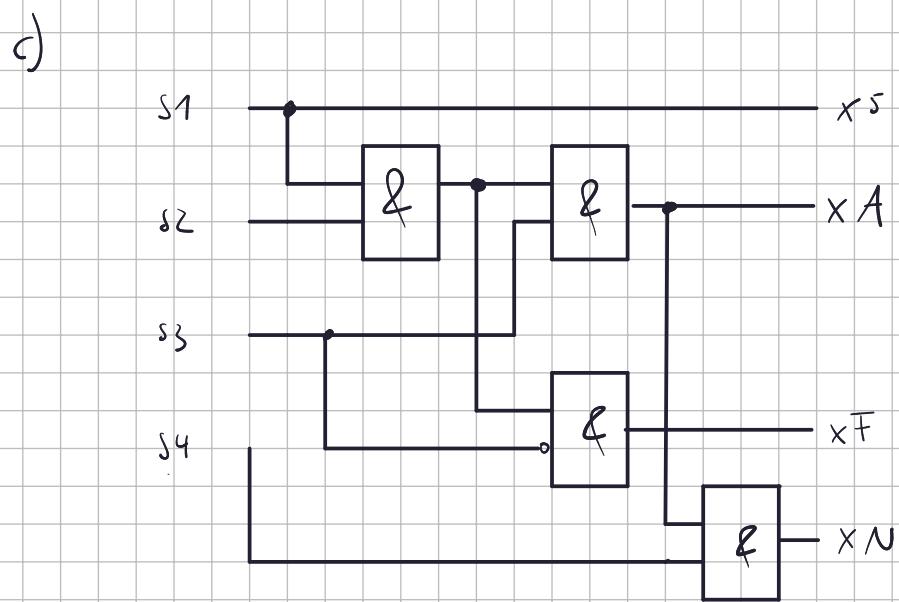
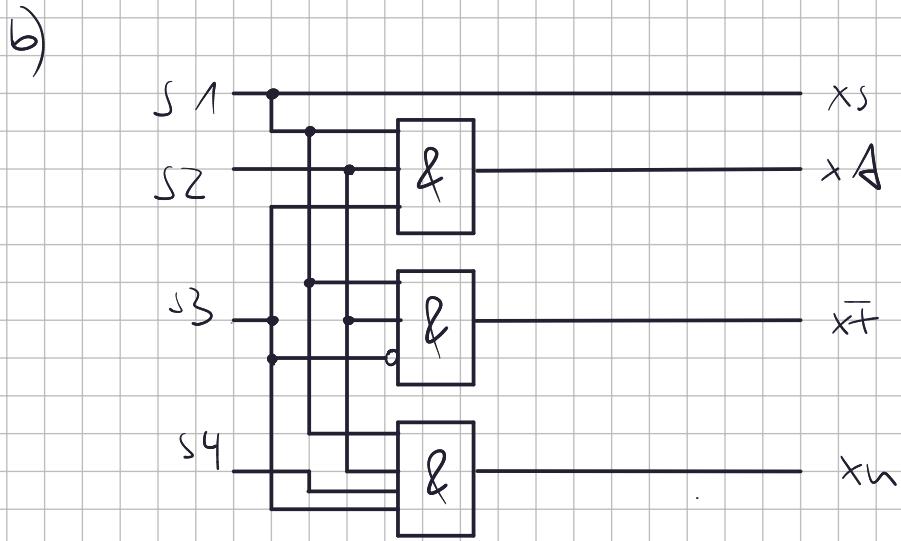
$$X_A = \begin{array}{c|cc|c} & S_1 & \bar{S}_1 \\ \hline S_2 & \begin{array}{|c|c|c|} \hline & 1 & 1 \\ \hline & 1 & 1 \\ \hline \end{array} & = & \bar{S}_4 \\ \hline \bar{S}_2 & \begin{array}{|c|c|c|} \hline & 1 & 1 \\ \hline & 1 & 1 \\ \hline \end{array} & = & S_4 \\ \hline \bar{\bar{S}}_2 & \begin{array}{|c|c|c|} \hline & 1 & 1 \\ \hline & 1 & 1 \\ \hline \end{array} & = & \bar{S}_4 \\ \hline \bar{S}_3 & \bar{S}_3 & \bar{S}_3 & \bar{S}_4 \end{array}$$

$$X_A = S_1 \wedge S_2 \wedge S_3$$

$$X_T = \begin{array}{c|cc|c} & S_1 & \bar{S}_1 \\ \hline S_2 & \begin{array}{|c|c|c|} \hline & 1 & 1 \\ \hline & 1 & 1 \\ \hline \end{array} & = & \bar{S}_4 \\ \hline \bar{S}_2 & \begin{array}{|c|c|c|} \hline & 1 & 1 \\ \hline & 1 & 1 \\ \hline \end{array} & = & S_4 \\ \hline \bar{\bar{S}}_2 & \begin{array}{|c|c|c|} \hline & 1 & 1 \\ \hline & 1 & 1 \\ \hline \end{array} & = & \bar{S}_4 \\ \hline \bar{S}_3 & \bar{S}_3 & \bar{S}_3 & \bar{S}_4 \end{array}$$

$$X_T = S_1 \wedge S_2 \wedge \bar{S}_3$$

$$X_N = S_1 \wedge S_2 \wedge S_3 \wedge S_4$$

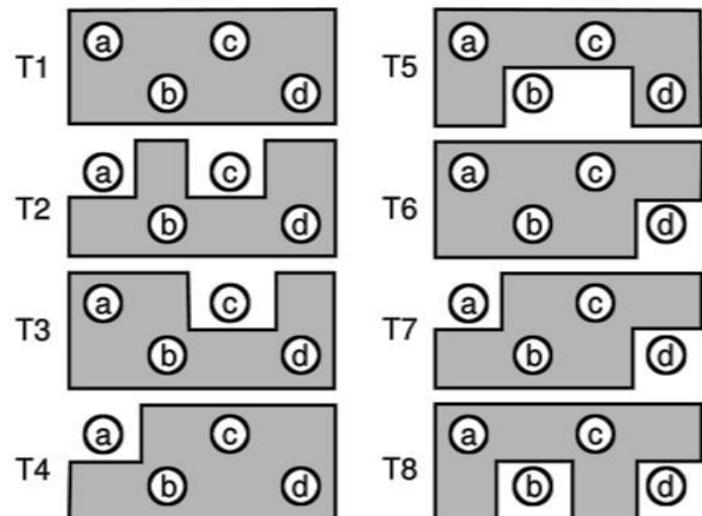


Aufgabe: Stanzteilelackierung

Eine Anlage muss verschiedene Stanzteile aus Blech (T1 bis T8) unterscheiden, um sie anschließend richtig behandeln zu können. Die Teile werden dabei von einem Förder-band unter eine Gruppe von vier Sensoren transportiert.

Die Sensoren (a, b, c, d) liefern an ihrem Ausgang den logischen Zustand '0', wenn darunter Blech festgestellt wird.

Die Teile T1, T3, T4, T5 und T7 müssen lackiert werden.



Es ist eine digitale Schaltung zu entwerfen, deren Ausgang die Lackiereinrichtung einschaltet ($y = 1$: Lackieren):

- a) Ermitteln Sie die minimierte Gleichung (DMF) und zeichnen Sie die Schaltung.
 - b) Entwerfen Sie die Schaltung mit ausschließlich NAND - Gattern.
 - c) Anschließen Aufbau und Test.