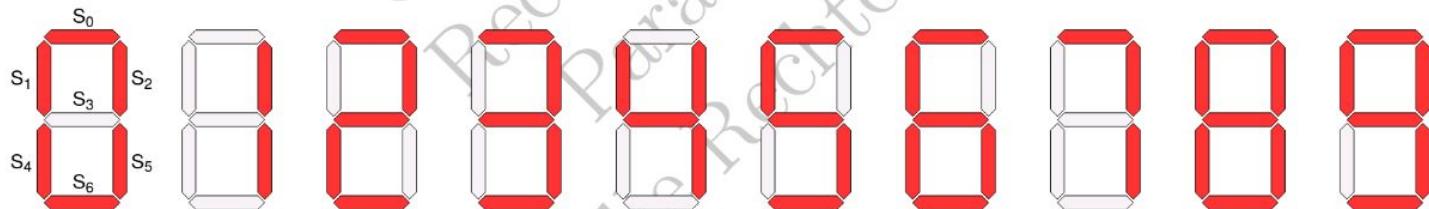




Willkommen zur
zwölften Woche der
ERA-Tutorials!

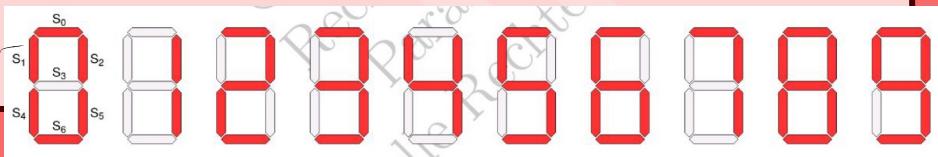


Entwirf einen Dekodierer, welcher eine 4-Bit Binärzahl $A = (a_3 \ a_2 \ a_1 \ a_0)_2$ als Eingabe erhält und diese entsprechend für die direkte Ansteuerung einer 7-Segmentanzeige (wie unten dargestellt) dekodiert. Dabei bedeutet eine 1 an einem Ausgang S_0, S_1, \dots, S_6 , dass das entsprechende LED-Segment leuchtet. Die Eingaben mit den Dezimalwerten 10 bis 15 werden niemals angelegt und sollen daher zur Minimierung mittels *Don't Care*-Werten verwendet werden.



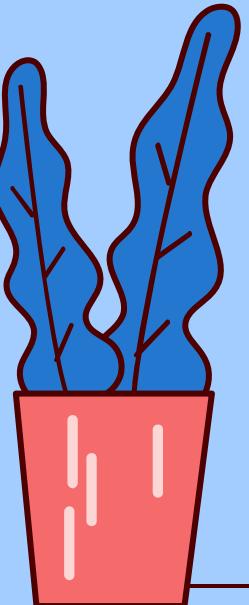
- a) Entwirf die Wahrheitstabelle für die Segmente S_0 bis S_6 .

Det.	a3	a2	a1	a0	S ₆	S ₅	S ₄	S ₃	S ₂	S ₁	S ₀
0	0	0	0	0	1	1	1	0	1	1	1
1	0	0	0	1	0	1	0	0	1	0	0
2	0	0	1	0	1	0	1	1	1	0	1
3	0	0	1	1	1	1	0	1	1	0	1
4	0	1	0	0	0	1	0	1	1	1	0
5	0	1	0	1	1	1	0	1	0	1	1
6	0	1	1	0	1	1	1	1	0	1	1
7	0	1	1	1	0	1	0	0	1	0	1
8	1	0	0	0	0	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	1	1	1
10	1	0	1	0	1	1	0	1	1	1	1
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	—	—	—	—	—	—	—



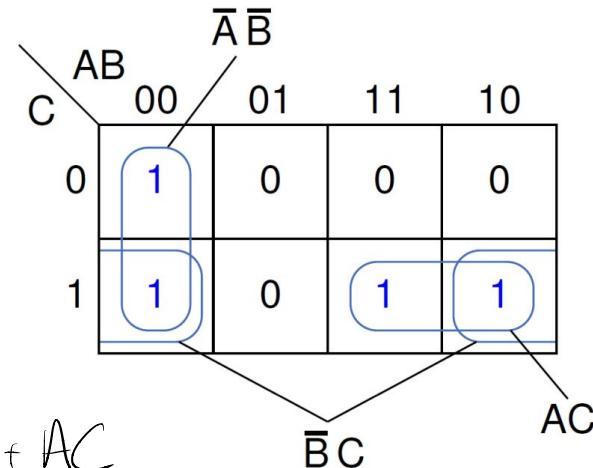
Karnaugh Maps

- Darstellung einer Funktion als 2-dimensionales Feld
- Einzelne Zellen geben den Funktionswert des jeweiligen Minterms an
- Benachbarte Zellen unterscheiden sich in ihren Koordinaten nur durch ein Bit* (dadurch lassen sich benachbarte Zellen mit gleichem Funktionswert kürzen)



		AB	00	01	11	10	
		C	0	$\bar{A}\bar{B}\bar{C}$	$\bar{A}B\bar{C}$	$A\bar{B}\bar{C}$	$A\bar{B}\bar{C}$
			1	$\bar{A}\bar{B}C$	$\bar{A}BC$	ABC	$A\bar{B}C$

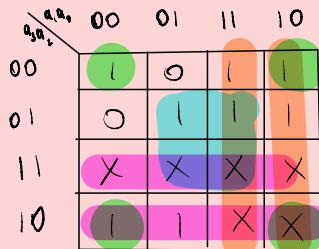
$$AC + \bar{A}\bar{B} \equiv \bar{A}\bar{B} + \bar{B}C + AC$$



- b) Minimiere die Boolesche Funktion für die Segmente S_0 , S_2 und S_6 mittels eines Karnaugh–Veitch Diagramms und gib die gefundenen Booleschen Funktionen an.

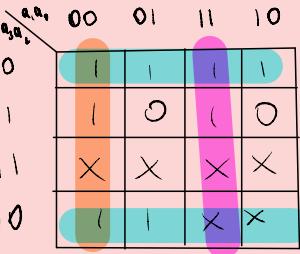
Dez.	a_3	a_2	a_1	a_0	S_6	S_5	S_4	S_3	S_2	S_1	S_0
0	0	0	0	0	1	1	1	0	1	1	1
1	0	0	0	1	0	1	0	0	1	0	0
2	0	0	1	0	1	0	1	1	1	0	1
3	0	0	1	1	1	1	0	1	1	0	1
4	0	1	0	0	0	1	0	1	1	1	0
5	0	1	0	1	1	1	0	1	0	1	1
6	0	1	1	0	1	1	1	1	0	1	1
7	0	1	1	1	0	1	0	0	1	0	1
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	0	1	1	1	1
10	1	0	1	0	-	-	-	-	-	-	-
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	-	-	-	-	-	-	-

$S_0:$



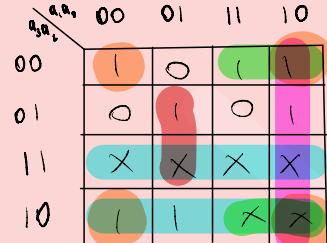
$$\bar{a}_0 \bar{a}_2 + a_1 + a_3 \\ + a_0 a_2$$

$S_2:$



$$\bar{a}_2 + \bar{a}_1 \bar{a}_0 \\ + a_1 a_0$$

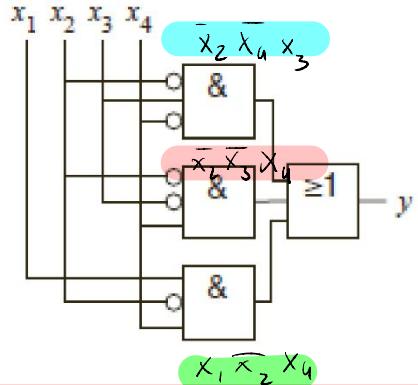
$S_6:$



$$\bar{a}_2 a_1 + a_2 \bar{a}_1 a_0 \\ a_3 + a_1 \bar{a}_0 + \bar{a}_2 \bar{a}_0$$

2 Logik-Hazards

Betrachte das folgende Schaltnetz.

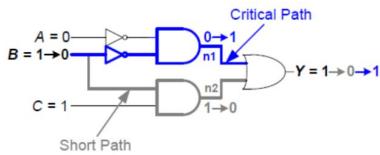
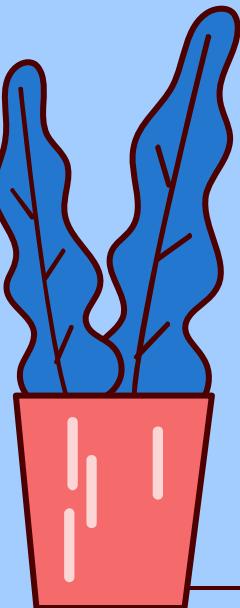


a) Bestimme das zugehörige KV-Diagramm.

$x_4 \backslash x_3$	00	01	11	10
00	0	0	0	0
01	1	1	0	0
11	0	1	0	0
10	1	1	0	0

Logik-Hazards

- Änderung eines Eingangs ändert kurzzeitig Ausgang, obwohl nach den Regeln der Booleschen Algebra keine Änderung auftritt
- Kann bei sequenziellen Schaltungen problematisch werden (z.B. Oszillation)



	AB	00	01	11	10
C	0	1	0	0	0
	1	1	1	1	0

$Y = \bar{A}\bar{B} + BC$

new Topic!
(Und AIGs sind
dieses Jahr nicht
Klausurrelevant!)

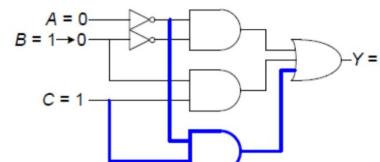


- Logik-Hazard kann(!) auftreten, wenn 2 benachbarte Eins-Felder nicht durch einen gemeinsamen Primblock abgedeckt werden

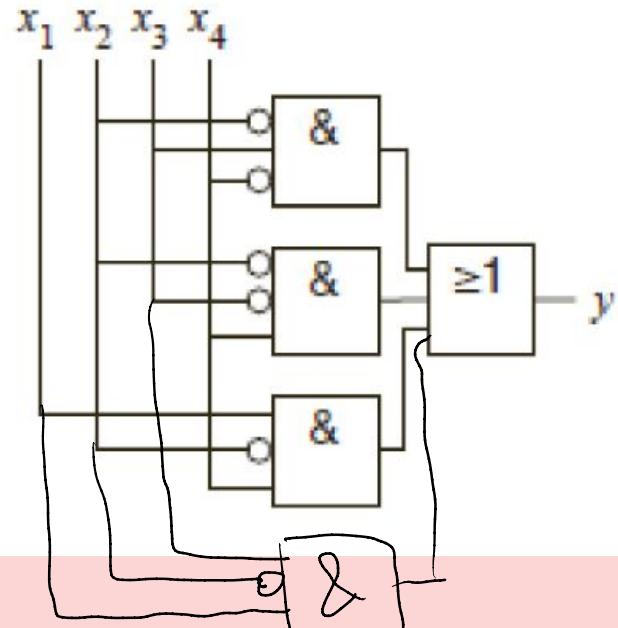
→ Behebung des Logik-Hazards durch hinzufügen eines weiteren Primblocks

	AB	00	01	11	10
C	0	1	0	0	0
	1	1	1	1	0

$Y = \bar{A}\bar{B} + BC + \bar{A}C$



b) Ist die Schaltung gegen Logik-Hazards abgesichert? Falls ja, warum? Falls nein, sicher das Schaltnetz gegen Logik-Hazards ab.

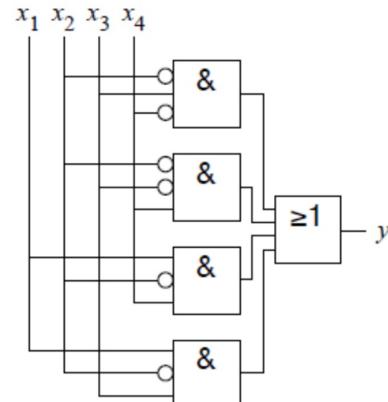


x_2x_1	$x_3\bar{x}_2x_1$			
x_4x_3	00	01	11	10
00	0	0	0	0
01	1	1	0	0
11	0	1	0	0
10	1	1	0	0

- b) Ist die Schaltung gegen Logik-Hazards abgesichert? Falls ja, warum? Falls nein, sicher das Schaltnetz gegen Logik-Hazards ab.

Die abgebildete Schaltung ist nicht vollständig gegen Logik-Hazards abgesichert, da die Einsmenge nicht überlappungsfrei überdeckt ist. Die Überlappungsfreiheit wird durch das Hinzufügen eines Blocks (Orange) erreicht:

x_4x_3	x_2x_1	00	01	11	10
00		0	0	0	0
01		1	1	0	0
11		0	1	0	0
10		1	1	0	0

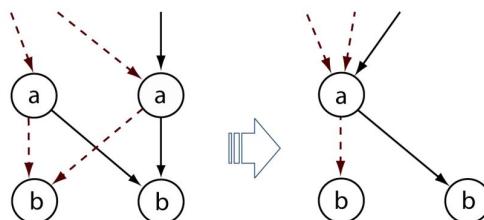


Binäres Entscheidungsdiagramm

- azyklischer Graph mit einer Wurzel
- Innere Knoten
 - sind mit einer Variablen markiert
 - haben zwei Nachfolger,
 - das low-Kind (hier: gestrichelte Kante) und
 - das high-Kind (hier: durchgezogene Kante)
- Blätter/Terminale Knoten
 - sind mit einer Konstanten (0 oder 1) markiert

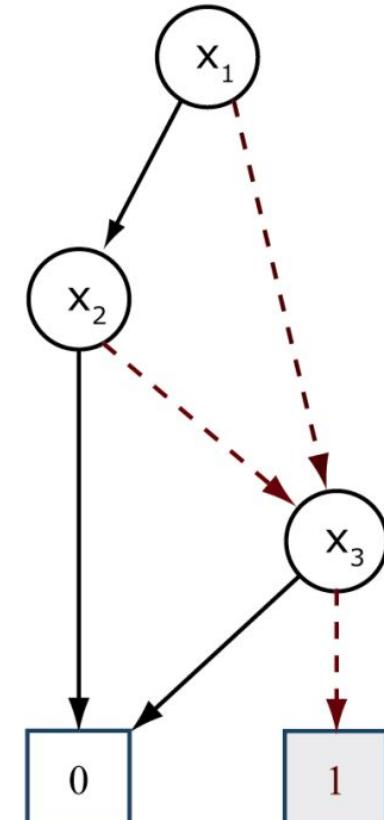
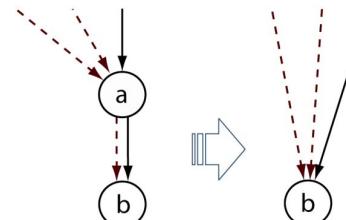
Regel 1: Isomorphismus

Führe isomorphe Knoten zusammen.
(I-Reduction)



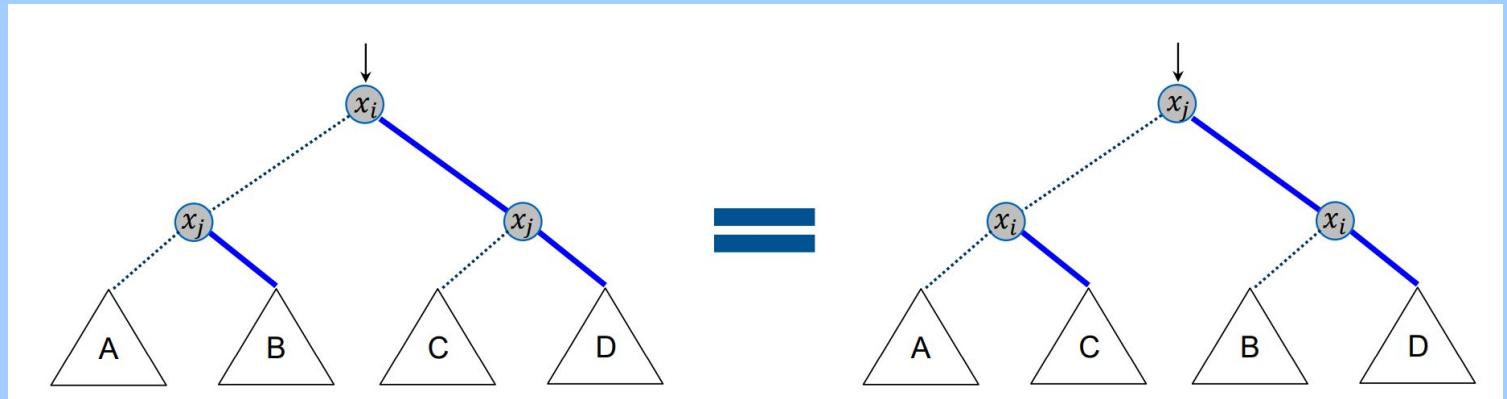
Regel 2: Elimination

Entferne Knoten, dessen beide Kinder zum gleichen Nachfolger zeigen.
(S-Reduktion)



- Ein Entscheidungsdiagramm heißt **frei**, wenn auf jedem Pfad von der Wurzel zu einem Blatt jede Variable höchstens einmal als Markierung vorkommt.
- Ein Entscheidungsdiagramm heißt **geordnet**, wenn auf jedem Pfad von der Wurzel zu einem Blatt die Variablen in der gleichen Reihenfolge abgefragt werden.
- Ein Entscheidungsdiagramm heißt **reduziert**, wenn sich keine Reduktionsregeln mehr anwenden lassen.

→ Reduced Ordered Binary Decision Diagrams (ROBDDs)

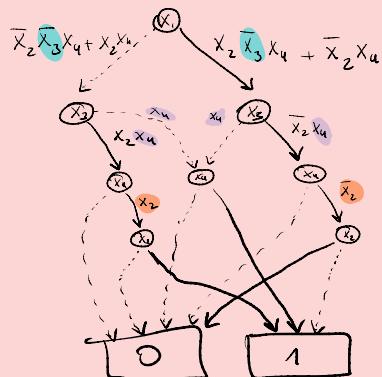


4 Konstruktion von BDDs

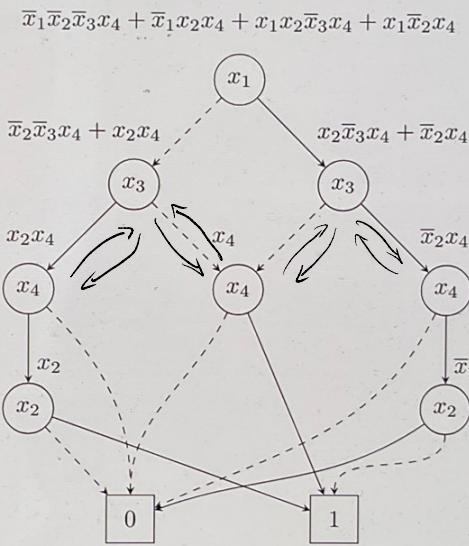
Betrachte die Boolesche Funktion

$$f(x_1, x_2, x_3, x_4) = \bar{x}_1 \bar{x}_2 \bar{x}_3 x_4 + \bar{x}_1 x_2 x_4 + x_1 x_2 \bar{x}_3 x_4 + x_1 \bar{x}_2 x_4$$

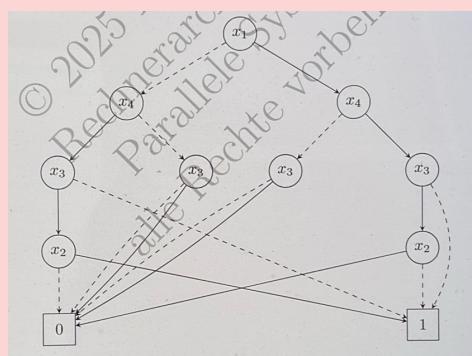
- a) Konstruiere den ROBDD der Funktion f für die Variablenordnung $x_1 < x_3 < x_4 < x_2$. Baue den BDD dafür ausgehend von der Wurzel auf und gib in jedem Knoten die übrige Schaltfunktion an die sich durch die Shannon-Entwicklung ergibt. Fasse dabei nach Möglichkeit entsprechende Knoten sofort zusammen.



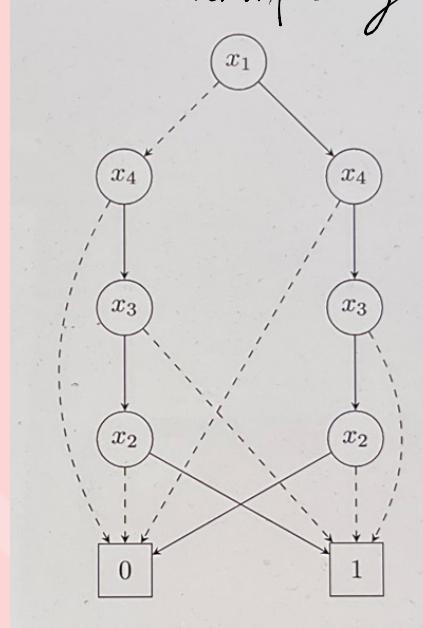
- b) Konstruiere den ROBDD der Funktion f für die Variablenordnung $x_1 < x_4 < x_3 < x_2$ durch Umordnen.



1. Schritt:



Nach Vereinfachung



- c) Beweise folgende Aussage aus der Vorlesung: Sei \otimes ein beliebiger(!) zweistelliger Operator, dann gilt

$$(x_i \cdot f_{x_i=1} + \bar{x}_i \cdot f_{x_i=0}) \otimes (x_i \cdot g_{x_i=1} + \bar{x}_i \cdot g_{x_i=0}) = x_i \cdot (f_{x_i=1} \otimes g_{x_i=1}) + \bar{x}_i \cdot (f_{x_i=0} \otimes g_{x_i=0})$$

Hinweis: Da der Operator \otimes beliebig gewählt ist, können die Gesetze der Booleschen Algebra nicht direkt auf \otimes angewendet werden. Versuche stattdessen eine Fallunterscheidung nach $x_i = 0$ und $x_i = 1$ und wende die Gesetze der Booleschen Algebra ausschließlich auf die Operatoren $+$ und \cdot an.

Fall 1: $x_i = 0$

$$\begin{aligned} & (0 \cdot m + 1 \cdot f_{x_i=0}) \otimes (0 \cdot n + 1 \cdot g_{x_i=0}) = 0 \cdot mn + 1 \cdot (f_{x_i=0} \otimes g_{x_i=0}) \\ \equiv & f_{x_i=0} \otimes g_{x_i=0} = f_{x_i=0} \otimes g_{x_i=0} \end{aligned}$$

Fall 2: $x_i = 1$

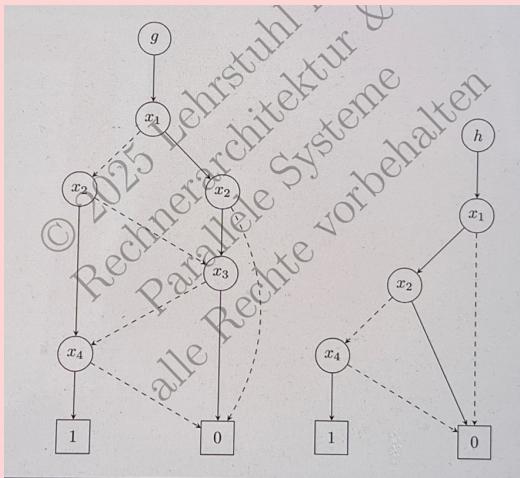
$$f_{x_i=1} \otimes g_{x_i=1} = f_{x_i=1} \otimes g_{x_i=1}$$

d) Im Folgenden sind ROBDDs für die Funktionen

$$g(x_1, x_2, x_3, x_4) = \bar{x}_1 \bar{x}_2 \bar{x}_3 x_4 + \bar{x}_1 x_2 x_4 + x_1 x_2 \bar{x}_3 x_4 \text{ und } h(x_1, x_2, x_3, x_4) = x_1 \bar{x}_2 x_4$$

gegeben. Konstruiere davon ausgehend den ROBDD der Funktion f für die Variablenordnung $x_1 < x_2 < x_3 < x_4$.

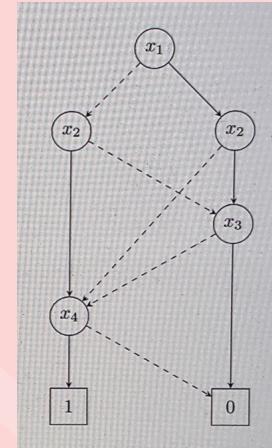
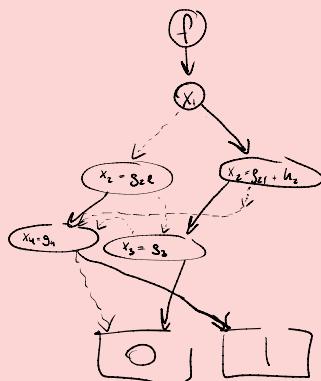
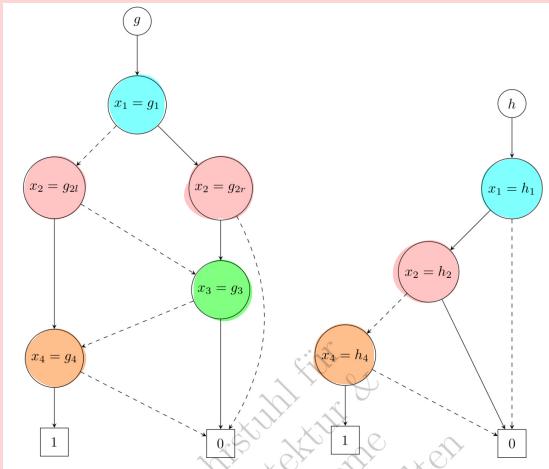
$$f = g + h$$



d) Im Folgenden sind ROBDDs für die Funktionen

$$g(x_1, x_2, x_3, x_4) = \bar{x}_1 \bar{x}_2 \bar{x}_3 x_4 + \bar{x}_1 x_2 x_4 + x_1 x_2 \bar{x}_3 x_4 \text{ und } h(x_1, x_2, x_3, x_4) = x_1 \bar{x}_2 x_4$$

gegeben. Konstruiere davon ausgehend den ROBDD der Funktion f für die Variablenordnung $x_1 < x_2 < x_3 < x_4$.



3 BDD Reduktion

Reduziere das folgende Entscheidungsdiagramm mittels I- und S-Reduktion so weit wie möglich. Markiere dafür bei jedem Zwischenschritt die verwendeten Knoten und gib die Art der verwendeten Reduktion, sowie das durch die Reduktion entstandene Diagramm an.

