Digitaltechnik

1 Moore'sches Gesetz

- alle 18-24 Monate verdoppelt sich die Anzahl der Transistoren auf gleicher Fläche
- Exponentielles Wachstum der Transistorzahl, exponentieller Rückgange des Preises pro Transistor
- Herstellungskosten (Fixkosten, Variable Kosten, Technologiefaktor), Entwicklerproduktivität, Verlustleistungsdichte

2 Einheiten

Potenz	Vorsatz
10^{12}	T
10^{9}	G
10^{6}	M
10^{3}	k
10^{2}	h
10^{1}	da

Potenz	Vorsatz
10^{-1}	d
10^{-2}	c
10^{-3}	m
10^{-6}	μ
10^{-9}	n
10^{-12}	p
10^{-15}	f

Hz	s^{-1}
N	$kgms^{-2}$
J	$Nm = VAs$
W	$VA = Js^{-1}$
C	As
V	JC^{-1}
F	CV^{-1}
Ω	VA^{-1}
H	VsA^{-1}

$$Bit \xrightarrow{\cdot 8} Byte \xrightarrow{\cdot 1024} kByte \xrightarrow{\cdot 1024} MByte$$

3 Polyadische Zahlensysteme

$Z = \sum_{i=-n}^{p-1} r^i \cdot d_i = d_{p-1}...d_1 d_0 . d_{-1}...d_n$

Z:Zahl, r:Basis, d_i:Ziffer, p:#Ziffern vorne n:#Nachkommastellen

Binäres Zahlensystem:

$d_{i2} \in 0,1 \qquad B = \sum_{i=-n}^{p-1} 2^i \cdot d_i \quad d_{-n} : LSB; \quad d_{p-1} : MSB$

Octalsystem:	Hexadezimalsystem:
$d_{i8} \in 0,1,2,3,4,5,6,7$	$d_{i16} \in 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F$

Benötigte Bits: $N : n$ Bit. $M : m$ Bit
$N + M : \max\{n, m\} + 1$ Bit
$N \cdot M : n + m$ Bit

3.1 Umrechnung

	$Z \geq 1$	$Z < 1$
$r \to 10$	$Z_{10} = \sum r^i \cdot d_i$ $101_2 \to 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1$	$Z_{10} = \sum r^{-i} \cdot d_{-i}$ $0.11_2 \to 1 \cdot 0.5 + 1 \cdot 0.25$
$10 \to r$	$d_i = Z_{10} \% r^i$ ($d_i = Z_{10} \bmod r^i$) $58/8 = 7$ Rest $2(LSB)$ $7/8 = 0$ Rest $7(MSB)$ (Ende wenn 0 erreicht) Auf Ende achten $1r3\%5 \to 0r1$	 $0.4 \cdot 2 = 0.8$ Übertrag $0(MSB)$ $0.8 \cdot 2 = 1.6$ Übertrag 1 (Wiederholen bis 1 oder Periodizität)

3.2 Zweierkomplement **Wertebereich:** $-2^{n-1} \leq Z \leq 2^{n-1} - 1$

$Z \to -Z$ (Umkehrung gleich)	Bsp: Wandle 2 in -2 um
1. Invertieren aller Bits	$0010 \Rightarrow 1101$
2. Addition von 1	$1101 + 1 = 1110$
3. Ignoriere Überträge beim MSB	$\Rightarrow -2_{10} = 1110_2$

3.3 Gleitkommadarstellung nach IEEE 754

Bitverteilung(single/double):

$s(1)$	$e(8/11)$	$f(23/52)$

s: Vorzeichen, e: Exponent, f: Mantisse (**Nachkommastellen!** $2^{-1}2^{-2}...$)

Spezialwerte: $Z = 0 \Leftrightarrow e = 0 \qquad Z = +(-)\infty \Leftrightarrow e = 255, s = 0(1)$

IEEE $\to$ **Wert** Z $Z = (-1)^s \cdot (1 + 0.f) \cdot 2^{e-127}$	Bsp: $s = 1$, $e = 126$, $f = 01_2$ $Z = -1 \cdot 2^{-1} \cdot 1.01_2 = -0.101_2 = -0.625$
Wert Z $\to$ **IEEE** (Binärdarstellung) $s = 0$(positiv), $s = 1$(negativ) $Z \to Z_2$ (beim Komma teilen) Z_2 n-mal shiften $\to 1.xxx\ldots$ Exponent $e = n + 127 \to e_2$ Mantisse $f_2 = xxx\ldots$	Bsp: $Z = 11.25$ $s = 0$ $Z = 1011.01_2$ $Z = 1.01101_2 \cdot 2^3$ $e = 3 + 127 = 130 = 10000010_2$ $f = 01101000\ldots_2$
Wert Z $\to$ **IEEE** (Formel) $s = 0$(positiv), $s = 1$(negativ) $E = \lfloor \log_2 \lvert Z \rvert \rfloor$ $e = E + 127 \to e_2$ $f = \left(\frac{\lvert Z \rvert}{2^E} - 1\right) \cdot 2^{23} \to f_2$	Bsp: $Z = 11.25$ $s = 0$ $E = \lfloor \log_2 \lvert 11.25 \rvert \rfloor = \lfloor 3,49\ldots \rfloor = 3$ $e = 3 + 127 = 130 = 10000010_2$ $f = \left(\frac{\lvert 11.25 \rvert}{2^3} - 1\right) \cdot 2^{23} = 3407872 = 01101000\ldots_2$

3.4 Zahlenoperationen

Zahlensystem	Add	Sub	Mul	Div
Vorzeichenlos	Normal	Normal	Normal	Normal
Einser Komplement	Vorzeichen Vergleichen			
Zweier Komplement	Normal	Normal	$n + mbit$ Vordere Bits mit Multiplikant MSB auffüllen	
Float				

- Festkomma (Vorzeichenlos)
  - Erweiterung: Null vorne anhängen
  - Addition: Bitweise
  - Subtraktion: Bitweise
  - Multiplikation: Add-Shift (Add für jede 1 im Multiplikant) (Resultat eins rechts Shiften)
    Sonderfall: Multiplikation mit 2-er Potenz $\to$ um Potenz mal shiften.
  - Division:
- Festkomma (Einser Komplement)
  - Erweiterung: Null an Stelle 2 einfügen.
  - Addition:
    1. Prüfe Beide Vorzeichen
    2. Gleiches Vorzeichen $\to$ reguläre Addition
    3. Verschieden $\to$ Subtraktion kleiner Operator von großem Operator. Übernahme Vorzeichen des großen Operators.
- Festkomma (Zweier Komplement)
  - Erweiterung: 1 vorne anhängen
  - Addition: Regulär (Gleiche Parameterlänge) (Overflow ignorieren)
  - Subtraktion: Addition mit komplementiertem Subtraktor (Gleiche Parameterlänge)(Overflow ignorieren)
  - Multiplikation:
    1. Zahlen auf Produktlänge erweitern.
    2. Zahlen mittels Add-Shift multiplizieren (Überflüssige Bits nach links rausschieben und ignorieren)
- Gleitkomma (IEEE Float)
  - Addition: Exponenten auf größeren angleichen, Mantissen addieren. Vorzeichen inspizieren.
  - Subtraktion:
  - Multiplikation: Exponenten auf größeren angleichen, Mantissen multiplizieren. Vorzeichen multiplizieren.
    Sonderfall: Multiplikation mit 2-er Potenz $\to$ Potenz zu Exponent addieren.
  - Division:

4 Zeichenkodierung

4.1 ASCII

American Standard Code for Information Exchange
Fixe Codewortlänge (7 Bit, 128 Zeichen)
$0x00 - 0x7F$

4.2 UTF-8

Universal Character Set Transformation Format
Variable Codewortlänge (1-4 Byte) $\to$ Effizient

Schema

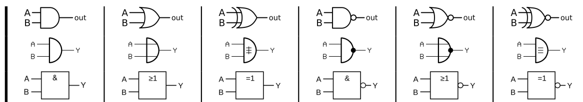
- MSB = 0 $\to$ 8 Bit (restliche Bit nach ASCII)
- MSB = 1 $\to$ 16, 24 oder 32 Bit
  - Byte 1: Die ersten 3, 4, 5 Bit geben die Länge des Codewortes an (110, 1110, 11110)
  - Byte 2-4: Beginnen mit Bitfolge 10

4.3 Zahlensysteme

Base 10	Base 2	Base 8	Base 16
00	0000	**0o**00	**0x**0
01	0001	**0o**01	**0x**1
02	0010	**0o**02	**0x**2
03	0011	**0o**03	**0x**3
04	0100	**0o**04	**0x**4
05	0101	**0o**05	**0x**5
06	0110	**0o**06	**0x**6
07	0111	**0o**07	**0x**7
08	1000	**0o**10	**0x**8
09	1001	**0o**11	**0x**9
10	1010	**0o**12	**0x**A
11	1011	**0o**13	**0x**B
12	1100	**0o**14	**0x**C
13	1101	**0o**15	**0x**D
14	1110	**0o**16	**0x**E
15	1111	**0o**17	**0x**F

5 Boolsche Algebra

5.1 Boolesche Operatoren (Wahrheitstabelle WT)



x	y	AND $x \cdot y$	OR $x + y$	XOR $x \oplus y$	NAND $\overline{x \cdot y}$	NOR $\overline{x + y}$	EQV $\overline{x \oplus y}$
0	0	0	0	0	1	1	1
0	1	0	1	1	1	0	0
1	0	0	1	1	1	0	0
1	1	1	1	0	0	0	1

Konfiguration: $f = c_1 + c_2 + c_3 \Rightarrow cov(f) = \{c_1, c_2, c_3\}$

5.2 Gesetze der booleschen Algebra

	Boolesche Algebra $(0,1;\cdot,+,\overline{x})$	Mengenalgebra $(P(G);\cap,\cup,\overline{A};G,\emptyset)$
Kommutativ	$x\cdot y=y\cdot x$	$A\cap B=B\cap A$
	$x+y=y+x$	$A\cup B=B\cup A$
Assoziativ	$x\cdot(y\cdot z)=(x\cdot y)\cdot z$	$(A\cap B)\cap C=A\cap(B\cap C)$
	$x+(y+z)=(x+y)+z$	$(A\cup B)\cup C=A\cap(B\cup C)$
Distributiv	$x\cdot(y+z)=x\cdot y+x\cdot z$	$A\cap(B\cup C)=(A\cap B)\cup(A\cap C)$
	$x+(y\cdot z)=(x+y)\cdot(x+z)$	$A\cup(B\cap C)=(A\cup B)\cap(A\cup C)$
Idempotenz	$x\cdot x=x$	$A\cap A=A$
	$x+x=x$	$A\cup A=A$
Absorption	$x\cdot(x+y)=x$	$A\cap(A\cup B)=A$
	$x+(x\cdot y)=x$	$A\cup(A\cap B)=A$
Neutral	$x\cdot 1=x$	$A\cap G=A$
	$x+0=x$	$A\cup\emptyset=A$
Dominant	$x\cdot 0=0$	$A\cap\emptyset=\emptyset$
	$x+1=1$	$A\cup G=G$
Komplement	$x\cdot\overline{x}=0$	$A\cap\overline{A}=\emptyset$
	$x+\overline{x}=1$	$A\cup\overline{A}=G$
	$\overline{\overline{x}}=x$	$\overline{\overline{A}}=A$
De Morgan	$\overline{x\cdot y}=\overline{x}+\overline{y}$	$\overline{A\cap B}=\overline{A}\cup\overline{B}$
	$\overline{x+y}=\overline{x}\cdot\overline{y}$	$\overline{A\cup B}=\overline{A}\cap\overline{B}$

5.3 Boolesche Funktionen

$$f:\{0,1\}^n\to\{0,1\}\qquad f(\underline{x})=f(x_1,x_2,\dots,x_n)$$

Einsmenge F von f: $F=\{\underline{x}\in\{0,1\}^n|f(\underline{x})=1\}$
Nullmenge $\overline{F}$ von f: $\overline{F}=\{\underline{x}\in\{0,1\}^n|f(\underline{x})=0\}$

Kofaktor bezüglich

- $x_i: f_{x_i}=f|_{x_i=1}=f(x_1,\dots,1,\dots,x_n)$
- $\overline{x}_i: f_{\overline{x}_i}=f|_{x_i=0}=f(x_1,\dots,0,\dots,x_n)$

Eigenschaften von $f(\underline{x})$

- tautologisch $\Leftrightarrow f(\underline{x})=1\quad\forall\underline{x}\in\{0,1\}^n$
- kontradiktorisch $\Leftrightarrow f(\underline{x})=0\quad\forall\underline{x}\in\{0,1\}^n$
- unabhängig von $x_i\Leftrightarrow f_{x_i}=f_{\overline{x}_i}$
- abhängig von $x_i\Leftrightarrow f_{x_i}\neq f_{\overline{x}_i}$

5.4 Multiplexer

$f=x\cdot a+\overline{x}\cdot b$ (2 Eingänge a,b und 1 Steuereingang x)
$f=\overline{x}_1\overline{x}_2a+\overline{x}_1x_2b+x_1\overline{x}_2c+x_1x_2d$ (Eingänge: a,b,c,d Steuerung: x_1, x_2)

5.5 Wichtige Begriffe

Wichtige Begriffe:	Definition	Bemerkung
Signalvariable	x	$\hat{x}\in\{0,1\}$
Literal	$l_i=x_i$ oder $\overline{x_i}$	$i\in I_0=\{1,\dots,n\}$
Minterme,0-Kuben	M0C $\ni m_j=\prod_{i\in I_0}l_i$	$\lvert$M0C$\rvert=2^n$
d-Kuben	MC $\ni c_j=\prod_{i\in I_j\subseteq I_0}l_i$	$\lvert$MC$\rvert=3^n$
Distanz	$\delta(c_i,c_j)=\lvert\{l\,\vert\, l\in c_i\wedge\bar{l}\in c_j\}\rvert$	$\delta_{ij}=\delta(c_i,c_j)$
Implikanten	$MI=\{c\in MC\,\vert\, c\subseteq f\}$ Terme, dessen Erfüllbarkeit identisch mit die der Formel sind	
Primimplikanten	$MPI=\{p\in MI\,\vert\, p\not\subset c\ \forall c\in MI\}$ Implikanten, die maximal freie Variablen besitzen	$MPI\subseteq MI\subseteq MC$
Kernprimimplikanten	Primimplikanten die für Überdeckung zwingend notwendig sind	Spalten mit 1 Eintrag in Überdeckungstabelle

DNF (DNF)	eine Summe von Produkttermen	Terme sind ODER-verknüpft
KNF (KNF)	ein Produkt von Summentermen	Terme sind UND-verknüpft
KDNF (KDNF)	Summe aller Minterme	WT: 1-Zeilen sind Minterme
KKNF (KKNF)	Menge aller Maxterme	WT: 0-Zeilen negiert sind Maxterme
VollSOP (nur 1)	Menge aller Primimplikanten	Bestimmung siehe Quine Methode oder Schichtenalgorithmus
MinSOP (min. 1)	Minimale Summe v. Primimplikanten	durch Überdeckungstabelle

FPGA: Field Programmable Gate Array
LUT: Look Up Table

6 Beschreibungsformen

6.1 Disjunktive Normalform/Sum of products (DNF/DNF)

Eins-Zeilen als **Implikanten** (UND) schreiben und alle Implikanten mit **ODER** verknüpfen:
$Z=\overline{A}\cdot\overline{B}+\overline{C}\cdot D$

6.2 Konjunktive Normalform/Product of sums (KNF/KNF)

Null-Zeilen negiert als Implikat (ODER) schreiben und alle Implikaten **UND** verknüpfen:
$Z=(\overline{A}+\overline{C})\cdot(\overline{A}+\overline{D})\cdot(\overline{B}+\overline{C})\cdot(\overline{B}+D)$

6.3 Umwandlung in jeweils andere Form

1. Doppeltes Negieren der Funktion: $Z=\overline{\overline{\overline{A}\cdot\overline{B}+\overline{C}\cdot D}}$
2. Umformung "untere" Negation (DeMorgan) : $Z=\overline{\overline{\overline{A}\cdot\overline{B}}\cdot\overline{\overline{C}\cdot D}}=\overline{(A+B)\cdot(C+\overline{D})}$
3. Ausmultiplizieren: $Z=\overline{(A+B)\cdot(C+\overline{D})}=\overline{A\cdot C+A\cdot\overline{D}+B\cdot C+B\cdot\overline{D}}$
4. Umformung "obere" Negation (DeMorgan) :
$Z=\overline{AC}\cdot\overline{A\overline{D}}\cdot\overline{BC}\cdot\overline{B\overline{D}}=(\overline{A}+\overline{C})\cdot(\overline{A}+D)\cdot(\overline{B}+\overline{C})\cdot(\overline{B}+D)$

Analog von KNF (KNF) nach DNF (DNF).

6.4 Shannon Entwicklung

$$f=x_i\cdot f_{x_i}+\overline{x}_i\cdot f_{\overline{x}_i}=(x_i+f_{\overline{x}_i})\cdot(\overline{x}_i+f_{x_i})=(f_{x_i}\oplus f_{\overline{x}_i})\cdot x_i\oplus f_{\overline{x}_i}$$
$$\overline{f}=x_i\cdot\overline{f}_{x_i}+\overline{x}_i\cdot\overline{f}_{\overline{x}_i}$$

7 Logikminimierung

7.1 Nomenklatur

- m_i Minterm: UND-Term in dem alle Variablen vorkommen (aus KDNF)
- M_i Maxterm: ODER-Term in dem alle Variablen vorkommen (aus KKNF)
- c_i Implikant: UND-Term in dem freie Variablen vorkommen können
- C_i Implikat: ODER-Term in dem freie Variablen vorkommen können
- p_i Primimplikant: UND-Term mit maximal freien Variablen
- P_i Primimplikat: ODER-Term mit maximal freien Variablen

7.2 Karnaugh-Diagramm

Zyklische Gray-Codierung:

2-dim	00 01 11 10
3-dim	000 001 011 010 110 111 101 100

$z\backslash^{xy}$	00	01	11	10
0	1	0	0	0
1	X	1	1	0

Gleiche Zellen zusammenfassen: z.B. $\overline{xy}+y\cdot z$

Don't Care Werte ausnutzen!
Achtung: Auf eventuelle Unterdefiniertheit überprüfen (Redundante Zeilen) (Kreiert Don't Cares)
Immer vollständig mit Nullen und Einsen ausfüllen

7.3 Quine Methode

geg.: DNF/DNF oder Wertetabelle von $f(x)$
ges.: alle Primimplikanten p_i (VollSOP)

Spezielles Resoltutionsgesetz: $x\cdot a+\overline{x}\cdot a=a$
Absorptionsgesetz: $a+a\cdot b=a$

1. KDNF/KDNF bestimmen (z.B. $f(x,y,z)=xy=xyz+xy\overline{z}$)
2. Alle Minterme in Tabelle eintragen (Index von m ist (binär)Wert des Minterms)
3. 1-Kubus: Minterme die sich um eine Negation unterscheiden, zu einem Term verschmolzen (Resolutionsgesetz)
4. Der 1-Kubus muss zusammenhängend sein! (d.h. alle 1-Kubus Minterme müssen zusammenhängen)
5. Wenn möglich 2-Kubus bilden.
6. Wenn keine Kubenbildung mehr möglich $\to$ Primimplikanten

Beispiel (Quine Methode):

	0-Kubus	A	1-Kubus	R	A	2-Kubus	A
m_1	$\overline{x}_1\overline{x}_2x_3$	$\checkmark$	$\overline{x}_2x_3$	$m_1\&m_5$	p_1		
m_4	$x_1\overline{x}_2\overline{x}_3$	$\checkmark$	$x_1\overline{x}_2$	$m_4\&m_5$	$\checkmark$	x_1	p_2
m_5	$x_1\overline{x}_2x_3$	$\checkmark$	$x_1\overline{x}_3$	$m_4\&m_6$	$\checkmark$		
m_6	$x_1x_2\overline{x}_3$	$\checkmark$	x_1x_3	$m_5\&m_7$	$\checkmark$		
m_7	$x_1x_2x_3$	$\checkmark$	x_1x_2	$m_6\&m_7$	$\checkmark$		

$\Rightarrow f(x_1,x_2,x_3)=p_1+p_2=\overline{x}_2x_3+x_1$

7.4 Resolventenmethode

Ziel: alle Primimplikanten

Wende folgende Gesetze an:
Absorptionsgesetz: $a+ab=a$
allgemeines Resolutionsgesetz: $x\cdot a+\overline{x}\cdot b=x\cdot a+\overline{x}\cdot b+ab$

Anwendung mit Schichtenalgorithmus

1. schreibe die Funktion f in die 0. Schicht
2. bilde **alle möglichen** Resolventen aus der 0. Schicht und schreibe sie in die nächste Schicht als ODER Verknüpfungen (Resolventen zu f "hinzufügen")
3. überprüfe ob Resolventen aus der 1. Schicht Kuben aus Schicht 0 überdecken(Absorption) und streiche diese Kuben aus Schicht 0
4. Schicht i besteht aus den möglichen Resolventen von Schicht 0 bis $(i-1)$. Abgestrichene Kuben aus vorherigen Schichten brauchen **nicht** mehr beachtet werden.
5. Sobald in der i-ten Schicht +1 steht oder keine weiteren Resolventen gebildet werden können, ist man fertig. $\Rightarrow$ alle nicht ausgestrichenen Terme bilden die VollSOP

$f(x_1,\dots,x_n)$	Schicht
$x\cdot w+\overline{x}\cdot w+x\cdot y\cdot w\cdot\overline{z}+\overline{x}\cdot y\cdot w\cdot\overline{z}+\overline{y}\cdot w\cdot\overline{z}$	0
$+w+y\cdot w\cdot\overline{z}$	1
$+w\cdot\overline{z}$	2
$+w$	3

7.5 Überlagerung Bestimmung der MinSOP

Geg: KDNF/KDNF ($\sum m_i$) und VollSOP ($\sum p_i$) Ges: MinSOP (Minimalform)

Überdeckung:
$$C=\quad(m_0\subseteq p_1)\quad\cdot(m_2\subseteq p_1+m_2\subseteq p_2)\quad\overset{!}{=}1$$
$$C=\quad\tau_1\quad\cdot(\tau_1+\tau_2)\quad=\tau_1+\tau_1\tau_2=\tau_1$$

Alternativ: Mit Überdeckungstabelle bestimmen. Bsp:

	Minterme				
Primterme	m_1	m_2	$\dots$	m_N	$L(p_i)$
p_1	$\checkmark$				$L(p_1)$
p_2	$\checkmark$			$\checkmark$	$L(p_2)$
$\vdots$					$\vdots$
p_K		$\checkmark$			$L(p_K)$

Algorithmus:

1. Suche Spalten mit nur einem Minterm.
2. Streiche andere Spalten des zugehörigen Primterms.
3. Streiche Primterme, dessen Minterme alle gestrichen sind.

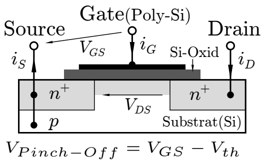
K: Anzahl der Primterme
N: Anzahl der Minterme
$L(p_i)$: Kosten/Länge der Primimplikanten

8 Halbleiter

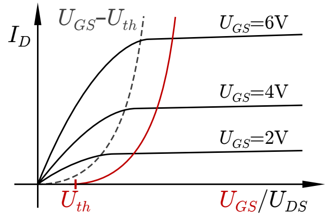
	Isolator	Metall	undotiert	N-Typ	P-Typ
Ladungsträger	Keine	e^-	e^-/e^+	e^-	e^+
Leitfähigkeit	Keine	Sehr hoch	$\propto T$	Hoch	Mittel

9 MOS-FET's

Metal **O**xide **S**emiconductor **F**ield **E**ffekt **T**ransistor



$V_{Pinch-Off} = V_{GS} - V_{th}$



9.1 Bauteilparameter

Verstärkung: $\boxed{\beta = K' \frac{W}{L} \text{ mit } K' = \frac{\mu \varepsilon_{ox} \varepsilon_0}{t_{0x}} \qquad [\beta] = \frac{A}{V^2}}$

Kanalweite	W
Kanallänge	L
Elektronenbeweglichkeit	$\mu \quad \mu_n \approx 250 \cdot 10^{-4} \frac{m^2}{Vs}$, $\mu_p \approx 100 \cdot 10^{-4} \frac{m^2}{Vs}$
rel. Dielektrizität des Gateoxyds	$\varepsilon_{ox} \approx 3,9$
Dielektrizitätskonstante	$\varepsilon_0 = 8.8541878 \cdot 10^{-12} \frac{\mathrm{A\,s}}{\mathrm{V\,m}}$
Gateoxyddicke	t_{ox}
Verstärkung	$\beta = \frac{\mu_n \varepsilon_{ox} \varepsilon_0}{t_{ox}} \cdot \frac{W}{L} = K' \frac{W}{L} = \frac{\mu_n C_G}{L^2}$
Kapazität	$C_G = \varepsilon_{ox} \varepsilon_0 \frac{WL}{t_{ox}}$
Verzögerungszeit	$t_{pHL} \propto \frac{C_L t_{ox} L_p}{W_p \mu_p \varepsilon_{ox}(V_{DD} - \lvert V_{th} \rvert)}$
Verzögerungszeit (2 Signale)	Zeit zwischen $S_1 = 50\%$ und $S_2 = 50\%$
	LH/HL bezieht sich auf Ausgang
Anstiegszeit (Selbes Signal)	t_r Zeit zwischen 10% und 90%
Abfallzeit (Selbes Signal)	t_f Zeit zwischen 90% und 10%

- große Kanalweite $\Rightarrow$ große Drain-Störme
  $\Rightarrow$ schnelle Schaltgeschwindigkeit (da $i_d \propto \beta \propto \frac{W}{L}$)
  Aber: große Fläche.
- nMos schaltet schneller als pMOS

9.2 Drainstrom

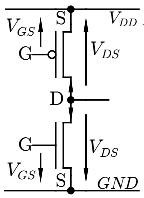
nMos (p-dotiertes Substrat, n-dotierte Drain/Source), schlechter pull up (Pegeldegenerierung)

$$I_d = \begin{cases} 0, & \text{für } U_{gs} - U_{th} \leq 0 \quad \text{(Sperrber.)} \\ \beta[(u_{gs} - U_{th}) \cdot u_{ds} - \frac{1}{2}u_{ds}^2], & \text{für } 0 \leq U_{gs} - U_{th} \geq u_{ds} \quad \text{(linearer Ber.)} \\ \frac{1}{2}\beta \cdot (u_{gs} - U_{th})^2, & \text{für } 0 \leq U_{gs} - U_{th} \leq u_{ds} \quad \text{(Sättigungsber.)} \end{cases}$$

pMos (n-dotiertes Substrat, p-dotierte Drain/Source), schlechter pull down (Pegeldegenerierung)

$$I_d = \begin{cases} 0, & \text{für } U_{gs} - U_{th} \geq 0 \quad \text{(Sperrber.)} \\ -\beta[(u_{gs} - U_{th}) \cdot u_{ds} - \frac{1}{2}u_{ds}^2], & \text{für } 0 \geq U_{gs} - U_{th} \leq u_{ds} \quad \text{(linearer Ber.)} \\ -\frac{1}{2}\beta \cdot (u_{gs} - U_{th})^2, & \text{für } 0 \geq U_{gs} - U_{th} \geq u_{ds} \quad \text{(Sättigungsber.)} \end{cases}$$

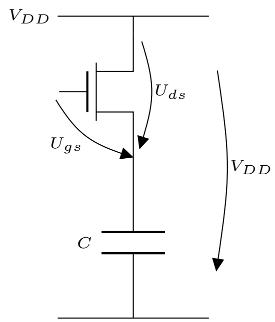
9.3 pMos und nMos



Transistor	Source liegt immer am	V_{GS}, V_{DS}, I_D	Substrat
pMos normally on	höheren Potential	< 0	$+(V_{DD})$
nMos normally off	niedrigeren Potential	> 0	$-(GND)$

Vorsicht: $U_{GS,p} = V_{DD} - U_a$

9.4 Kondensatoraufgaben



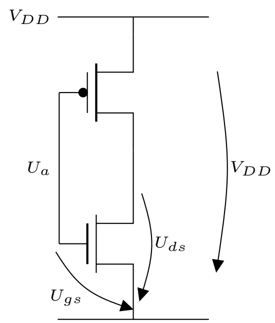
9.4.1 Laden

Kondensator C lädt, solange $I_D > 0$
$\rightarrow C$ lädt, solange $u_{gs} - U_{th} \geq 0$ und $u_{ds} \geq 0$

9.4.2 Entladen

Source und Drain werden vertauscht.
Auf Gatespannung achten.

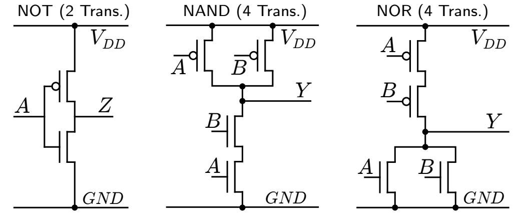
9.5 Gatterschwellspannungsaufgaben



Gatterschwellspannung ist der Punkt, wo sich beide Transistoren in Sättigung befinden.
Dann Ströme mittels Knotengleichung ausrechnen.
$I_{sat,n} = -Isat,p$
Vorsicht: $U_{GS,p} = V_{DD} - U_a$

10 CMOS - Logik

Vorteil: (Fast) nur bei Schaltvorgängen Verlustleistung - wenig statische Verluste
Drei Grundgatter der CMOS-Technologie:



Falls GND und V_{DD} vertauscht würden, dann $NAND \rightarrow AND$ und $NOR \rightarrow OR$
Allerdings schlechte Pegelgenerierung.

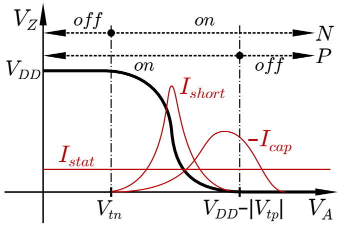
10.1 Gatterdesign

Netzwerk	Pull-Down	Pull-Up
Transistoren	**n**Mos	**p**Mos
AND	Serienschaltung	Parallelschaltung
OR	Parallelschaltung	Serienschaltung

1. Möglichkeit: Direkt; ggf. Inverter vor die Eingänge und Ausgänge schalten.
2. Möglichkeit: Mit boolesche Algebra die Funktion nur mit NAND und NOR darstellen.

10.2 CMOS Verlustleistung

Inverterschaltvorgang $V_A : 0 \rightarrow 1$:



Achtung: Logikpegel sind über die Steigung der $\lvert VTC \rvert \leq 1$ des Inverters definiert.
Zusammensetzung I_{short}:

Transistor	$(0, V_{tn})$	$(V_{tn}, V_{DD}/2)$	Um $V_{DD}/2$	$(V_{DD}/2, V_{DD} - \lvert V_{tp} \rvert)$	$(V_{DD} - V_{tp}, V_{DD})$
n-MOS	Sperrt	Sättigung	Sättigung	Linear	Linear
p-MOS	Linear	Linear	Sättigung	Sättigung	Sperrt

Dynamische Verlustleistung	$P_{dyn} = P_{cap} + P_{short}$
Kapazitive Verluste	$P_{cap} = \alpha_{01} f C_L V_{DD}^2$
Kurzschlussstrom	$P_{short} = \alpha_{01} f \beta_n \tau (V_{DD} - 2V_{tn})^3$
Schalthäufigkeit	$\alpha_{0\rightarrow1} = \frac{\text{Schaltvorgänge(pos. Flanke)}}{\text{\# Betrachtete Takte}}$ (max 0.5)
Schalthäufigkeit (periodisch)	$\alpha = \frac{f_{\text{switch}}}{f_{\text{clk}}}$

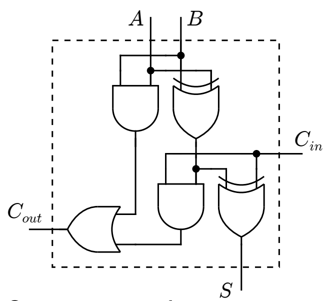
Abhängig von den Signalflanken, mit Schaltfunktionen verknüpft
$\approx V_{DD}1/ \propto$ Schaltzeit: $\frac{V_{DD2}}{V_{DD1}} = \frac{t_{D1}}{t_{D2}}$ (bei Schaltnetzen t_{log})

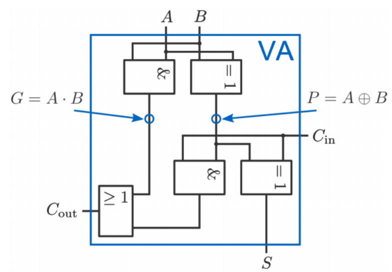
Verzögerungszeit $t_{pd} \propto \frac{C_L t_{ox} L_p}{W_p \mu_p \varepsilon (V_{DD} - V_{th})}$

t_{pd} ist Zeit zwischen crossover 50% von Eingang zu crossover 50% am Ausgang.
Steigend mit: Kapazitiver Last, Oxiddicke, Kanallänge, Schwellspannung
Sinkend mit: Kanalweite, Ladungsträger Beweglichkeit, Oxyd Dielektrizität, Versorgungsspannung

Statische Verlustleistung P_{stat}: Sub-Schwellströme, Leckströme, Gate-Ströme Abhängigkeit:
$V_{DD} \uparrow$: $P_{stat} \uparrow$ $\quad V_{th} \uparrow$: $P_{stat} \downarrow$ (aber nicht proportional)

11 Volladdierer (VA)/Ripple-C(u)arry-Adder





Generate $g_n = a_n \cdot b_n$
Propagate $p_n = a_n \oplus b_n$
Summenbit $S_n = c_n \oplus p_n = a_n \oplus b_n \oplus c_n$
$S_n = \underbrace{a_n \overline{b_n} \overline{c_n} + \overline{a_n} b_n \overline{c_n} + \overline{a_n} \overline{b_n} c_n}_{\text{genau ein Eingang high}} + \underbrace{a_n b_n c_n}_{\text{alle Eingänge high}}$ (Ungerade Anzahl von Eingängen 1)
Carry-out $c_{n+1} = c_n \cdot p_n + g_n$

$$c_{n+1} = \underbrace{a_n b_n \overline{c_n} + a_n \overline{b_n} c_n + \overline{a_n} b_n c_n}_{\text{zwei Eingänge 1}} + \underbrace{a_n b_n c_n}_{\text{drei Eingänge 1}} \quad \text{(Mindesten zwei Eingänge 1)}$$

Laufzeiten

$$t_{sn} = \begin{cases} t_{cn} + t_{xor} & t_{cn} > t_{xor} \\ 2t_{xor} & sonst \end{cases}$$

$$t_{cn+1} = \begin{cases} t_{and} + t_{or} & a_n = b_n = 1 & (g_n = 1) \\ t_{xor} + t_{and} + t_{or} & a_n = b_n = 0 & (p_n = 0, g_n = 0) \\ t_{cn} + t_{and} + t_{or} & a_n \neq b_n & (p_n = 1) \end{cases}$$

12 Sequentielle Logik

Logik mit Gedächtnis (Speicher).

12.1 Begriffe/Bedingungen

t_{Setup}	Stabilitätszeit vor der aktiven Taktflanke
t_{hold}	Stabilitätszeit nach der aktiven Taktflanke
t_{c2q}	Eingang wird spätestens nach t_{c2q} am Ausgang verfügbar
Min. Taktperiode	$t_{clk} \geq t_{1,c2q} + t_{logic,max} + t_{2,setup}$
Max. Taktfrequenz	$f_{max} = \left\lfloor \frac{1}{t_{clk}} \right\rfloor$ (Nicht aufrunden)
Holdzeitbedingung	$t_{hold} \leq t_{c2q} + t_{logic,min}$ → Dummy Gatter einbauen
Durchsatz	$\frac{\text{1Sample}}{t_{clk,pipe}} = f$
Latenz	$t_{clk} \cdot$ #Pipelinestufen (das zwischen den FFs)

12.2 Pipelining

Nur bei synchronen(taktgesteuerten) Schaltungen möglich!

- Aufteilen langer kombinatorischer Pfade durch Einfügen zusätzlicher Registerstufen → Möglichst Halbierung des längsten Pfades
- Zeitverhalten beachten (evtl. Dummy-Gatter einfügen)
- Durchsatz erhöht sich entsprechend der Steigerung der Taktfrequenz
- Gesamtlatenz wird eher größer
- Taktfrequenz erhöht sich

12.3 Parallel Processing

Durchsatz $= \frac{\#\text{Modul}}{t_{clk,Modul}} = f$ Latenz $= t_{clk}$

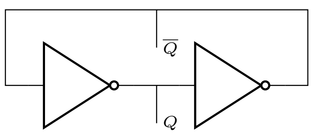
- Paralleles, gleichzeitiges Verwenden mehrere identischer Schaltnetze
- Zusätzliche Kontrolllogik nötig (Multiplexer)
- Taktfrequenz und Latenz bleiben konstant
- Durchsatz steigt mit der Zahl der Verarbeitungseinheiten ABER: deutlich höherer Ressourcenverbrauch

13 Speicherelemente

Flüchtig Speicherinhalt gehen verloren, wenn Versorgungsspannung V_{DD} wegfällt - Bsp: *RAM
Nicht Flüchtig Speicherinhalt bleibt auch ohne V_{DD} erhalten - Bsp: Flash
Asynchron Daten werden sofort geschrieben/gelesen.
Synchron Daten werden erst mit $clk_{0\to 1}$ geschrieben.
Dynamisch Ohne Refreshzyklen gehen auch bei angelegter V_{DD} Daten verloren - Bsp: DRAM
Statisch Behält den Zustand bei solange V_{DD} anliegt (keine Refreshzyklen nötig) - Bsp: SRAM
Bandbreite: Bitanzahl, die gleichzeitig gelesen/geschrieben werden kann.
Latenz: Zeitverzögerung zwischen Anforderung und Ausgabe von Daten.
Zykluszeit: Minimale Zeitdifferenz zweier Schreib/Lesezugriffe.

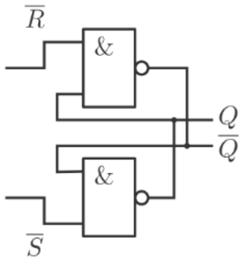
$\boxed{\text{Speicherkapazität} = \text{Wortbreite} \cdot 2^{\text{Adressbreite}}}$

13.1 Speicherzelle/Register



Ring aus zwei Invertern.
Logikpegel kann nur mit öffnen des Inverter-Rings gesetzt werden.

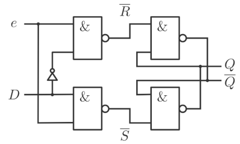
13.2 Latch



Set-Reset Latch:
Zwei gegenseitig rückgekoppelte NAND-Gatter.

Active Low Logik:
$\overline{S} = 0 \Rightarrow Q = 1$, $\overline{R} = 0 \Rightarrow Q = 0$

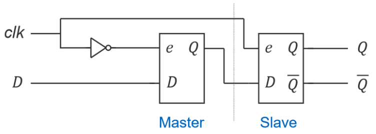
$\overline{R}$	$\overline{S}$	Q
1	1	Q
0	1	0
1	0	1
0	0	$Q = \overline{Q}$



Enable-Latch: ändert Speicherzustand auf D nur wenn $e = 1$.
Level-Controlled ⇔ Latch.

e	Q
0	Q
1	D

13.3 Flip-Flop



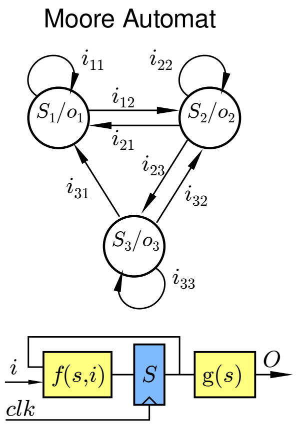
clk	Q	$\overline{Q}$
$0 \to 1$	D	$\overline{D}$
sonst	Q	$\overline{Q}$

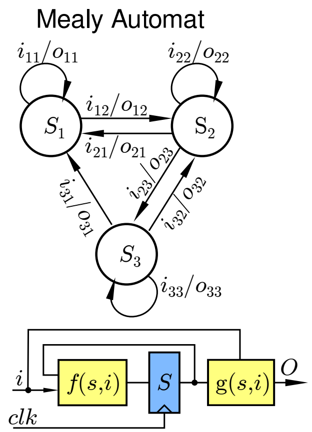
Besteht aus zwei enable-Latches
Flip-Flop: Ändert Zustand bei steigender/(fallender) Taktflanke.

14 Automaten

DFA 6-Tupel $\{I, O, S, R, f, g\}$

I	Eingabealphabet
O	Ausgabealphabet
S	Menge von Zuständen
$R \subseteq S$	Menge der Anfangszustände
$f : S \times I \to S$	Übergangsrelation
g	Ausgaberelation





Zustandsnummerierung immer einfügen.

Moore	Mealy
Ouput hängt nur vom Zustand ab	Output hängt von Zustand und Eingabe
Kein direkter kombinatorischer Pfad Eingang⇒Ausgang	Generell weniger Zustände als Moore
$s' = f(s,i)$, $o = g(s)$	$s' = f(s,i)$, $o = g(s,i)$
$g : S \to O$	$g : S \times I \to O$

14.1 Wahrheitstabelle einer FSM

i	$S = S_0 \ldots S_n$	o	$S' = S'_1 \ldots S'_n$
0	$0\ldots0$	$o_{0,0\ldots0}$	$S'_{0,0\ldots0}$
⋮	⋮	⋮	⋮
1	$1\ldots1$	$o_{1,1\ldots1}$	$S'_{1,1\ldots1}$

Moore: o ist $f(S)$, nächster Zustand $S' = f(i, S)$
Mealy: o ist $f(i, S)$, nächster Zustand $S' = f(i, S)$