

## 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 Tran-
- Herstellungskosten (Fixkosten, Variable Kosten, Technologiefaktor), Entwicklerproduktivität, Verlustleistungsdichte

### 2 Einheiten

Potenz	Vorsatz	Potenz Vorsatz	Hz	$s^{-1}$
10 <sup>12</sup>	Т	10 <sup>-1</sup> d	N	$kgms^{-2}$
$10^{9}$	G	$10^{-2}$ c	J	Nm = VAs
$10^{6}$	М	10 <sup>-3</sup> m	W	$VA = Js^{-1}$
$10^{3}$	k	$10^{-6}$ $\mu$	C	As
$10^{2}$	h	10 <sup>-9</sup> n	V	$JC^{-1}$
$10^{1}$	da	10 <sup>-12</sup> p	F	$CV^{-1}$
	ļ.	$10^{-15}$ f	Ω	$VA^{-1}$ $VsA^{-1}$
		- 1	H	$VsA^{-1}$

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

# 3 Polyadische Zahlensysteme

$$Z = \sum_{i=-n}^{p-1} r^i \cdot d_i = d_{p-1}...d_1d_0.d_{-1}...d_n$$
 
$$Z: \mathsf{Zahl}, \quad r: \mathsf{Basis}, \quad d_i: \mathsf{Ziffer}, \quad p: \#\mathsf{Ziffern} \text{ vorne} \quad n: \#\mathsf{Nachkommastellen}$$

#### Binäres Zahlensystem:

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

### 3.1 Umrechnung

	$Z \ge 1$	Z < 1						
$r \rightarrow 10$	$Z_{10} = \sum_{i} r^{i} \cdot d_{i}$ $101_{2} \rightarrow 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1$	$Z_{10} = \sum_{i} r^{-i} \cdot d_{-i}$ 0.11 <sub>2</sub> \to 1 \cdot 0.5 + 1 \cdot 0.25						
	$d_i=Z_{10}\%r^i~(d_i=Z_{10}~{ m mod}~r^i)$ $58/8=7~{ m Rest}~2(LSB)$ $7/8=0~{ m Rest}~7(MSB)$ (Ende wenn 0 erreicht) Auf Ende achten $1r3\%5  ightarrow 0r1$	$0.4 \cdot 2 = 0.8$ Übertrag $0(MSB)$ $0.8 \cdot 2 = 1.6$ Übertrag $1$ (Wiederholen bis 1 oder Periodizität)						

#### Wertebereich: $-2^{n-1} \le Z \le 2^{n-1} - 1$ 3.2 Zweierkomplement

 $Z \rightarrow -Z$  (Umkehrung gleich) 1. Invertieren aller Bits

Bsp: Wandle 2 in -2 um  $0010 \Rightarrow 1101$ 2. Addition von 1 1101 + 1 = 11103. Ignoriere Überträge beim MSB  $\Rightarrow$   $-2_{10} = 1110_2$ 

# 3.3 Gleitkommadarstellung nach IEEE 754

Bitverteilung(single/double):

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

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

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

	$\begin{array}{llllllllllllllllllllllllllllllllllll$
Wert $Z  o$ IEEE (Binärdarstellung) $s = 0$ (positiv), $s = 1$ (negativ) $Z  o Z_2$ (beim Komma teilen) $Z_2$ n-mal shiften $ o 1.xxx\dots$ Exponent $e = n + 127  o e_2$ Mantisse $f_2 = xxx\dots$	$\begin{aligned} & \text{Bsp: } Z = 11.25 \\ & s = 0 \\ & Z = 1011.01_2 \\ & Z = 1.01101_2 \cdot 2^3 \\ & e = 3 + 127 = 130 = 10000010_2 \\ & f = 01101000 \ldots_2 \end{aligned}$
$\begin{aligned} & \text{Wert } Z \to \text{IEEE (Formel)} \\ & s = 0 (\text{positiv}),  s = 1 (\text{negativ}) \\ & E = \lfloor \log_2  Z  \rfloor \\ & e = E + 127 \to e_2 \\ & f = \left(\frac{ Z }{2^E} - 1\right) \cdot 2^{23} \to f_2 \end{aligned}$	$\begin{array}{l} Bsp:\ Z=11.25\\ s=0\\ E=\lfloor\log_2 11.25 \rfloor=\lfloor3,49\ldots\rfloor=3\\ e=3+127=130=10000010_2\\ f=\left(\frac{ 11.25 }{2^3}-1\right)\cdot 2^{23}=3407872=\\ 01101000\ldots_2 \end{array}$

# 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) → Effizient

### Schema

- MSB = 0 → 8 Bit (restliche Bit nach ASCII)
- MSB =  $1 \rightarrow 16$ , 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	<b>0o</b> 00	<b>0</b> x0
01	0001	<b>0o</b> 01	0×1
02	0010	<b>0o</b> 02	<b>0</b> x2
03	0011	<b>0o</b> 03	<b>0</b> x3
04	0100	<b>0o</b> 04	0×4
05	0101	<b>0o</b> 05	<b>0</b> x5
06	0110	<b>0o</b> 06	<b>0</b> x6
07	0111	<b>0o</b> 07	0x7
80	1000	<b>0o</b> 10	<b>0x</b> 8
09	1001	0o11	<b>0</b> x9
10	1010	<b>0o</b> 12	0xA
11	1011	<b>0o</b> 13	0xB
12	1100	<b>0</b> o14	0xC
13	1101	<b>0o</b> 15	<b>0</b> xD
14	1110	<b>0o</b> 16	0xE
15	1111	<b>0</b> o17	0xF

# 5 Boolsche Algebra

## 5.1 Boolesche Operatoren (Wahrheitstabelle WT)

			A D out	Aout	A B — out	Ao—out	A Do—out	A Do—out
			n	n P	n	Р	n P	n → → → → →
			A — & B — Y	A 21 -Y	A =1 =1 -Y	А — & D—Y	A = ≥1 D=Y	A ==1 D-Y
	×	у	AND	OR	XOR	NAND	NOR	EQV
			$x \cdot y$	x + y	$x \oplus y$	$\overline{x \cdot y}$	$\overline{x+y}$	$\overline{x \oplus y}$
	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	Mengenalgebra
	$(0,1;\cdot,+,\overline{x})$	$(P(G); \cap, \cup, \overline{A}; G, \emptyset)$
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	$x \cdot \overline{x} = 0$	$A \cap \overline{A} = \emptyset$
	$x + \overline{x} = 1$	$A \cup \overline{A} = G$
	$\overline{x} = x$	$\overline{\overline{A}} = A$
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

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

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

### Kofaktor bezüglich

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

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

- tautologisch  $\Leftrightarrow f(x) = 1 \quad \forall x \in \{0, 1\}^n$
- kontradiktorisch  $\Leftrightarrow f(\mathbf{x}) = 0 \quad \forall \mathbf{x} \in \{0, 1\}^n$
- 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)  $f = \overline{x}_1 \overline{x}_2 a + \overline{x}_1 x_2 b + x_1 \overline{x}_2 c + x_1 x_2 d$  (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 $\overline{x_i}$	$i \in I_0 = \{1,, n\}$
Minterme,0-Kuben	$MOC  ightarrow m_j = \prod_{i \in I_0} l_i$	$ M0C  = 2^n$
d-Kuben	$MC  i c_j = \prod_{i \in I_j \subseteq I_0} l_i$	$ MC  = 3^n$
Distanz	$\delta(c_i, c_j) =  \{l \mid l \in c_i \land \overline{l} \in c_j\} $	$\delta_{ij} = \delta(c_i, c_j)$
Implikanten	$MI = \{c \in MC \mid c \subseteq f\}$	
Primimplikanten	$MPI = \{ p \in MI \mid p \not\subset c \ \forall c \in MI \}$	$MPI \subseteq MI \subseteq MC$

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 ieweils andere Form

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

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

# 7 Logikminimierung

### 7.1 Nomenklatur

- $\bullet$   $m_i$  Minterm: UND-Term in dem alle Variablen vorkommen (aus KDNF)
- Mi Maxterm: ODER-Term in dem alle Variablen vorkommen (aus KKNF)
- c<sub>i</sub> Implikant: UND-Term in dem freie Variablen vorkommen können
- C<sub>i</sub> Implikat: ODER-Term in dem freie Variablen vorkommen können
- pi Primimplikant: UND-Term mit maximal freien Variablen
- P<sub>i</sub> Primimplikat: ODER-Term mit maximal freien Variablen

### 7.2 Karnaugh-Diagramm

00 01 11 10 Zyklische Gray-Codierung: 000 001 011 010 110 111 101 100 Gleiche Zellen zusammenfassen: z.B.  $\overline{xy} + y \cdot z$ X 1 1

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  $p_i$  (VolISOP)

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 zusam
- 5. Wenn möglich 2-Kubus bilden.
- 6. Wenn keine Kubenbildung mehr möglich → Primimplikanten

### Beispiel (Quine Methode):

	0-Kubus	A	1-Kubus	R	A	2-Kubus	A	ĺ
$m_1$	$\overline{x}_1\overline{x}_2x_3$	$  \sqrt{ }$	$\overline{x}_2x_3$	$m_1 \& m_5$	$p_1$			
$m_4$	$x_1\overline{x}_2\overline{x}_3$	√	$x_1\overline{x}_2$	$m_4 \& m_5$	√	$x_1$	$p_2$	
$m_5$	$x_1\overline{x}_2x_3$	√	$x_1\overline{x}_3$	$m_4 \& m_6$	√			ĺ
$m_6$	$x_1x_2\overline{x}_3$	√	$x_{1}x_{3}$	$m_5 \& m_7$	√			ĺ
$m_7$	$x_1x_2x_3$	$ $	$x_{1}x_{2}$	$m_6 \& m_7$	√			ı
$\Rightarrow f(x_1, x_2, x_3) = p_1 + p_2 = \overline{x}_2 x_3 + x_1$								

# 7.4 Resolventenmethode

Ziel: alle Primimplikanten

Wende folgende Gesetze an:

Absorptionsgesetz: a + ab = aallgemeines 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. ⇒ alle nicht ausgestrichenen Terme bilden die VollSOP

$f(x_1,\ldots,x_n)$	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)

Alternativ: Mit Überdeckungstabelle bestimmen. Bsp:

		Minterme			
Primterme	$m_1$	$m_2$		$m_N$	$L(p_i)$
$p_1$	√				$L(p_1)$
$p_2$	√			$\checkmark$	$L(p_2)$
:					:
$p_K$		√			$L(p_K)$

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

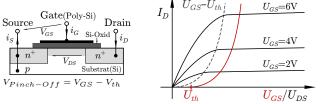
 $L(p_i)$ : Kosten/Länge der Primimplikanten

# 8 Halbleiter

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

# 9 MOS-FET's

Metal Oxide Semiconductor Field Effekt Transistor



# 9.1 Bauteilparameter

 $\beta = K' \frac{W}{L} \text{ mit } K' = \frac{\mu \varepsilon_{OX} \varepsilon_0}{t_{OX}}$ Verstärkung: Kanalweite Kanallänge  $\mu \quad \mu_n \approx 250 \cdot 10^{-4} \frac{m^2}{V_s}, \, \mu_p \approx 100 \cdot 10^{-4} \frac{m^2}{V_s}$ Elektronenbeweglichkeit rel. Dielektrizität des Gateoxyds  $\varepsilon_0 = 8.8541878 \cdot 10^{-12} \frac{\text{A s}}{\text{V}}$ Dielektrizitätskonstante Gateoxyddicke Verstärkung Kapazität Verzögerungszeit  $t_{pHL} \propto \frac{1}{W_p \mu_p \varepsilon_{ox}(V_{DD} - |V_{th}|)}$ Verzögerungszeit  $t_{vHL}$  Propagation delay von 90% auf 10% Verzögerungszeit  $t_{pLH}$  Propagation delay von 10% auf 90%

- große Kanalweite ⇒ 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)

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

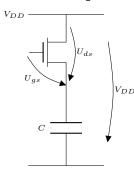
pMos (n-dotiertes Substrat, p-dotierte Drain/Source), schlechter pull down (Pegeldegenerierung)

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

### 9.3 pMos und nMos

$V$ S $V_{DD}$ =	Transistor	Source liegt immer am	$V_{GS}, V_{DS}, I_D$	Substrat	
$V_{GS}$ $V_{DS}$ $V_{DS}$	pMos normally on	höheren Potential	< 0	$+(V_{DD})$	
$V_{DS}$	nMos normally off	niedrigeren Potential	> 0	-(GND)	
Vorsicht: $U_{GS,p} = V_{DD} - U_a$					

### 9.4 Kondensatoraufgaben



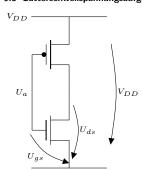
### 9.4.1 Laden

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

### 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.

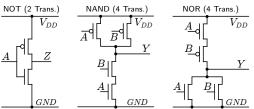
Dann Ströme mittels Knotengleichung ausrechnen.

 $V_{DD}$   $I_{sat,n} = -Isat, p$ 

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.

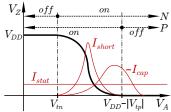
### 10.1 Gatterdesign

Netzwerk	Pull-Down	Pull-U <b>p</b>
Transistoren	nMos	pMos
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  $V_A:0\to 1$ :



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

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

 $P_{dyn} = P_{cap} + P_{short}$ Dynamische Verlustleistung  $P_{cap} = \alpha_{01} f C_L V_{DD}^2$ Kapazitive Verluste Kurzschlussstrom  $P_{short} = \alpha_{01} f \beta_n \tau (V_{DD} - 2V_{tn})^3$ 

 $\alpha_{0 
ightarrow 1} = rac{ ext{Schaltvorgänge(pos. Flanke)}}{\# ext{Betrachtete Takte}} \; ( ext{max 0.5})$ Schalthäufigkeit  $\alpha = \frac{f_{\text{switch}}}{2}$ Schalthäufigkeit (periodisch)

Abhängig von den Signalflanken, mit Schaltfunktionen verknüpft

 $\approx V_{DD}1/\propto \text{Schaltzeit:} \ \frac{V_{DD}2}{V_{DD}1} = \frac{t_{D1}}{t_{D2}} \ (\text{bei Schaltnetzen} \ t_{log})$   $\text{Verzögerungszeit} \ t_{pd} \propto \frac{C_L t_{ox} L_p}{W_p \mu_p \varepsilon (V_{DD} - V_{th})}$ 

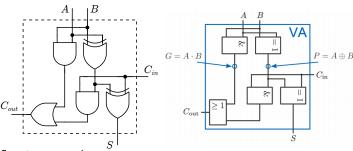
 $t_{nd}$  ist Zeit zwischen crossover 50% von Eingang zu crossover 50% am Ausgang.

Steigend mit: Kapazitiver Last, Oxiddicke, Kanallänge, Schwellspannung

Sinkend mit: Kanalweite, Ladungsträger Beweglichkeit, Oxyd Dielektrizität, Versorgungsspannung

 $\textbf{Statische Verlustleistung} \ \ P_{stat} \text{: Sub-Schwellstr\"{o}me, Leckstr\"{o}me, Gate-Str\"{o}me \ Abh\"{a}ngigkeit:}$  $V_{DD} \uparrow: P_{stat} \uparrow V_{th} \uparrow: P_{stat} \downarrow \text{ (aber nicht proportional)}$ 

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



Generate  $g_n = a_n \cdot b_n$ Propagate  $p_n = a_n \oplus b_n$ 

Summerbit  $S_n = c_n \oplus p_n = a_n \oplus b_n \oplus c_n$ 

 $S_n = a_n \overline{b_n} \overline{c_n} + \overline{a_n} b_n \overline{c_n} + \overline{a_n} \overline{b_n} c_n + a_n b_n c_n$  (Ungerade Anzahl von Eingängen 1)

genau ein Eingang high alle Eingänge high

Carry-out  $c_{n+1} = c_n \cdot p_n + g_n$ 

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

Laufzeiten

$$\begin{aligned} t_{sn} &= \begin{cases} t_{cn} + t_{xor} & t_{cn} > t_{xor} \\ 2t_{xor} & sonst \end{cases} \\ t_{cn+1} &= \begin{cases} t_{and} + t_{or} & a_n = b_n = 1 \\ t_{xor} + t_{and} + t_{or} & a_n = b_n = 0 \\ t_{cn} + t_{and} + t_{or} & a_n \neq b_n \end{cases} \qquad (p_n = 1)$$

# 12 Sequentielle Logik

Logik mit Gedächtnis (Speicher)

# 12.1 Begriffe/Bedingungen

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

### 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

$$\label{eq:Durchsatz} \mathsf{Durchsatz} = \frac{\#\mathsf{Modul}}{t_{clk,Modul}} = f \qquad \qquad \mathsf{Latenz} = t_{clk}$$

- 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  $clk_{0\rightarrow 1}$  geschrieben.

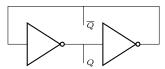
Dynamisch Ohne Refreshzyklen gehen auch bei angelegter  $V_{DD}$  Daten verloren - Bsp: DRAM Statisch Behält den Zustand bei solange  $V_{DD}$  anliegt (keine Refreshzyklen nötig) - Bsp: SRAM Bandbreite: Bitanzahl, die gleichzeitig gelesen/geschrieben werden kann.

Latenz: Zeitverzögerung zwischen Anforderung und Ausgabe von Daten.

Zykluszeit: Minimale Zeitdifferenz zweier Schreib/Lesezugriffe.

$${\sf Speicherkapazit\"{a}t} = {\sf Wortbreite} \cdot 2^{{\sf Adressbreite}}$$

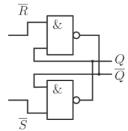
### 13.1 Speicherzelle/Register



Ring aus zwei Invertern.

Logikpegel kann nur mit öffnen des Inverter-Rings gesetzt werden.

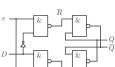
### 13.2 Latch



### Set-Reset Latch:

Zwei gegenseitig rückgekoppelte NAND-Gatter.

 $Q = \overline{Q}$ 



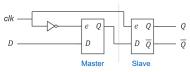
**Enable-Latch:** ändert Speicherzustand auf D nur wenn e=1.

Level-Controlled ⇔ Latch.



0 0

### 13.3 Flip-Flop



clk	Q	$\overline{Q}$
$0 \rightarrow 1$	D	$\overline{D}$
sonst	0	$\overline{o}$

Besteht aus zwei enable-Latches

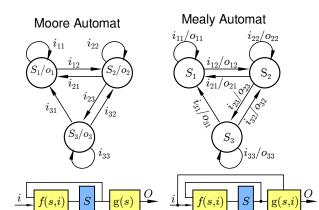
Flip-Flop: Ändert Zustand bei steigender/(fallender)

Taktflanke.

# 14 Automaten

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

$$\begin{array}{c|c} I & \mathsf{Eingabealphabet} \\ O & \mathsf{Ausgabealphabet} \\ S & \mathsf{Menge} \ \mathsf{von} \ \mathsf{Zust"anden} \\ R \subseteq S & \mathsf{Menge} \ \mathsf{der} \ \mathsf{Anfangszust"ande} \\ f: S \times I \to S & \mathsf{Übergangsrelation} \\ g & \mathsf{Ausgaberelation} \end{array}$$



Zustandsnummerierung immer einfügen.

Moore	Mealy
Ouput hängt nur vom Zustand ab	Output hängt von Zustand und Eingabe ab
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 o O	$g: S \times I \to O$

### 14.1 Wahrheitstabelle einer FSM

i	$S = S_0S_n$	o	$S' = S_1' S_n'$
0	00	00,00	$S'_{0,00}$
÷	:	:	:
1	11	$o_{1,11}$	$S'_{1,1,\ldots 1}$

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