Entwurfsverfahren digitaler Schaltungen

- Quine's Methode zur Bestimmung aller Primimplikanten von f

(Skript S. 24)

1) Bestimmung der CSOP aus SOP

(z.B.: SOP: $f(x,y,z) = xy$ ⇒ CSOP: $f(x,y,z) = xyz + xy\bar{z}$)

2) auf CSOP anwenden von:

a) spezielle Resolutionsgesetz (R): $x \cdot a + \bar{x} \cdot a = a$

b) Absorptionsgesetz (A): $a + ax = a$

Prim-Implikanten-Tabelle:

m_i	0-Kubus	A	R	1-Kubus	A	...
		✓ bzw. P_1	$m_i \cup m_j$			

Def.: Eine vollständige SOP von f enthält alle Primimplikanten P_i von f

VollSOP: $f = P_1 + P_2 + \dots$

- Quine's und Mc Cluskey's Bestimmung der MinSOP aus der VollSOP

(Skipt S. 25)

Überdeckungstabelle (aus VollSOP):

P \ m	m_a	m_b	...
P_1	a_1	a_2	
P_2	b_1	b_2	
⋮			

$a_i, b_i, \dots$: 0 oder 1

(Der müssen der Übersichtlichkeit halber nicht eingetragen werden)

a) Bestimmung essentieller Prim-Implikanten

b) Streichung von Mintermen (Spalten), die durch eine Überdeckung eines dominanten Minterms immer mit überdeckt werden.

c) Auswahl von maximaler Zahl von Einsen in den Zeilen

- Resolventenmethode (Deduktionsschema: Schichtenalgorithmus)

gg.: SOP oder (verkürzte) WT

gs.: alle Prim-Implikanten (Quine's PI-Theorem)

1) Absorptionsgesetz (A): $a + a \cdot b = a$ bzw. $a \cdot b \subseteq a$

2) allg. Resolutionsgesetz (R): $x \cdot a + \bar{x} \cdot b = x \cdot a + \bar{x} \cdot b + a \cdot b$

in disjunkter Form $\beta(x,a,b) = \beta(x,a,b) + \underbrace{a \cdot b}_{\text{Resolvente}}$

Mengenkubus

mit $x \in \sup(f)$, $a \in MC$, $b \in MC$; es gilt: FALLS $a \cdot b \neq 0$, DANN $\delta(xa, \bar{x}b) = 1$

Spezialfälle von R:

a) $\beta(x,a,a) = x \cdot a + \bar{x} \cdot a = xa + \bar{x}a + a = a$

b) $\beta(x,a,1) = x \cdot a + \bar{x} = xa + \bar{x} + a = \bar{x} + a$

c) $\beta(x,1,b) = x + \bar{x} \cdot b = x + \bar{x}b + b = x + b$

d) $\beta(x,a,a \cdot b) = xa + \bar{x}ab = xa + \bar{x}ab + ab = xa + ab$

e) $\beta(x,a \cdot b,a) = xab + \bar{x}a = xab + \bar{x}a + ab = ab + \bar{x}a$

SOP	Schicht
$c_i \quad c_{i+1}$ $\ldots + x \cdot a + \bar{x} \cdot b + \ldots$	0
$c_i \cdot c_{i+1}$ $\ldots + a \cdot b + \ldots$	1
$\vdots$	$\vdots$
+1 oder max. Anreicherung mit Prim-Implikanten	n

R

(A): z.B.

SOP	Schicht
$\cancel{yz} + \ldots$ (A)	i
$z + \ldots$	i+1

- Kombinatorische Optimierung (Skript S. 37)

geg.: $cov(f)$ bzw. f in SOP-Form, $F \subseteq f \subseteq F \cup D$

$c \in cov(f)$, $[cov(f)] \setminus \{c\} = cov(h)$

$f = c + h$, NB: $c_\ell = c(\ell = 1)$; $c_\ell \cdot \ell = c$

ges.: Nachfolger bzw. Vorgänger-Konfiguration von $cov_i(f)$

1. Expand (Entfernen eines Literals ℓ von c): $f = c + h = c_\ell + h$

Zulässigkeitsbedingung: $c_\ell \cdot \bar{\ell} \subseteq h \iff f_{c_\ell} \stackrel{!}{=} 1$

2. Reduce (Hinzufügen eines Literals ℓ zu c): $f = c + h = c \cdot \ell + h$

Zulässigkeitsbedingung: $c \cdot \bar{\ell} \subseteq h \iff f_c \stackrel{!}{=} 1$

3. Remove (Entfernen eines ganzen Produktterms): $f = c + h = h$

Zulässigkeitsbedingung: $c \subseteq h \iff h_c \stackrel{!}{=} 1$

- Tautologie-Nachweis (Skript S. 38)

a) · falls $f_{\bar{x}_i} \subseteq f_{x_i}$: $f(\underline{x}) = 1 \iff f_{\bar{x}_i}(\underline{x}) = 1$

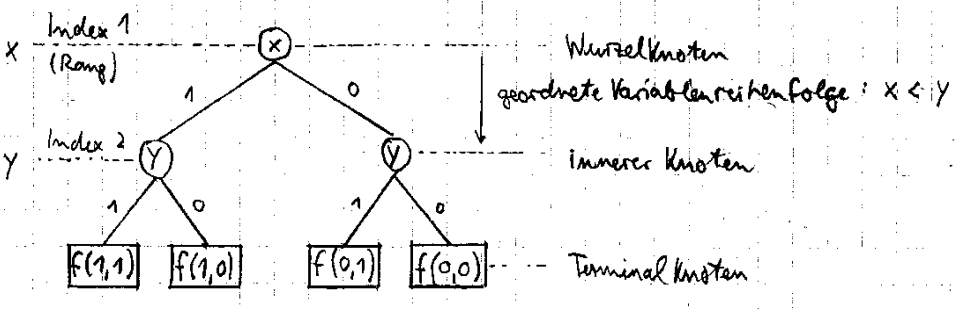
· falls $f_{x_i} \subseteq f_{\bar{x}_i}$: $f(\underline{x}) = 1 \iff f_{x_i}(\underline{x}) = 1$

(es genügt, den Tautologienachweis für $f_{\bar{x}_i}(\underline{x})$ bzw. $f_{x_i}(\underline{x})$ zu führen ...)

b) Anwendung der Resolventenmethode führt in der letzten Schicht zu dem Ergebnis "1"

c) alle Pfade des "Ordered Binary Decision Diagramm" (OBDD) führen letztendlich zu "1"

- OBDD:



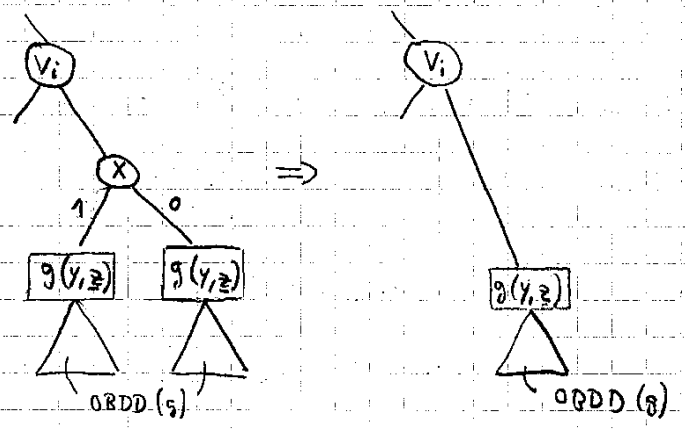
OBDD (f) $\quad f = f(x, y)$

- Vereinfachungsregeln (VR1 und VR2) für „baumförmige" OBDDs:

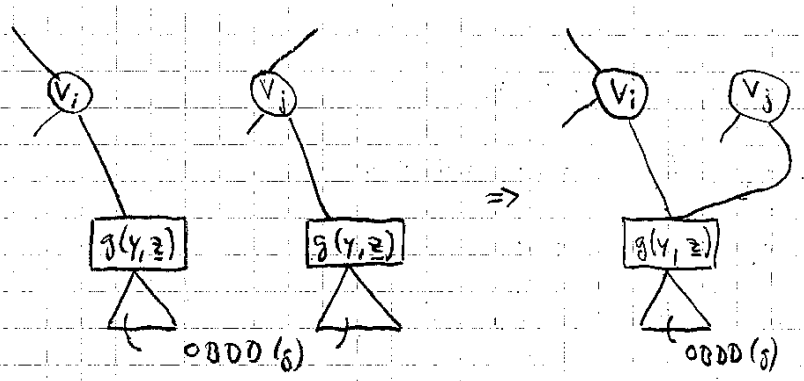
a) VR1: spezielles Resolutionsgesetz

Für $f(x,y,z)$ gilt: FALLS $f(1,y,z) = f(0,y,z) = g(y,z)$

DANN $f(x,y,z) = g(y,z)$



b) VR2: Vereinigung auf äquivalente Sub-OBDDs



Vollständige Anwendung von VR1 und VR2 führt zu "Reduced OBDD" (ROBDD)

- Die Darstellung einer Boolschen Funktion $f(\underline{x})$ durch ein ROBDD (f) ist mit vorgegebener Variablenreihenfolge $x_1 < x_2 < \ldots < x_n$ kanonisch (eindeutig)

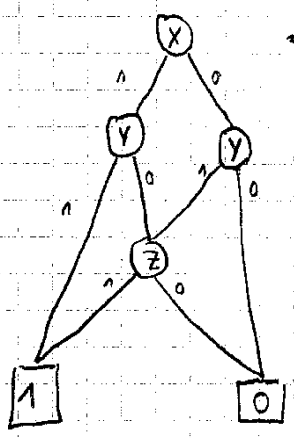
- Zwei ROBDDs sind äquivalent, FALLS gilt:
  a) Strukturgleichheit (isomorphismus)
  b) Belegungsgleichheit der Terminalknoten
  c) Indexgleichheit der Terminalknoten

- $\text{ROBDD}(g) \sim \text{ROBDD}(h) \iff g(\underline{x}) = h(\underline{x})$

$x_1 < x_2 < \ldots < x_n$

• MinSOP aus ROBBD:

$f(\underline{z})$ => MinSOP $f(\underline{z})$: Pfade nach [1] folgen und Knoten "und"-verknüpfen



=> MinSOP $f(\underline{z}) = \overbrace{xy + x\bar{y}z + \bar{x}yz}^{\text{SOP } f(\underline{z})} \underbrace{+ xz + yz}_{\text{Resolvente}}$

$= xy + xz + yz$

• VollSOP aus ROBBD:

a) Terminalknoten tauschen ([1] → [0] und [0] → [1])

b) Pfade nach [1] folgen und Knoten "und" verknüpfen => $\overline{f(\underline{z})}$

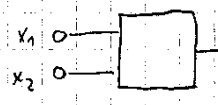
c) $\overline{f(\underline{z})}$ negieren und Resolventenmethode anwenden:

VollSOP $(f(\underline{z}))$: $\overline{f(\underline{z})} = x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}$

$f(\underline{z}) = \overline{x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}} \cdot (\bar{x}+y+z)\cdot(x+\bar{y}+z)\cdot(x+y)$

$= \left(\bar{x}\bar{y} + \bar{x}\bar{z} + \bar{y}\bar{z} + yx + xz + \bar{y}\bar{z} + z\right)\cdot(x+y)$

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



Fehlerfreie Schaltung

$y(x=0/1) = y_\mu$ Fehler f_μ

Fehlererkennung: $\boxed{y \oplus y_\mu = 1}$ Fehlerunterscheidung: $y_\mu \oplus y_k = 1$

Einfachfehler: x_1 ständig auf 1 $\Leftrightarrow$ $x_1/1$

x_1 ständig auf 0 $\Leftrightarrow$ $x_1/0$

Bsp.: $y = x_1 \cdot x_2$, $x_1/1$: $x_1 x_2 \oplus x_2 = x_2(x_1 \oplus 1) = x_2 \cdot \bar{x}_1 = 1$

=> man muß $x_2 = 1$ und $x_1 = 0$ anlegen, um den Fehler $x_1/1$ zu erkennen

- Fehlerüberdeckungstabelle

	f_1	f_2	f_3	f_4	f_5
t_1	0	1	0	0	1
t_5	0	1	0	1	1
t_{10}	1	0	1	0	1

↳ Test t_{10} erkennt Fehler f_5

$T = \{ t_1, t_5, t_{10} \}$ allg: $T = \{ t_\nu \mid \nu \in N \}$ Testmenge

$N = \{ 1, 5, 10 \}$, $t_\nu \in T = \nu \in N$ Testnummernmenge

n: Anzahl der Eingangsvariablen, $|T| = |N| = 2^n$

$F = \{ f_1, f_2, f_3, f_4, f_5 \}$ allg: $F = \{ f_\mu \mid \mu \in M \}$ Fehlermenge

$M = \{ 1, 2, 3, 4, 5 \}$ $f_\mu \in F = \mu \in M$ Fehlernummernmenge

m: Anzahl der zu testenden Fehler, $m = |F| = |M|$

- $t_\nu \, R \, f_\mu = y(\underline{x}_\nu) \oplus y_\mu(\underline{x}_\nu) = f_\mu \in F_\nu$: Test t_ν erkennt Fehler f_μ
- $y_\mu(\underline{x}_\nu) \oplus y_k(\underline{x}_\nu) = 1$: mit Eingangsbelegung $\underline{x}_\nu$ können Fehler μ und k unterschieden werden

$y_\mu(\underline{x}_\nu) \oplus y_k(\underline{x}_\nu) = t_\nu \, R \, f_\mu \oplus t_\nu \, R \, f_k$

- $C_{M,N} = \bigwedge_{\mu \in M} \bigvee_{\nu \in N} t_\nu \, R \, f_\mu$: Fehlerüberdeckungstabelle (Vereinigungsmenge der Fehlergruppen aller angenommenen Fehler)

Vollständiges Test: $C_{M,N} = 1$

- Algorithmus zur suboptimalen Bestimmung einer Testmenge (Heuristik):

Testauswahl in Reihenfolge der Mächtigkeiten der Fehlergruppe

1. Bestimme die Mächtigkeiten der Fehlergruppe
2. Wähle die Fehlergruppe F_v mit größter Mächtigkeit
3. Füge korrespondierenden Test t_v zur Testmenge hinzu
4. Streiche alle von t_v erkannten Fehler und streiche F_v
5. FALLS alle Fehler erkannt sind,

   DANN ist eine Testmenge gefunden

   SONST gehe zu 1.

Bsp.: ① ① ② ① ② ① ④ ③ ③ — (i): als i. gestrichen

t_v \ f_μ	2	4	8	9	15	5	6	11	3	21	$\lvert F_v^{(i)} \rvert$				
0		1			1					1	2	0	0	0	0
1	~~1~~	~~1~~	1	~~1~~	~~1~~	~~1~~	~~1~~	~~1~~	~~1~~	~~1~~	(4)	0	0	0	0
4	1				1	1					2	1	0	0	0
6	1			1	1	1					4	(3)	0	0	0
2	1					1	1		1	1	3	2	(2)	0	0
3						1			1	1	2	2	2	0	0
5	1					1		1		1	3	3	2	(1)	0
7								1		1	2	2	2	1	0
Auswahl:											t_1	t_6	t_2	t_5	

- Testbestimmung - Schaltnetz

• Boolsche Differenz:

$$y_z = y(z, \underline{x}) \oplus y(\bar{z}, \underline{x})$$

$$y_z = y(z=1) \oplus y(z=0)$$

• Test: $\underline{x}\, R\, z/0 = \bar{z} \cdot y_z = 1$

$\underline{x}\, R\, z/1 = z \cdot y_z = 1$

$\underline{x}\, R\, z/0 = 1$: Bedingung zum Erkennen des Fehlers z ständig auf "0"

• Fehlerbelegung für $z/0$: $z(\underline{x}) = 1$

(Einstellbarkeit!) für $z/1$: $z(\underline{x}) = 0$

• Sensibilisierungs-belegung: $y_z(\underline{x}) = 1$

(Beobachtbarkeit!)

• Rechenregeln: (Beispiele: Skript Teil 2, S. 85)

1) $y_x = 0$, falls $y \neq f(x)$

2) $y_y = 1$ (wg. $y \oplus \bar{y} = 1$)

3) $(\bar{y})_x = y_x$

4) $(z \oplus w)_x = z_x \oplus w_x$

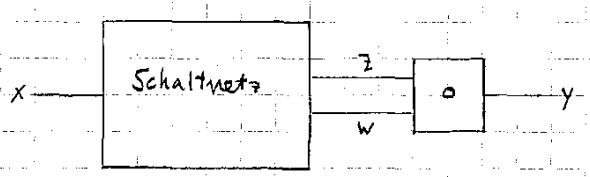
5) $(z \cdot w)_x = z \cdot w_x \oplus z_y \cdot w \oplus z_x \cdot w_x$

6) $(z + w)_x = \bar{z} \cdot w_x \oplus z_x \cdot \bar{w} \oplus z_x \cdot w_x$

7) $y_x = y_z \cdot z_x$, falls $y = y(z(x))$

8) $(y_z)_w = (y_w)_z$

- Strukturbezogene Berechnung der Boolschen Differenz



$y = \underbrace{z \circ w}$ beliebige logische Verknüpfung

$y_x = 1$ vorausgesetzt (Fehler an x führt zu Fehlerbelegung an y)

• ein Einfachfehler an x kann an y beobachtet werden ...:

$$\boxed{y \oplus y_x = (z \oplus z_x) \circ (w \oplus w_x)}$$

1, Lokal: $y_z = (\bar{z} \circ w) \oplus (z \circ w)$

$y_w = (z \circ w) \oplus (z \circ \bar{w})$

2, Global: $y_x = \left[(z \oplus z_x) \circ (w \oplus w_x) \right] \oplus \left[z \circ w \right]$

• allg. Berechnungsvorschrift:

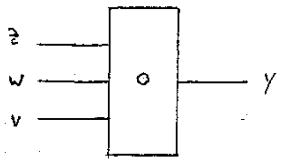
$$\boxed{y_x = y_z \cdot z_x \cdot \overline{w}_x + y_w \cdot w_x \cdot \overline{z}_x + z_x \cdot w_x \cdot \left[\bar{z} \circ \bar{w} \oplus z \circ w \right]}$$

a, für EXOR-Gatter gilt: $\left[\bar{z} \oplus \bar{w} \oplus z \oplus w \right] = 0$; $y_z = 1$, $y_w = 1$

b, für AND, NAND, OR- und NOR-Gatter gilt: $\left[\bar{z} \circ \bar{w} \oplus z \circ w \right] = \overline{z \oplus w}$

c, $\overline{z \oplus w} = 1$: gleichsinnige Belegung von z und w

$\overline{z \oplus w} = 0$: gegensinnige Belegung von z und w



$(y \oplus y_x) = (z \oplus z_x) \circ (w \oplus w_x) \circ (v \oplus v_x)$

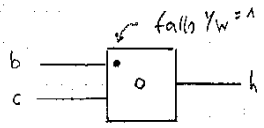
$y_x = \left[(z \oplus z_x) \circ (w \oplus w_x) \circ (v \oplus v_x) \right] \oplus \left[z \circ w \circ v \right]$

Redeweisen für y_x:

- Boolsche Differenz von y nach x (Boolean Difference)
- Beobachtbarkeit von x an y (Observability)
- Empfindlichkeit von y gegen x (Sensitivity)

Redeweisen für $y_x = 1$:

- y ist von x funktionell abhängig
- eine Fehlerbelegung von x (am Signal x) führt zu einer Fehlerbelegung von y (am Signal y)



$h_c = (b \circ c) \oplus (b \circ \bar{c})$

OR / NOR : $y_b = \bar{c}$

AND / NAND : $y_b = c$

XOR / XNOR : $y_b = 1$

- AND - EXOR : benötigte Formeln

$x \oplus x = 0 \qquad x \oplus 0 = x \qquad x \oplus 1 = \bar{x}$

DG: $x \cdot (y \oplus z) = xy \oplus xz \qquad x + y = x \oplus y \oplus xy$

Bsp.: $y = \bar{x}_1 x_2 + \bar{x}_3 \bar{x}_4$

$= (x_1 \oplus 1)\, x_2 + (x_3 \oplus 1)(x_4 \oplus 1)$

$= (x_1 x_2 \oplus x_2) \oplus (x_3 x_4 \oplus x_3 \oplus x_4 \oplus 1) \oplus (x_1 x_2 \oplus x_2)(x_3 x_4 \oplus x_3 \oplus x_4 \oplus 1)$ DG

$= \cancel{x_1 x_2} \oplus \cancel{x_2} \oplus x_3 x_4 \oplus x_3 \oplus x_4 \oplus x_1 x_2 x_3 x_4 \oplus x_1 x_2 x_3 \oplus x_1 x_2 x_4$

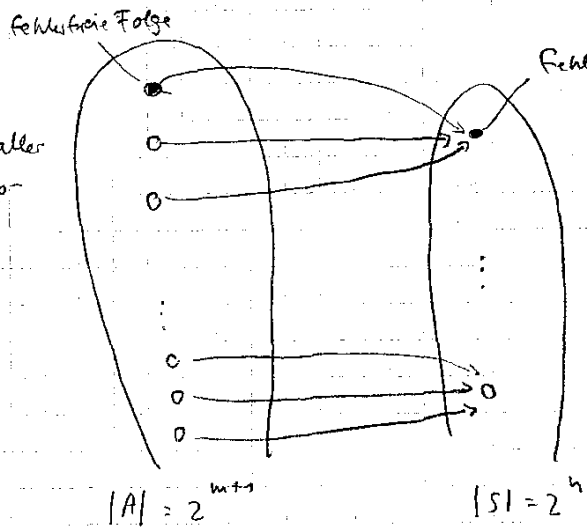
$+ \cancel{x_1 x_2} \oplus x_2 x_3 x_4 \oplus x_2 x_3 \oplus x_2 x_3 \oplus \cancel{x_2}$

Boolsche Differenz: $y_{x_1} = x_2 x_3 x_4 \oplus x_2 x_3 \oplus x_2 x_4$

(hier: Terme, die kein "x_1" enthalten, entfallen: $x_3 x_4 \rightsquigarrow 0$
Terme, die "x_1" enthalten, werden um diesen gekürzt: $x_1 x_2 x_3 \rightsquigarrow x_2 x_3$)

- D-Algorithmus: siehe Skript S. 96 ...

- Testen mit Signaturanalyse:



$2^{m+1-n} - 1$: Zahl der fehlerhaften Folgen, die auf die Sollsignatur abgebildet werden

$2^{m+1} - 1$: Zahl der fehlerhaften Folgen

Im Mittel werden $\frac{|A|}{|S|}$ Registereingangsfolgen auf eine Signatur abgebildet

Fehlernichterkennungswahrscheinlichkeit: $P_f = \frac{2^{m+1-n} - 1}{2^{m+1} - 1}$; für $m \gg n$: $P_f = \frac{1}{2^n}$

Fehlererkennungswahrscheinlichkeit: $p = 1 - P_f = 1 - 2^{-n}$ (für $m \gg n$)