Formelsammlung EDS

Vorsätze für Maßeinheiten

E	10^{15}	Exa -
T	10^{12}	Terra -
G	10^{9}	Giga -
M	10^{6}	Mega -
k	10^{3}	Kilo -
c	10^{-2}	Centi -
mm	10^{-3}	Milli -
µ	10^{-6}	Micro -
n	10^{-9}	Nano -
p	10^{-12}	Pico -
f	10^{-15}	Femto -
a	10^{-18}	Atto -

I. Logiksynthese

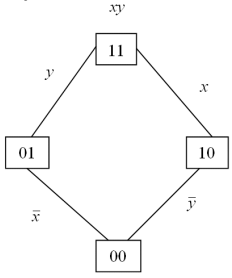
1. Grundlagen der Logiksynthese

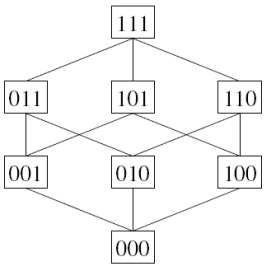
- Zweiwertige Boolesche Algebra
  - Schaltalgebra $(\{1, 0\}; \cdot; +; \bar{\ })$
  - Aussagenalgebra $(\{w, f\}; \wedge; \vee; \neg)$
- Schaltfunktionen
  - UND, ODER, NICHT, XOR, XNOR
- Boolesche Algebra

Kommutativgesetz	$xy = yx$	$x + y = y + x$
Assoziativgesetz	$(xy)z = x(yz)$	$(x + y) + z = x + (y + z)$
Distributivgesetz	$x(y + z) = xy + xz$	$x + yz = (x + y)(x + z)$
Idempotenzgesetz	$xx = x$	$x + x = x$
Absorbtionsgesetz	$x(x + y) = x$	$x + xy = x$
Neutrales Element	$x \cdot 1 = x$	$x + 0 = x$
Dominantes Element	$x \cdot 0 = 0$	$x + 1 = 1$
Negation	$x \cdot \bar{x} = 0$	$x + \bar{x} = 1$
Doppelte Negation	$\bar{\bar{x}} = x$	
De Morgan	$\overline{x \cdot y} = \bar{x} + \bar{y}$	$\overline{x + y} = \bar{x} \cdot \bar{y}$

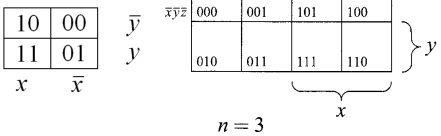
2. Binäre Boolesche Funktionen

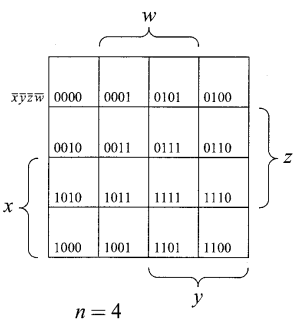
- Darstellungsmöglichkeiten
  - Wertetabelle
  - Abbildungsvorschrift: $z = f(\underline{x})$
  - Gatterschaltung
  - Binärer Entscheidungsbaum (BDD)
  - Pfeildiagramm der Abbildung $f : B^2 \longrightarrow B$; B = {1, 0}
  - Einsmenge, $onset(f)$, $offset(f)$
    - $onset(x \cdot y) = \{11\}$
    - $onset(x + y) = \{01,10,11\}$
    - $offset(x + y) = \{00\}$
  - Kubengraphen

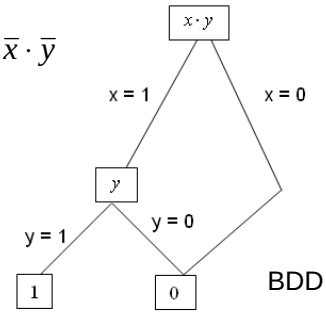




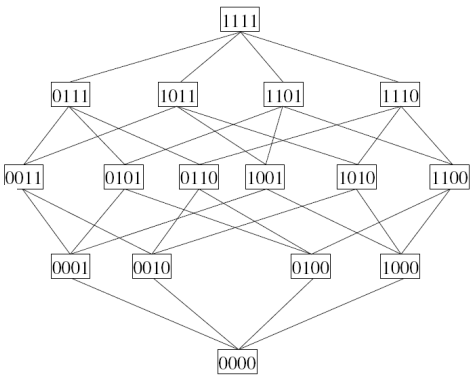
  - Karnaugh-Diagramme



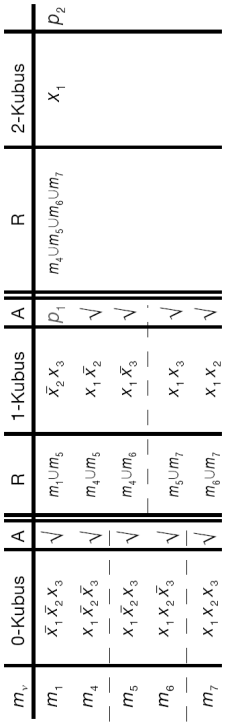




BDD



- Boolescher Raum, Definitionen und Formale Schreibweisen
  - o $\hat{\underline{x}}_i$: Belegungs-n-Tupel, i wird als Dualzahl interpretiert ($\hat{\underline{x}}_5 \mathrel{\hat{=}} 101$)
  - o m_i: Vollproduktterm, Minterm, 0-Kubus, enthält alle Literale, i wird wie oben
  - o c_j: d_j -Kubus, Produktterm (einer Literalteilmenge)
  - o d_j: Dimension des Kubus = n - Anzahl der Literale des Kubus
    (n = Dimension des Boolschen Raumes)
  - o $x_i, \bar{x}_i$: Literal
  - o $\delta(c_i, c_j) = \delta_{ij} = \left|\{l \mid l \in c_i \wedge \bar{l} \in c_j\}\right|$
    Belegungsabstand zwischen zwei Kuben = Kardinalität der Menge von Literalen die im ersten Kubus positiv und im zweiten negativ enthalten sind
  - o Kubengraph: $\delta = 1$ => Kante
  - o MOC: Menge aller 0-Kuben in B^n
  - o MC: Menge aller Kuben in B^n
- n-stellige Boolsche Funktion
  - o $f(\underline{x}) = f(x_1, x_2, ..., x_n)$; $f : B^n \longrightarrow B$
  - o $f = \{(\hat{\underline{x}}, \hat{y}) \mid f(\hat{\underline{x}}) = \hat{y}\}_{B^n \times B}$; $f \subseteq B^n \times B$
  - o $onset(f) = \{\hat{\underline{x}} \in B^n \mid f(\hat{\underline{x}}) = 1\}$
  - o Vereinbarung $f = onset(f)$
- KDNF = Canonical sum of products formula = CSOP
  - o CSOP: $f = \sum_k m_k = \bigcup_k m_k$; $m_k \subseteq f$, $m_k \in MOC$
- DNF = sum of products formula = SOP
  - o SOP: $f = \sum_k c_k = \bigcup_k c_k$; $c_k \subseteq f$, $c_k \in MC$
  - o $cov(f) = \{c_1, c_2, c_3, ..., c_k\}$ mit $f = \sum_{c_k \in \mathrm{cov}(f)} c_k$ (Überdeckung, Hülle von f)
- KKNF = CPOS
  - o CPOS: $f = \prod_k \overline{m}_k = \bigcap_k \overline{m}_k$; $m_k \subseteq \bar{f}$, $m_k \in MOC$ (NOT aus De Morgan !!!)
- KNF = POS
  - o POS: $f = \prod_k \bar{c}_k = \bigcap_k \bar{c}_k$; $c_k \subseteq \bar{f}$, $c_k \in MC$ (NOT aus De Morgan !!!)
- Implikanten von Funktionen
  - o Menge der Implikanten: $MI = \{c \in MC \mid c \subseteq f\}$
  - o Menge der Prim-Implikanten $MPI = \{p \in MI \mid p \not\subset c, \text{für alle } c \in MI\}$;
    kann durch Literalentfernung überprüft werden
  - o $MPI \subseteq MI \subseteq MC$



3. Optimierungsverfahren für boolesche Funktionen

- Bestimmung der VollSOP (Quine / McKluskey)
  - o SOP durch erweitern in CSOP umwandeln
  - o Problem: Zahl der Minterme in der SOP kann sehr groß werden
  - o Ermittlung der Prim-Implikanten durch:
    - Spezielles Resolutionsgesetz (R): $xa + \bar{x}a = a$
    - Absorbtionsgesetz (A): $a + ab = a$
  - o VollSOP: Summe aller Prim-Implikanten einer Funktion

- o Prim-Überdeckung: $pcov(f) = \{p_1, p_2, ...\}$
- o MinSOP: Summe aller essentiellen Prim-Implikanten

- Bestimmung der MinSOP aus der VollSOP (Quine / McKluskey)
  a) Überdeckungsbedingung (C = 1)
  - o $C = (m_0 \subseteq p_1) \cdot (m_2 \subseteq p_1 + m_2 \subseteq p_2)...$
  - o Einführung einer Auswahlvariable τ_ν von p_ν mit τ_ν = 1 wenn p_ν gewählt
  - o $C = \tau_1(\tau_1 + \tau_2)...$
  - o vereinfachen (Absorption), Produkt als Ergebnis liefert Lösung
  b) Überdeckungstabelle

	m_1	m_4	m_5	m_6	m_7
p_1	1		1		
p_2		1	1	1	1

- Bestimmung der VollSOP: Resolventenmethode
  - o SOP kann direkt verwendet werden
  - o Ermittlung der Prim-Implikanten durch:
    - Allgemeines Resolutionsgesetz: $xa + \bar{x}b = xa + \bar{x}b + ab$
      $a, b \in MC$ und ab ist Resolvente
    - Absorptionsgesetz: $a + ab = a$
  - o Schichtenalgorithmus:
    - Entwicklung der Schichten durch Resolventenbildung
    - Streichung von Kuben durch Absorption
    - Ergebnis, wenn Tautologie oder keine weitere Schichtenbildung mehr möglich ist

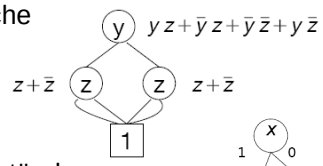
f	Schicht
$x\,y\,z + \bar{x}\,\bar{y}\,w + x\,y\,\bar{w} + x\,\bar{y}\,\bar{z}\,w$	0
$+ yzw + xzw + xyw + \bar{y}\,\bar{w} + y\,\bar{z}\,\bar{w} + x\,\bar{z}\,w$	1
$+ \bar{z}\bar{w} + y\bar{w} + x\bar{w} + \bar{z}w$	2
$+ w$	3

- Bestimmung der VollSOP aus POS
  - o Theorem: $f_1 \cdot f_2$ ist VollSOP wenn f_1 und f_2 jeweils VollSOP ist

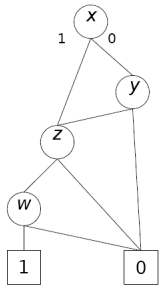
- Operationen auf Boolsche Funktionen
  - o Kofaktor: teilweises Einsetzen von Variabeln: $f_{x_i} = f\big|_{x_i=1}$ $f_{\bar{x}_i} = f\big|_{\bar{x}_i=0}$
  - o Kommutativgesetz: $f(0,1,x_3) = f_{\bar{x}_1 x_2} = (f_{\bar{x}_1})_{x_2} = (f_{x_2})_{\bar{x}_1}$
  - o Substitutionsregel: $c \cdot f = c \cdot f_c$
  - o Entwicklungssatz (Shannon Dekomposition):
    $f(x_1,...,x_i,...,x_n) = x_i f_{x_i} + \bar{x}_i f_{\bar{x}_i} = \beta(x_i, f_{x_i}, f_{\bar{x}_i})$
  - o Tautologie: $f(\underline{x}) = 1 \Leftrightarrow f_{x_i} = 1 \; und \; f_{\bar{x}_i} = 1$
  - o Monotonie:
    - monoton fallend: $f_{x_i} \subseteq f_{\bar{x}_i}$ $f = \bar{x} \cdot g + h$
    - monoton steigend: $f_{\bar{x}_i} \subseteq f_{x_i}$ $f = x \cdot g + h$
    - vereinfachter Tautologienachweis, es muss nur noch h auf Tautologie überprüft werden

- Heuristische Optimierung (Lokale Suche)
  - o Suche nach einem minimalen $cov(f)$ (= MinSOP) aus der Menge aller $cov(f)$:
    $mincov(f) = min\{...cov_i(f)...\}$
  - o Menge aller zulässigen cover: $H = \{cov_1(f),...,cov_i(f),....,cov_N(f)\}$
  - o Nummernmenge: $H = \{1, ..., i, ..., N\}; N < \infty$
  - o Kosten der Konfiguration(= des covers i), Literalanzahl: $\varphi(i)$
  - o Konfigurationsgraph:
    - Knoten: cover, Konfiguration
    - Kante: Modifikation, Literalentfernung, Literalzahlerhöhung, Kubenentfernung
    - Kantenrichtung: Abnahme der Kosten, Abnahme der Literale
    - Konfigurationsgraph: $G = (H, R)$, H = Kontenmenge, R = Kantenmenge
      $R \subseteq H \times H$
    - Knoten mit ausschließlich Pfeilspitzen ist lokales Minimum

- Modifikationen ($f = c + h$, $c \in cov(f)$, $\text{cov}(h) = [\text{cov}(f)]/c$):
  - Expand (Literalentfernung): wenn $c_l \subseteq f$ oder $c_l \cdot \bar{l} \subseteq h$
  - Reduce (Literalzahlerhöhung): wenn $c \cdot \bar{l} \subseteq h$
  - Remove (Kubenentfernung): wenn $c \subseteq h$
- Inklusionsprüfung durch Tautologienachweis:
  - $c \subseteq f \Leftrightarrow f_c = 1$
  - $c_1 + c_2 \subseteq f \Leftrightarrow f_{c_1} = 1 \wedge f_{c_2} = 1$
  - Kofaktorisierung einer Funkion führt zu einfacheren Kofaktoren
  - Allgemeines Standartproblem (-> Schichtenalgorithmus + OBB)
  - Effizientes Verfahren: Resolventenmethode mit Tiefensuche



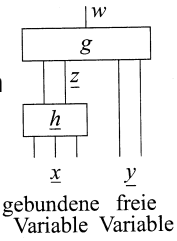
- OBDD (= Ordered Binary Decision Diagramm)
  - Anwendung der Shannon-Entwicklung
    $f(x_1,...,x_i,...,x_n) = x_i f_{x_i} + \bar{x}_i f_{\bar{x}_i} = \beta(x_i, f_{x_i}, f_{\bar{x}_i})$
  - kein Zusammenfassen gleicher Zustände oder weglassen von Zuständen



- ROBDD = (Reduced OBDD): OBDD so weit wie möglich vereinfachen

4. Schaltungen mit mehren Ausgängen und mehr als 2 Ebenen

- Synthese von Schaltungen mit mehreren Ausgängen
  - Bestimmung der Mehrfach-Implikanten: $f_1 \cdot f_2$
  - Funktionen von SOP auf MinSOP expandieren (Expand)
  - Funktionen von durch Addition der Mehrfach-Implikanten reduzieren (Reduce)
  - Funktionen durch Kubenentfernung minimieren (Remove)
  - ➢ Irredundant Cover
- Synthese von Schaltungen mit Don't Cares
  - Hinzufügen der Don't Care Sets
  - Vereinfachung
  - Don't Care Sets können wieder weggelassen werden
- Synthese von Schaltungen mit mehr als zwei Ebenen
  - optimierte zweistufige Form geschickt faktorisieren -> geschicktes Ausklammern
  - algebraische Umformungen -> ungünstige Lösungen
  - boolesche Verfahren -> sehr aufwändig



- Funktionale Dekomposition
  - Ziel: Zerlegung eines Moduls f in Module g und $\underline{h}$
  - Bedingung Dekompositionsgewinn: $|z| \leq |x| - 1$
  - Ermittlung einer günstigen Aufteilung mittels BDDs
  - Schritt 1:
    - Auswerten von $f(\underline{\hat{x}}_i, \underline{y})$ -> Dekompositionsmatrix
    - Zusammenfassen der $\underline{\hat{x}}_i$ zu Äquivalenzklassen
  - Schritt 2:
    - Codierung der Äquivalenzklassen durch $\underline{z} = \underline{h}(\underline{x})$
    - Aufstellen einer Zuordnungstabelle
    - Ermittlung der Dekompositionsfunktion $\underline{h}(\underline{x})$
  - Schritt 3:
    - Konstruktion einer Kompositionsfunktion $w = g(\underline{z}, \underline{y})$
    - Produktterme mit Don't Care Werte dürfen beliebig hinzuaddiert werden

	$(\hat{x}_1,\hat{x}_2,\hat{x}_3)_i$							
$(\hat{y}_1,\hat{y}_2)$	000	010	100	001	111	011	101	110
00	1	1	1	1	1	1	1	1
01	1	1	1	0	0	0	0	0
10	0	0	0	1	1	1	1	1
11	0	0	0	1	1	0	0	0
$f(\hat{x}_i,y)$	$\bar{y}_1$			$y_1+\bar{y}_2$		$\bar{y}_2$		

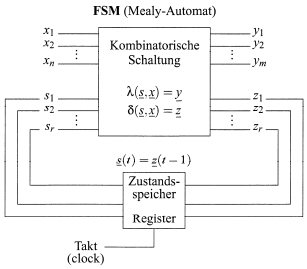
Dekompositions-Matrix für $f(\underline{x},\underline{y})$

$\hat{\underline{x}}_i \in X$	$\hat{\underline{z}}_k \in Z$ $(\hat{z}_1,\hat{z}_2)_k$	$\underline{z} = \underline{h}(\underline{x})$	$g(\hat{\underline{z}}_k,\underline{y})$
000,010,100	00	$z_1 \cdot z_2 = m_0 + m_2 + m_4$	y_1
001,111	10	$z_1 \cdot \bar{z}_2 = m_1 + m_7$	$y_1 + \bar{y}_2$
011,101,110	11	$z_1 \cdot z_2 = m_3 + m_5 + m_6$	$\bar{y}_2$
	01	$\bar{z}_1 \cdot z_2 = 1$ nicht möglich	

Zuordnungs-Tabelle

$z_1 = z_1 \cdot \bar{z}_2 + z_1 \cdot z_2 = m_1 + m_7 + m_3 + m_5 + m_6$
$z_2 = \bar{z}_1 \cdot z_2 + z_1 \cdot z_2 = m_3 + m_5 + m_6$

5. Endliche Automaten - Finite State Machines (FSM)



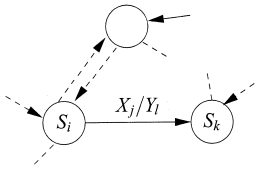
- $FSM = (S, I, O, \delta, \lambda, S^0)$
  - o S: Zustandsmenge
  - o I: Eingabealphabet
  - o O: Ausgabealphabet
  - o δ: Zustandsübergangsfunktion $S \times I \to S$
  - o λ: Ausgangsfunktion $S \times I \to O$ (Mealy); $S \to O$ (Moore)
- Zustandsübergangstabelle / Ausgangstabelle:
  - o X_j : Eingabezeichen Y_l : Ausgabezeichen
  - o S_i : Zustand S_k : Folgezustand

δ	$\cdots$	X_j	$\cdots$
$\vdots$		$\vdots$	
S_i	$\cdots$	S_k	$\cdots$
$\vdots$		$\vdots$	

λ	$\cdots$	X_j	$\cdots$
$\vdots$		$\vdots$	
S_j	$\cdots$	Y_l	$\cdots$
$\vdots$		$\vdots$	

- Zustands-Ausgangs-Funktion: $\mu : S \times I \to S \times O$
- ZA-Tabelle / Zustandsgraph / Codierte Kuben-Tabelle

μ	$\cdots$	X_j	$\cdots$
$\vdots$		$\vdots$	
S_i	$\cdots$	(S_k, Y_l)	$\cdots$
$\vdots$		$\vdots$	



x	$s_1\ s_2$	$z_1\ z_2$	y
0	0 0	0 0	0
1	0 0	0 1	1
0	0 1	0 0	0
1	0 1	1 0	0
0	1 0	1 0	0
1	1 0	0 0	1
0	1 1	$(1)**(0)$	$*(0)$
1	1 1	$(1)**(0)$	$*(0)$

- Realisierung der FSM durch Zustandskodierung
- Zustandsminimierung
  - o Streichung aller nichterreichbaren Zustände
  - o Zusammenfassen äquivalenter Zustände:
    - Zwei Zustände S_i und S_j sind genau dann äquivalent, wenn sie für jede Folge von Eingabezeichen die gleiche Folge von Ausgabezeichen lieferen.
    - Ziel: Partitionierung der ZA-Tabelle bis Zustände nicht mehr unterscheidbar sind
    - Sind mehrere Zustände untereinander nicht mehr unterscheidbar, so sind sie äquivalent zueinander
    - $S_i \sim S_j \Leftrightarrow S_i \overset{1}{\sim} S_j \wedge S_i \overset{2}{\sim} S_j \wedge S_i \overset{3}{\sim} S_j \wedge \ldots$
    - $S_i \overset{1}{\sim} S_j \Leftrightarrow \left[\lambda(S_i,0) = \lambda(S_j,0)\right] \wedge \left[\lambda(S_i,1) = \lambda(S_j,1)\right]$ (1-äquivalent)
    - $S_i \overset{2}{\sim} S_j \Leftrightarrow S_i \overset{1}{\sim} S_j \wedge \left[\delta(S_i,0) \overset{1}{\sim} \delta(S_j,0)\right] \wedge \left[\delta(S_i,1) \overset{1}{\sim} \delta(S_j,1)\right]$
    - $S_i \overset{3}{\sim} S_j \Leftrightarrow S_i \overset{2}{\sim} S_j \wedge \left[\delta(S_i,0) \overset{2}{\sim} \delta(S_j,0)\right] \wedge \left[\delta(S_i,1) \overset{2}{\sim} \delta(S_j,1)\right]$ usw.

μ	$x=0$	$x=1$	
S_1	$(S_4,0)$	$(S_2,1)$	
S_4	$(S_1,0)$	$(S_2,1)$	
S_3	$(S_3,0)$	$(S_1,1)$	1
S_2	$(S_4,0)$	$(S_3,0)$	2

ZA-Tabelle

    - z.B. sind bei dieser Tabelle S_1 und S_4 3-äquivalent und damit nicht weiter unterscheidbar -> S_1 und S_4 sind zueinander äquivalent

II. Logiksimulation

- Schaltungsmodellierung
  - o einfügen von Gatterlaufzeiten und Leitungslaufzeiten in die Schaltung
  - o Hazard = Signal das der reinen Logik widerspricht und durch Verzögerungszeiten entsteht

OR	0	1	X
0	0	1	X
1	1	1	1
X	X	1	X

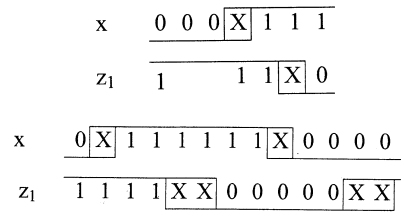
- Simulation
  - o Schritt 1: Auswertung der reinen Logikschaltung
  - o Schritt 2: Auswertung des Zeitverlaufs ohne Logik

  -> ablesen der maximalen Einschwingunsicherheit möglich

AND	0	1	X
0	0	0	0
1	0	1	X
X	0	X	X

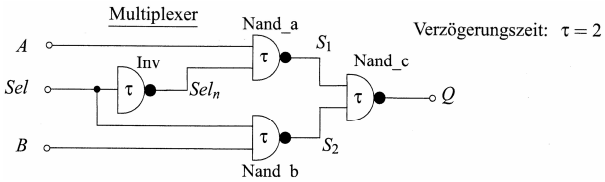
NOT	
0	1
1	0
X	X

- Simulation mit Flankenunsicherheit
  - Einführung eines neuen Signalwerts X
  - Wertetabellen mit den neuen Signalwert X
  - Graphik: Verzögerung des Gatters = 2Δ , X = Δ
- Simulation mit Laufzeitunsicherheit
  - worst-case Analyse, Auffüllung mit X
  - Graphik: Gatter: $3\Delta - 4\Delta$
- Simulation mit einer Ereignisstabelle:



t	A	B	Sel	Sel_n	S_1	S_2	Q	ausgewertete Elemente	neue Ereignisse ($signal, val, t_{gen}, t_{exe}$)
0	'0'	'0'	'1'	'0'	'1'	'1'	'0'	Initialzustand (Stimuli)	(A, '1', 0, 20); (B, '1', 0, 10); (Sel, '0', 0, 30)
10		'1'						Nand_b	(S_2, '0', 10, 12)
12						'0'		Nand_c	(Q, '1', 12, 14)
14							'1'	–	–
20	'1'							Nand_a	–
30			'0'					Inv, Nand_b	(Sel_n, '1', 30, 32); (S_2, '1', 30, 32)
32				'1'		'1'		Nand_a, Nand_c	(S_1, '0', 32, 34); (Q, '0', 32, 34)
34					'0'		'0'	Nand_c	(Q, '1', 34, 36)
36							'1'	–	–

- VHDL
  - transport delay: unendliche Bandbreite
  - inertial delay: reale Gatter



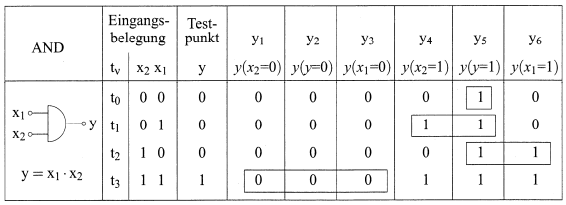
III. Testverfahren

1. Fehlerdiagnose

- Arten von Fehlern:
  - dynamisch (wird nicht weiter behandelt)
  - statisch (stuck at 1 / stuck at 0)
- Probleme der Fehlererkennung
  - Beobachtbarkeit
  - Einstellbarkeit

2. Fehlerüberdeckungstabelle

2.1 Fehlersimulation



AND	Eingangs-belegung		Test-punkt	y_1	y_2	y_3	y_4	y_5	y_6
	t_v	x_2 x_1	y	$y(x_2=0)$	$y(y=0)$	$y(x_1=0)$	$y(x_2=1)$	$y(y=1)$	$y(x_1=1)$
	t_0	0 0	0	0	0	0	0	1	0
	t_1	0 1	0	0	0	0	1	1	0
	t_2	1 0	0	0	0	0	0	1	1
	t_3	1 1	1	0	0	0	1	1	1

- Eingangsbelegung
  - Test t_v entspricht Testmuster …x_2x_1
- Testpunkt y (Wert der Logik bei fehlerfreier Funktion)
- Werte der Logik bezogen auf die (Einfach-)Fehler f_μ:

$$y_\mu = y(x = 1/0)$$ Wert der Logik für x stuck at 1 / 0

- Fehlererkennung
  - $y \oplus y_\mu = 1$ Fehler muss am Ausgang einen andern Wert als y erzeugen
  - x muss zur Beobachtbarkeit den negierten Fehlerwert besitzen
- Fehlerunterscheidung

$$y_\mu \oplus y_\kappa = 1$$ Testmuster muss unterschiedliche Werte erzeugen

2.2 Fehlerüberdeckungstabelle

- ist ein Fehler durch ein Testmuster beobachtbar, so erhält die Tabelle den Eintrag 1
- Begriffe
  - Menge aller Eingangsbelegungen (Tests): $T = \{t_\nu \mid \nu \in N\}$
    - N = Testnummernmenge
    - $|T| = |N| = 2^n$; n = Anzahl der Eingangsvariablen
  - Menge aller angenommenen Fehler: $F = \{f_\mu \mid \mu \in M\}$
    - M = Fehlernummernmenge
    - $m = |M| = |F|$; m = Anzahl der zu testenden Fehler

	f_1	f_2	f_3	f_4	f_5
t_0					1
t_1		1			1
t_2	1				1
t_4				1	1
t_5		1		1	1
t_6	1			1	1

- Definitionen
  - Test-Fehler-Relation R
    - $R = \{(t, f) \in T \times F \mid t_\nu R f_\mu\}$
    - Menge aller Paare $(t_\nu, f_\mu) \in T \times F$ in welchen der Test t_ν den Fehler f_μ beobachtbar macht
    - $t_\nu R f_\mu \in \{1,0\}$; Der Test t_ν steht in Relation zu den Fehler f_μ
  - Fehlererkennung: $t_\nu R f_\mu = y(\underline{x}_\nu) \oplus y_\mu(\underline{x}_\nu)$
  - Fehlergruppe:
    - $F_\nu = \{f_\mu \in F \mid t_\nu R f_\mu\}$
    - Menge aller Fehler, die durch den Test t_ν beobachtbar werden
  - Testgruppe:
    - $T_\mu = \{t\nu \in T \mid t_\nu R f_\mu\}$
    - Menge aller Tests, die den Fehler f_μ beobachtbar machen
  - Menge nicht unterscheidbarer Fehler
    - F_U
    - alle Fehler die eine identische Testgruppe enthalten
  - Mindesttestmenge zur Fehlererkennung
    - $C_{M,N} = \bigwedge_{\mu \in M} \bigvee_{\nu \in N} (t_\nu \in T) \cdot \tau_\nu = 1$
    - Schnitt aller Vereinigungen von Testwahlvariablen τ_ν für den jeweiligen Fehler f_μ
    - kürzeste Terme (Vereinigungen) markieren Mindestmenge $T_{c\min}$
  - Heuristische Minimierung mit Hilfe der Fehlerüberdeckungstabelle
    - suche fortwährend die mächtigste Fehlermenge und füge deren Test zur Testmenge hinzu
    - streiche alle abgedeckten Fehler

3. Testbestimmung in Schaltnetzen

3.1 Die Boolesche Differenz

- Eingangsbelegung: $\underline{x}$ Testvariable: z Testpunkt: $y(z(\underline{x}), \underline{x})$
- Entwicklungssatz: $y = y_z \cdot z \oplus y(z = 0)$
- Boolesche Differenz: $y_z = y(z, \underline{x}) \oplus y(\bar{z}, \underline{x})$ $y_z = y(z = 1) \oplus y(z = 0)$
- Berechnung der Testbelegung:
  - z stuck at 0
    - $\underline{x} R\ z/0 = z \cdot y_z = 1$
  - z stuck at 1
    - $\underline{x} R\ z/1 = \bar{z} \cdot y_z = 1$
  - $\underline{x}$ muss so jeweils so gewählt werden, dass obige Gleichung erfüllt ist
  - Überprüfung der Einstellbarkeit (Fehlerbelegung)
    - es muss gelten $z(\underline{x}) = 1$ für z/0
    - es muss gelten $z(\underline{x}) = 0$ für z/1
  - Überprüfung der Beobachtbarkeit (Sensibilisierungsbelegung)
    - es muss gelten $y_z(\underline{x}) = 1$

- Umrechung von Logikterme in das AND-EXOR-System (OR bzw. NOT stören)
  - $\overline{xy} = xy \oplus 1$
  - $\overline{x+y} = \bar{x} \cdot \bar{y} = (x \oplus 1)(y \oplus 1) = xy \oplus x \oplus y \oplus 1$
  - $x + y = x \oplus y \oplus xy$
  - $\bar{x} = x \oplus 1$
- Rechenregeln mit Booleschen Differenzen

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

2.) $y_y = 1$

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

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

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

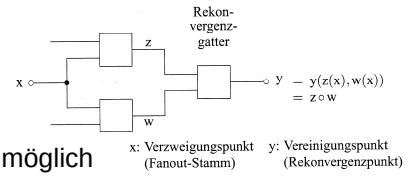
6.) $(z + w)_x = \bar{z} w_x \oplus z_x \bar{w} \oplus z_x w_x$

7.) $y_x = y_z z_x$ falls $y = y(z(x))$

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

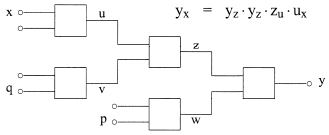
3.2 Strukturbezogene Berechnung der Booleschen Differenz

- lokale Sensibilität: $y_z = (\bar{z} \circ w) \oplus (z \circ w)$ bzw. $y_w = (z \circ \bar{w}) \oplus (z \circ w)$
- globale Sensibilität: $y_x = \underbrace{y_z z_x \bar{w}_x}_{\text{1.Einfachfehlerpfad}} + \underbrace{y_w w_x \bar{z}_x}_{\text{2.Einfachfehlerpfad}} + \underbrace{z_x w_x}_{\text{Mehrfachfehlerpfad}} \cdot \underbrace{[\bar{z} \circ \bar{w} \oplus z \circ w]}_{\text{Selbstmaskierung} -> = 0}$
- Schaltnetze mit Rekonvergenzmaschen
  - es kann zur Selbstmaskierung kommen
  - Zwischenvariablen daher nicht immer automatisch mitgetestet

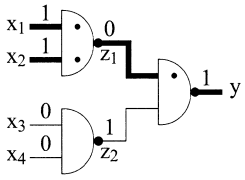


- Schaltnetze mit Baumstruktur
  - keine Mehrfachpfadsensibilisierung und keine Selbstmaskierung möglich
  - Berechnung der globalen Sensibilität mithilfe der Kettenregel
  - Kettenregel: $y_x = y_z z_x$
  - durch testen aller Eingangsvariablen werden alle Zwischenvariablen mitgetestet

3.3 Fehlersimulation



- Aufgabenstellung: Testmuster gegen, getestete Fehler gesucht
- Gutsimulation: Berechne alle Signale für fehlerfreies Schaltnetz
- Beobachtbarkeitsanalyse:
  - Simulation der Fehler an den Fanout-Stämmen (Selbstmaskierung…!)
  - Entfernung der Fanout-Stämme
  - Fehlerbaumkonstruktion
    - Aufbau von Ausgang zum Eingang
    - ist abhängig von der Signalbelegung der Gutsimulation
    - markiere Pfadsensibilisierung und sensiblen Pfad



- Menge der beobachtbaren Signale: $S^O = \{a, b, c, ...\}$; alle Signale der sensiblen Pfade
- Menge der getesteten Fehler: $F_t = \{a/0, b/1, ...\}$; soweit einstellbar bzgl. S^O

3.4 Der D-Algorithmus

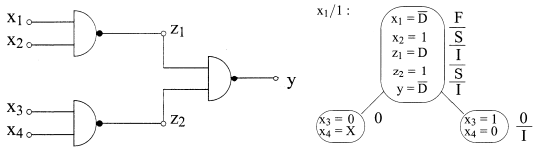
- 5-Wertige Logik:
  - X: undefiniert, unbekannt, nicht relevant
  - D: 1 im fehlerfreien Fall, 0 im fehlerhaften Fall
- Fehlerpfad $V_D = \{x, a, b, ..., y\}$
- Ablauf:
  - Initialisiere alle Signale mit X
  - Fehlerbelegung (F)
  - Sensibilisierung (S)
  - Implikationen (Vorwärts und Rückwärts)(I)
  - Optionale Wahl (O)
  - Backtracking: Suche nach Widersprüchen

·	0	1	X	D	$\bar{D}$
0	0	0	0	0	0
1	0	1	X	D	$\bar{D}$
X	0	X	X	X	X
D	0	D	X	D	0
$\bar{D}$	0	$\bar{D}$	X	0	$\bar{D}$

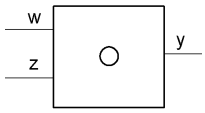
+	0	1	X	D	$\bar{D}$
0	0	1	X	D	$\bar{D}$
1	1	1	1	1	1
X	X	1	X	X	X
D	D	1	X	D	1
$\bar{D}$	$\bar{D}$	1	X	1	$\bar{D}$

⊕	0	1	X	D	$\bar{D}$
0	0	1	X	D	$\bar{D}$
1	1	0	X	$\bar{D}$	D
X	X	X	X	X	X
D	D	$\bar{D}$	X	0	1
$\bar{D}$	$\bar{D}$	D	X	1	0

a	$\bar{a}$
0	1
1	0
X	X
D	$\bar{D}$
$\bar{D}$	D



- Beschleunigungstechniken
  - Globale Implikation
    - Voraussetzung: keine Fehlereffektauswirkungen in benachbarter Teilschaltung
    - Kontrapositionsgesetz:
      - $(P \Rightarrow Q)\ \ gdw\ (\neg Q \Rightarrow \neg P)$
      - $(a=0) \Rightarrow (f=0)\ \ gdw\ \ (f=1) \Rightarrow (a=1)$
    - Lernkriterium:
      - Globale Implikation ist nur lernenswert wenn nicht durch lokale Implikation ausführbar und wenn gilt: $y_w \cdot y_z = 1$
      - Globale Rückwertsimplikation ist nur lernenswert, wenn Rekonvergenzmasche vorhanden ist



  - Auswahlbewertung für optionale Wertzuweisungen
    - C_0 = Nulleinstellbarkeit (Wahrscheinlichkeit)
    - C_1 = Einseinstellbarkeit (Wahrscheinlichkeit)
    - AND (IN: a, b -> OUT: c): OR (IN: a, b -> OUT: c):

$$C_1(c) = C_1(a) \cdot C_1(b) \qquad C_0(c) = C_0(a) \cdot C_0(b)$$
$$C_0(c) = 1 - C_1(c) \qquad C_1(c) = 1 - C_0(c)$$

    - es wird bei optionaler Wahl immer das Einstellmaß mit dem höchsten Wert ausgewählt

3.5 Testmustergenerierung bei sequentiellen Schaltungen

- Bestimmung der Zusammenhänge der sequentiellen Schaltung
  - Zustandsvariablen mit Zeit als Superskript kennzeichnen
  - Zeitdifferenz von einer Zeiteinheit vom Eingang zum Ausgang eines FFs
  - Bestimmung der Ausgangsvariable y der Schaltung in Abhängigkeit der Eingangsvariablen (möglichst keine Zwischensignale s oder z!)
- Bestimmung der Testbelegung ohne Boolesche Differenz
  - Bestimmung von y_μ durch Einsetzen des Fehlers in y
  - Bestimmung der Testbelegung durch $y \oplus y_\mu = 1$
  - Ziel: SOP
- Bestimmung der Testbelegung mit Boolescher Differenz
  - Bestimmung der Booleschen Differenz (Sensibilisierung des Fehlers)
  - Bestimmung der Testbelegung wie üblich
  - Ziel: SOP
- Bestimmung der Testbelegung mittels D-Algorithmus
  - Achtung: Implikationen auch über Zeiteinheiten hinweg möglich; d.h. es sind auch Implikationen in die Zukunft oder Vergangenheit möglich
- Schreibweisen
  - $X = \begin{matrix} x_1 \\ x_2 \end{matrix} \begin{bmatrix} 1 & X & 1 \\ 0 & 1 & 1 \end{bmatrix}$ (Spalten: $t-1$, t, $t+1$) Eingangsfolge
  - Y Ausgangsfolge ohne Fehler (aus Schaltung ermittelbar!)
  - Y_μ Ausgangsfolge mit Fehler

Formelsammlung DS

1. Moore'sches Gesetz

- alle 18 - 24 Monate verdoppelt sich die Anzahl der Transistoren auf gleicher Fläche
- Positive Seiten: Exponentielles Wachstum der Transistorzahl, exponentieller Rückgang des Preises pro Transistor
- Herausforderungen:
  - Herstellungskosten (Fixkosten, Variable Kosten, Technologiefaktor)
  - Entwicklerproduktivität (Problem des productivity gap)
  - Verlustleistungsdichte

$$n = \frac{d_{Wafer} \cdot \pi}{\sqrt{A_{Dies}}}$$

Unvollständige Dies (Abschätzung)

$$Ausbeute\,(yield) = \left(1 + \frac{D_f \cdot A_{Die}}{\alpha}\right)^{-\alpha}$$

D = Fehlerdichte, α = Technologiefaktor

2. Zahlensysteme und Signaldarstellung

- Polyadische Zahlensysteme:

  p = Anzahl der Ziffern rechts vom Basispunkt
  n = Anzahl der Ziffern links vom Basispunkt
  r = Basis, Radix
  d_i = Ziffer der i-ten Stelle

$$Z = \sum_{i=-n}^{p-1} r^i \cdot d_i$$

- Bekannte Zahlensysteme: Binär (Radix = 2), Oktal (Radix = 8), Hexadezimal (Radix = 16)
- Zahlenkonvertierung (Dezimal -> Radix n):
  - Systematisch (i > 0): Dezimalzahl durch Radix teilen, Rest ergibt Ziffer (LSB oben)
  - Systematisch (i < 0): Dezimalzahl mit Radix multiplizieren, Ergebnis - Radix = Ziffer (MSB oben)
- Zahlenkonvertierung (Radix n -> Dezimal): obige Formel verwenden
- Binäre Addition: wie bei Dezimalzahlen
- Negative Zahlen mit Vorzeichenbetrag: -> Aufwendige Logik
- Darstellung Negativer Zahlen mittels **Zweier-Komplement**:
  - Komplementbildung: alle Stellen invertieren
  - Addition von 1
  - MSB = Vorzeichenbit

$$K(Z) = r^n - Z = (r^n - 1) - Z + 1$$

- Zweier-Komplement -> Zahl in Betrag
  - Komplementbildung
  - Addition von 1
- Binäre Subtraktion: wie Addition, nur gültigen Stellenbereich auswerten
- Binäre Multiplikation (Addition- und Shiftoperation):
  - Systematik wie bei Dezimalzahlen
  - Multiplikation von links nach rechts -> partielle Produkte von links nach rechts verschieben
  - Multiplikation von rechts nach links -> partielle Produkte von rechts nach links verschieben
- Binäre Division (Subtraktion- und Shiftoperation)
  - Systematik wie bei Dezimalzahlen
  - gehe n - Stellen nach links bis zahl abziehbar ist, rechne mit Rest weiter

- Gleitkommadarstellung(GKD) für Binärsystem (IEEE 754):
  - Darstellung einer Gleitkommazahl (GKZ): **v e m**
  - Vorzeichen v (0 = positiv; 1 = negativ)
  - Exponent e (mit Bias addiert zum Vermeiden von negativen Zahlen; 8-bit)
  - Matisse m (enthält Wert zwischen 1 und 2 -> m = Nachkommazahl mit 23-bit)
  - Formel zur Berechnung einer Dezimalzahl aus der GKD:

$$Z = (-1)^v \cdot m \cdot 2^e$$

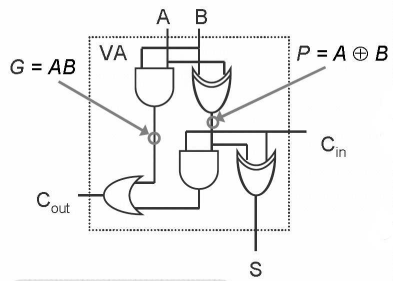
    - e muss durch Subtraktion der Bias = 127 ermittelt werden
    - m enthält nur die Nachkommazahl; 1. muss vorgestellt werden
  - Formeln zur Berechnung einer GKZ einer Dezimalzahl:

$$m = \left(\frac{|Z|}{2^e} - 1 \right) \cdot 2^{23} \qquad e = \lfloor \log_2 (|Z|) \rfloor$$

    - bei der Bestimmung von e muss zu e ein Bias von 127 hinzuaddiert werden (Definition! Kein Zweierkomplement)
    - bei der Bestimmung von m kann m direkt zur Binärzahl umgerechnet werden
  - Besonderheiten: e = 0 -> Z = 0; e = 0xFF -> Z = unendlich
  - Nachteile: Rundungsfehler (treten vor allem bei Subtraktion fast gleicher Zahlen auf)
- Analog - Digital Wandler (ADC):
  - Analog: wertkontinuierlich + zeitkontinuierlich
  - Digital: wertdiskret und zeitdiskret
  - Filtern = Begrenzung des Frequenzbereichs nach oben ($f < f_{max}$)
  - Abtasten ($f_{Sample} \geq 2 \cdot f_{signal,\max}$)
  - Quantisierung der analogen Signalamplitude (wertkontinuierlich -> wertdiskret)
  - Quantisierung: $B = f(V_{in} / V_{\mathrm{Ref}});\ V_{in} \leq V_{\mathrm{Ref}}$
  - Quantisierungsschritt: $\Delta = 2^{-N} \cdot V_{\mathrm{Ref}}$
  - Quantisierungsfehler: $\pm 0.5\Delta$
- Digital - Analog Wandler (DAC):
  - Wie oben nur in umgekehrter Reihenfolge

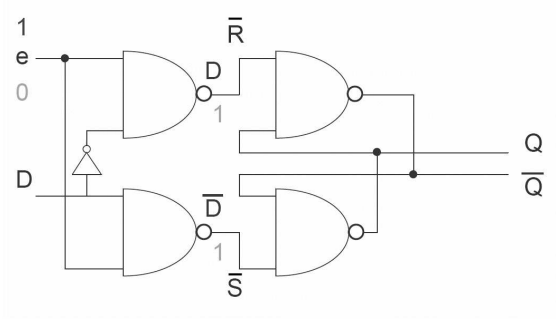
3. Kombinatorische Logik

- Boolsche Algebra: Boolsche Rechenregeln
- Logische Operationen auf Bitvektoren werden Bitweise durchgeführt
- Disjunktive Normalform: ODER auf oberer Ebene; UND und NOT auf unterer Ebene
- Konjunktive Normalform: UND auf oberer Ebene; ODER und NOT auf unterer Ebene
- Karnaugh Tabelle:
  - Kopfzeile und Kopfspalte immer aufteilen, sodass der Belegungsabstand eins beträgt
  - Produkttermminimierung durch Zusammenfassung benachbarter Felder
- Optimierung von Schaltungen durch reuse von Schaltnetzen
- Logik Zeitverhalten: Verzögerungen = propagation delay (50%in -> 50%out)
- Volladdierer: IN: A, B, C_{in} -> OUT: S, C_{out}
- Ripple-Carry-Adder: Serielle Verschaltung von Volladdierern
- Multiplexer (MUX): IN: A, B, S -> OUT: Z
- Bitwort-Schiebeoperator: Verschaltung von Multiplexern
  - erste Spalte: Verschiebung um eins wenn S = 1
  - zweite Spalte: Verschiebung um zwei wenn S = 1 usw.
- Vergleichsoperatoren:
  - Vergleich auf ≥ und > mit Hilfe eines Subtrahierers
  - Vergleich auf Gleichheit mittels EOR und OR



$$S = P \oplus C_{in}$$
$$C_{out} = G + P \cdot C_{in}$$

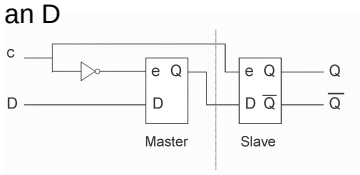
4. Sequentielle Logik



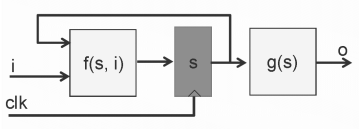
- Basisspeicherzelle: Ring aus Invertern
- Set-Reset-Latch: IN: S, R -> OUT: Q, Q'
  - Set durch S = 1
  - Reset durch R = 1
  - S = 1 und R = 1 sind nicht erlaubt
- Enable-Latch: IN: e, D -> OUT: Q, Q'
  - Enable durch E = 1 (Pegel der an D anliegt wird gespeichert)
  - Besteht aus einem RS-Latch mit zusätzlicher Eingangslogik (2 NAND - Gatter)

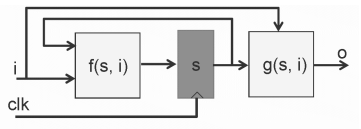
➢ Latches sind Pegelgesteuert (level-controlled) und NICHT Flankengesteuert

- Flip-Flop (FF): IN: c, D -> OUT: Q, Q'
  - taktflankengesteuert: low-high-Flanke an c führt zur Speicherung des an D anliegenden Pegels
  - Verschaltung von zwei Enable-Latch (Master-Slave-Schaltung)
  - Master speichert D im Low-Pegel von c
  - Slave speichert Ausgangspegel vom Master in der High-Phase von c



- Register: Zusammenfassung mehrer FFs
- Flip-Flop-Timing:
  - t_{setup} : Zeitintervall vor der Flanke, in welchem D konstant anliegen muss
  - t_{hold} : Zeitintervall nach der Flanke, in welchen D konstant anliegen muss
  - t_{c2q} : Latenzzeit des FFs, Verzögerung zwischen Ein- und Ausgang
- Sequentielles Schaltwerk
  - Schaltwerk mit Gedächtnis: $O = S(I,t)$
  - Die Ausgabe hängt von der Eingabe zu diskreten Zeiten ab
- Endliche Automaten (FSM):
  - Moore-Automat
  - Mealy-Automat (In der Praxis zu vermeiden!)
- Moore-Automat:
  - $s' = f(s,i)$; $o = g(s)$; i = Input; o = Output; s' = Folgezustand;
  - $f : S \times I \rightarrow S$; Übergangsrelation, wird mit kombinatorischen Schaltnetzen realisiert
  - $g : S \rightarrow O$; Ausgangsrelation, wird mit kombinatorischen Schaltnetzen realisiert
  - s wird in einen Register gespeichert
  - Vorteil: kurze kombinatorische Pfade (kein Pfad Eingang -> Ausgang)
  - Nachteil: hohe Anzahl von Zuständen
- Mealy-Automat:
  - $s' = f(s,i)$; $o = g(s,i)$;
  - $g : S \times I \rightarrow O$; Ausgangsrelation
  - Vorteile: übersichtlich, wenige Zustände
  - Nachteil: lange kombinatorische Pfade



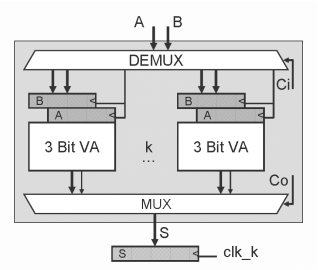


- Separation von Kontroll- und Datenpfad:
  - Datenpfad: Transformation der Daten
  - Kontrollpfad: Steuerung der Einheiten im Datenpfad (meist mit FSM)
- Timinganalyse sequentieller Schaltungen
  - Kritische Pfade: längster und kürzester Pfad

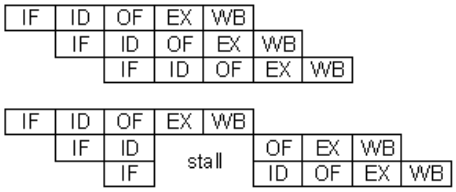
$$t_{clk} \geq t_{c2q} + t_{setup} + \sum_{längsterPfad} t_{gate}$$

- Fließbandverarbeitung Pipelining:
  - Einteilung der Logik in mehrere gleich große Teillogiken
  - Längster Pfad in einer Teilfunktion bestimmt t_{clk}
  - ➢ Taktfrequenz kann erhöht werden
  - ➢ Latenzzeit kann zunehmen!

$$t_{hold} \leq t_{c2q} + \sum_{kürzester\ Pfad} t_{gate}$$



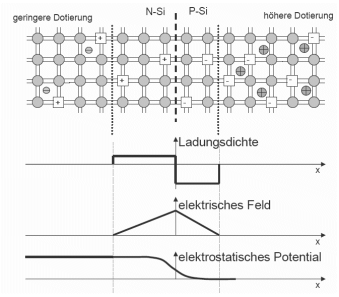
- Parallele Verarbeitung:
  - o kombinatorische Logik in mehrfacher Ausführung
  - o Kontrolllogik zum Verteilen der Signale (DEMUX -> MUX)
- RISC (= reduced instruction set computer)
  - o Prozessorbestandteile:
    - Rechenwerk (ALU)
    - Registerbank (Register File)
    - L1 Cache
    - Steuerwerk
  - o Bussystem:
    - I/O
    - L2 Cache
    - RAM
  - o einfaches Instructionset
  - o ALU-Operationen nur auf Register anwendbar
  - o separate Instruktionen für Speicherzugriff
  - o Instruktionsphasen:
    - IF (= instruction fetch)
    - ID (= instruction decode)
    - OF (= operand fetch)
    - EX (= execute)
    - WB (= write back)



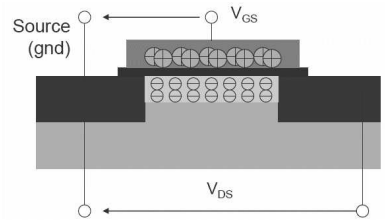
  - o die Instruktionsphasen werden durch pipelining in der Ausführung beschleunigt
  - ➢ CPI ist nahezu gleich eins (CPI = clocks per instruction)
  - o MIPS = million instuctions per second
  - o stall = Lehrlauf der pipeline durch warten auf ein Ergebnis der vorhergehenden Operation

5. MOSFET Transistor

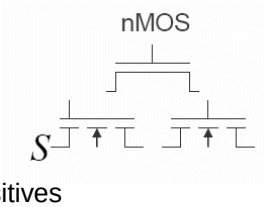
- undotierter Halbleiter
  - o Ladungsträger: e^- und „Löcher“ (entstehen durch thermische Generation)
  - o Leitfähigkeit nimmt mit zunehmender Temperatur zu
  - o z.B. Si (4-wertig)
- n-dotierter Halbleiter
  - o Ladungsträger: e^-, hohe Leitfähigkeit
  - o nach außen hin neutral geladen
  - o Dotierung mit Atomen der Wertigkeit 5
- p-dotierter Halbleiter
  - o Ladungsträger: „Löcher“, mittlere Leitfähigkeit
  - o nach außen hin neutral geladen
  - o Dotierung mit Atomen der Wertigkeit 3



- pn-Übergang
  - o hohe p-Dotierung, geringere n-Dotierung (-> Raumladungszone weiter im n-Bereich)
  - o e^- und „Löcher“ diffundieren über pn-Grenze
  - ➢ Raumladungszone mit Elektrischen Feld
  - ➢ verhindert weiteres diffundieren
  - o Betrieb in Sperrrichtung -> Raumladungszone vergrößert sich (- an p-Dot, + an n-Dot)
  - o Betrieb in Flussrichtung
    - ➢ E-Feld der Raumladungszone muss überwunden werden
    - ➢ Fluss der Ladungsträger (+ an p-Dot, - an n-Dot)
- MOS-Struktur (MOS = metal oxid semiconductor)
  - o p-dotierte Hauptschicht, SiO_2 als Isolator zwischen Halbleiter und Gate-Elektrode
  - o Verarmung: $V_{gs} < V_t$ (V_t = Thresholdspannung, Spannung zum Lösen der e^-)
  - ➢ durch negatives Potential am Gate werden e^- von der Gate verdrängt
  - o Inversion: $V_{gs} > V_t$
  - ➢ durch positives Potential am Gate werden e^- unter der Gate angereichert



- n-MOS-Transistor
  - Source und Drain n-doteriert
  - Ausbildung eines n-Kanals, wenn $V_{gs} > V_t$
  - V_{gs}, $V_{ds} > 0$, Source immer am niedrigeren Pot.
  - Eingangskennlinie: Parabellast, um V_t nach rechts verschoben
  - Einteilung in Arbeitsbereiche:
    - Sperrbereich: $V_{gs} < V_t$ $I_{Dn} = 0$
    - Linearbereich: $V_{gs} > V_t$ und $0 < V_{ds} < V_{gs} - V_t$
    - Sättigungsbereich (Pinch-Off): $V_{gs} > V_t$ und $V_{gs} - V_t < V_{ds}$



  - Pinch-Off: n-Kanal wird in Richtung Drain dünner, da Drain ein höheres positives Potential als das Gate besitzt
  - Pinch-Off-Potential: $V_{Pinch\text{-}Off} = V_{gs} - V_t$
  - Formel für den Drain-Strom des n-Kanals im Linearbereich:

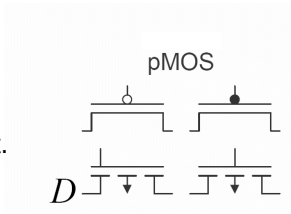
$$I_{Dn} = \beta\left(V_{GS} - V_t - \frac{V_{DS}}{2}\right)V_{DS} \qquad C_G = \varepsilon_{Ox}\varepsilon_0 \frac{WL}{t_{Ox}} \qquad \mu = -\frac{dx}{dV}\cdot v$$

  - Formel für den Drain-Strom des n-Kanals im Sättigungsbereich:

$$I_{Dn} = 0{,}5\beta\left(V_{GS} - V_t\right)^2 \qquad \beta = \frac{C_G\mu}{L^2} = \frac{\mu\varepsilon_{Ox}\varepsilon_0}{t_{Ox}}\cdot\frac{W}{L} = K'\cdot\frac{W}{L} \qquad I = \rho v = \frac{dQ}{dx}\cdot v$$

    C_G = Gate-Kapazität bei einer Oxiddicke von t_{Ox} und einer Fläche A = WL
    μ = Ladungsträgerbeweglichkeit
    L = DS-Abstand

  - Technologieparameter: μ, L_{min}, ε_{ox}, t_{ox}
  - Design Parameter: W



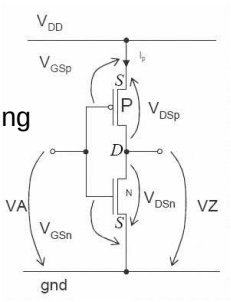
- p-MOS-Transistor
  - Drain und Source p-dotiert, V_{gs}, $V_{ds} < 0$, Source immer am höheren Pot.
  - Einteilung in Arbeitsbereiche:
    - Sperrbereich: $V_{gs} > V_t$ $I_{Dp} = 0$
    - Linearbereich: $V_{gs} < V_t$ und $V_{gs} - V_t < V_{ds} < 0$

$$I_{Dp} = -\beta\left(V_{GS} - V_t - \frac{V_{DS}}{2}\right)V_{DS}$$

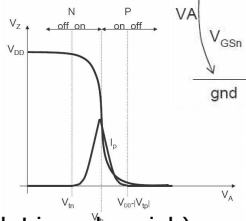
    - Sättigungsbereich (Pinch-Off): $V_{gs} < V_t$ und $V_{ds} < V_{gs} - V_t$

$$I_{Dp} = -0{,}5\beta\left(V_{GS} - V_t\right)^2$$

6. CMOS Inverter und Logik



- Vorteile der CMOS-Technologie (technisch)
  - Geringe Verlustleistung / Störungsunempfindlichkeit / nur eine Stromversorgung
  - saubere Logikpegel (Noise Margin) / Kaskadierbarkeit
- Vorteile der CMOS-Technologie (wirtschaftlich)
  - einfacher Entwurf / hoch integrierbar / ausgereifte Herstellung
- CMOS-Inverter
  - p-Dotierte Schicht als Träger
  - Statische Spannungs-Übertragung-Kennlinie(VTC)



- Lastkapazität und Verzögerungszeit (propagation delay)

$t_p = R_{on,p}C\left|\ln(0{,}5)\right|$ $R_{on,\,p}$ = Einschaltwiderstand des p-MOS (Modell: Linearbereich)

- Spannungspegel und Verzögerungszeit

$$R_{on,p} = \frac{V_{DSp}}{I_{Dp}} \approx \frac{1}{\beta \cdot (|V_{GSp}| - |V_{tp}|)} \qquad \text{für } |V_{DSp}| << (|V_{GSp}| - |V_{tp}|)$$

$$t_{pHL} \propto \frac{C_{load} t_{ox} L_p}{W \mu_p \varepsilon_{ox} (V_{DD} - |V_{tp}|)} \qquad \propto : \text{direkt proportional}$$

V_{tp} : manipulierbar durch t_{ox}, Kanaldotierung und Substratspannung V_{bulk}

- CMOS Verlustleistung
  - Dynamische Verlustleistung (50% der gesamten Verlustleistung)
    - Schalthäufigkeit α_{01}: $P = \alpha_{01} f_{clk} E$ E: Energie pro Schaltvorgang
    - Kapazitive Verlustleistung = Wärme an $R_{on,p}$ + Energie der Ladung auf C

$$P_{Cap} = \alpha_{01} f C V_{DD}^2$$

    - Kurzschlussverlustleistung

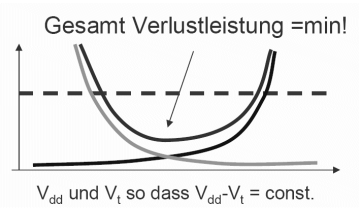
$$P_{Short} = \alpha_{01} f \beta_n \tau (V_{DD} - 2V_{tn})^3 \qquad \tau : \text{Rise-Fall-Time}$$

    - Verlustleistung pro Gatter sinkt pro Technologie Skalierung
    - Verlustleistung pro Chip/Fläche steigt
  - Statische Verlustleistung
    - Sub-Schwellströme (Ströme durch die n/p-Kanäle bei $V_{GS} \leq V_t$)
      - nicht ideale Ausgangspannungen vorheriger Gatter
      - Rauschen / kapazitive Kopplungen

$$I_D = I_0 e^{(V_{GS} - V_t)/nV_{Temp}} (1 - e^{-V_{DS}/V_{Temp}}) \qquad V_{GS} \leq V_t$$

$$I_0 = I_D(V_{GS} = V_t); \qquad \text{n: Prozesskonstante (1-2,5)}$$

$$I_D \approx I_0 e^{V_t/nV_{Temp}} \text{ für } V_{GS} = 0$$



    - Diodenleckstrom / Gate-Strom
      - Leckströme ins Substrat verursachen Verlustleistung
      - Gate-Oxid isoliert nicht perfekt
      - $I_{Gate} \propto \exp(t_{ox}^{-1})$
    - früher: Statische Verlustleistung vernachlässigbar (< 1% von P_{gesamt})
    - wird in Zukunft immer größere Rolle spielen
    - Subschwellstrom nimmt mit steigender V_t ab
- CMOS-Logik
  - Immer invertierende Logik, da n-MOS immer direkt an GND liegen muss
  - Seriellschaltung von n-MOS für logische UND
  - Parallelschaltung von n-MOS für logische ODER
  - im p-MOS Block wird die serielle Schaltung von n-MOS in eine parallele p-MOS Schaltung konvertiert
  - gleiches gilt für n-MOS parallel -> p-MOS seriell
- Fan-In und Fan-Out
  - Gatter mit hohem Fan-Out sind langsamer
  - Gatter mit hohem Fan-In sind größer und langsamer

7. Speicher

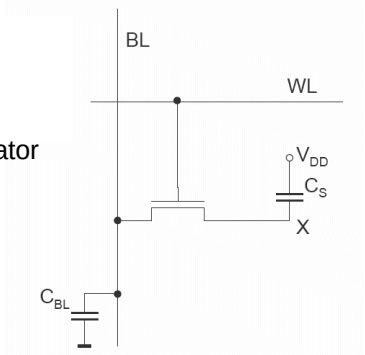
- Klassifizierung
  - Random Access RAM
    - SRAM (static random access memory)
    - DRAM (dynamic random access memory)
    - Register

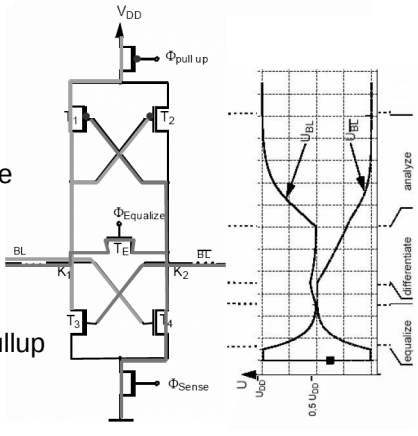
  - Non-Random Access RAM
    - FIFO (first in first out) / LIFO (last in first out)
    - CCD (charge coupled device); ähnlich einem Schieberegister
    - CAM (content addressable memory); inhaltsaddressierbarer Speicher
    - Shift-Register
  - Non-Volatile Read Write Memory
    - EPROM (Erasable Programmable Read-Only-Memory)
    - EEPROM (Electrically Erasable Programmable Read-Only Memory)
    - Magneto RAM (Widerstandsänderung durch Magnetfelder)
  - Read Only Memory
    - Mask-programmed / Fuse-programmed

- Aufbau von Speichern
  - Bandbreite [bit/sec]
  - Latenz: Zeitdifferenz zwischen Ein- und Ausgabe
  - Zykluszeit: Zeitdifferenz zwischen aufeinander folgende Schreib- und Lesezyklen
  - Asynchrone Speicher: Lesen / Schreiben mit anlegen der Adresse bzw. der Steuersignale
  - Synchroner Speicher: R/W-Operationen an Takt gebunden

1k	1024	2^{10}
1M	1024x1024	2^{20}
1G	1024x1024x1024	2^{30}

- Speicherarchitektur: Array-Struktur
  - Row Decoder / Column Decoder
  - Leseverstärker
  - adressiert wird immer ein der Wortleitungsbreite entsprechendes Datenwort
- Speicherarchitektur: Hierarchie
  - Gliederung der Speicherarrays zusätzlich in Blöcke
  - kürzere Verdrahtung -> schneller
  - nur einzelne Blöcke sind aktiv -> geringerer Verlustleistung
- 1-Transistor DRAM Zelle
  - Speicherzelle besteht aus einem Transistor und einem Kondensator
  - hohe Speicherdichte / symmetrische Zugriffszeit
  - Bitleitung wird beim lesen auf $0{,}5V_{DD}$ vorgeladen
  - Bitleitungskapazität wesentlich höher als Speicherkapazität
  - geringer Signalhub ΔV beim Lesen -> Leseverstärker notwendig
  - $\Delta V = \frac{(V_X - 0{,}5V_{DD})\ C_S}{C_S + C_{BL}}$
  - Speicherkondensator
    - Leiter (Polysilicium) mit V_{DD} verbunden
    - Isolator (SiO_2)
    - p-Si mit Transistor verbunden (X in der Graphik)
    - Trench Cell: Vergrößerung von C_S durch Bau in die Tiefe
    - Stacked-Capacitor Cell: fächerförmiger Bau in die Höhe
  - Leseverstärker
    - Positionierung in der Mitte eines Speicherfeldes
    - ➢ halbe Bitleitungskapazität
    - Lesen: Equalize, Aktivierung der Wortleitung, Sense, Pullup





- CMOS-SRAM (6 Transistoren)
  - zwei rückgekoppelte CMOS-Inverter
  - über jeweils einen n-MOS mit BL und $\overline{BL}$ verbunden
- FLASH-Speicher
  - einfügen eines floating gate in die Oxidschicht eines n-MOS Transistor
  - ‚0' speichern: 4mal V_{DD} an G und D, GND an S
  - ‚0' löschen: S von GND trennen, G an GND und D an 4mal V_{DD}
- ROM-Speicher
  - fuse programming; Programmierung ist irreversibel