



# 64-040 Modul IP7: Rechnerstrukturen

[http://tams.informatik.uni-hamburg.de/  
lectures/2011ws/vorlesung/rs](http://tams.informatik.uni-hamburg.de/lectures/2011ws/vorlesung/rs)

## Kapitel 16

Andreas Mäder



Universität Hamburg  
Fakultät für Mathematik, Informatik und Naturwissenschaften  
Fachbereich Informatik  
Technische Aspekte Multimodaler Systeme

Wintersemester 2011/2012



# Kapitel 16

## VLSI-Entwurf und -Technologie

### Halbleitertechnologie

- Halbleiter

- Herstellung von Halbleitermaterial

- Planarprozess

### CMOS-Schaltungen

- Logische Gatter

- Komplexgatter

- Transmission-Gate

- Tristate-Treiber

- Latch und Flipflop

- SRAM

### CMOS-Herstellungsprozess

### Programmierbare Logikbausteine



# Kapitel 16 (cont.)

Entwurf Integrierter Schaltungen  
Literatur



# Erinnerung

Das **Konzept** des Digitalrechners (von-Neumann Prinzip) ist völlig unabhängig von der Technologie:

- ▶ mechanische Rechenmaschinen
- ▶ pneumatische oder hydraulische Maschinen
- ▶ Relais, Vakuumröhren, diskrete Transistoren
- ▶ molekulare Schaltungen
- ▶ usw.

Aber:

- ▶ nur hochintegrierte Halbleiterschaltungen („VLSI“) erlauben die billige Massenfertigung mit Milliarden von Komponenten
- ▶ **Halbleiter** und **Planarprozess** sind essentielle Basistechnologien



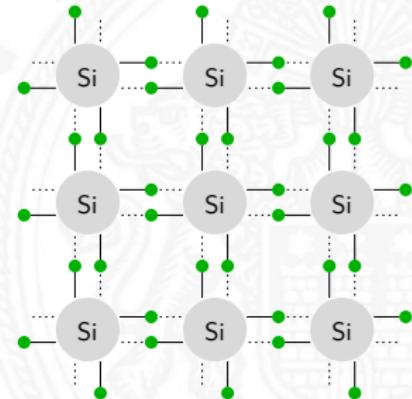
# Halbleiter

Halbleiter stehen zwischen *Leitern* (z.B.: Metalle) und *Isolatoren*.

- ▶ bei Raumtemperatur quasi nicht-leitend
- ▶ Leitfähigkeit steigt mit der Temperatur  $\Rightarrow$  Heißleiter
- ▶ physikalische Erklärung über Bändermodell  
siehe <http://de.wikipedia.org/wiki/Halbleiter>

Kristallstruktur aus 4-wertigen Atomen

- ▶ elementare Halbleiter: Ge, Si
- ▶ Verbindungshalbleiter: GaAs, InSb





## Leitung im undotierten Kristall

- ▶ Paarentstehung: Elektronen lösen sich aus Gitterverband  
Paar aus Elektron und „Loch“ entsteht.
- ▶ Rekombination: Elektronen und Löcher verbinden sich  
quasistatischer Prozess
- ▶ Eigenleitungsichte  $n_i$ : temperatur- und materialabhängig

Si :  $1,2 \cdot 10^{10} \text{ cm}^{-3}$

Ge :  $2,5 \cdot 10^{13} \text{ cm}^{-3}$

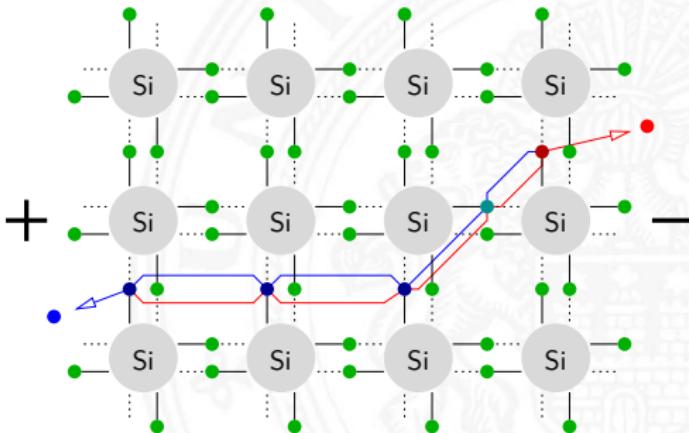
GaAs :  $1,8 \cdot 10^6 \text{ cm}^{-3}$

bei  $300^\circ K \approx 20^\circ C$

Atomdichte

Si :  $5 \cdot 10^{22} \text{ cm}^{-3}$

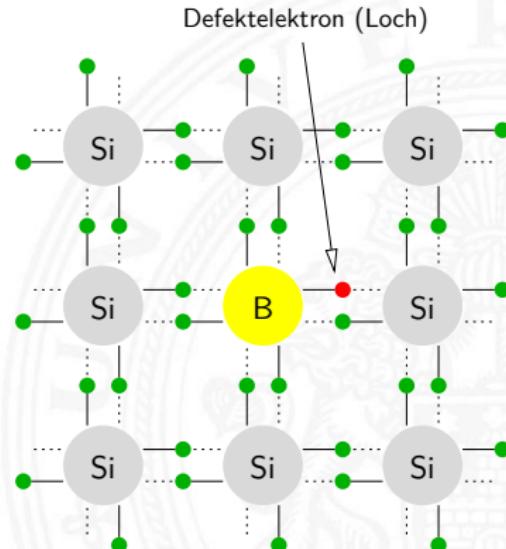
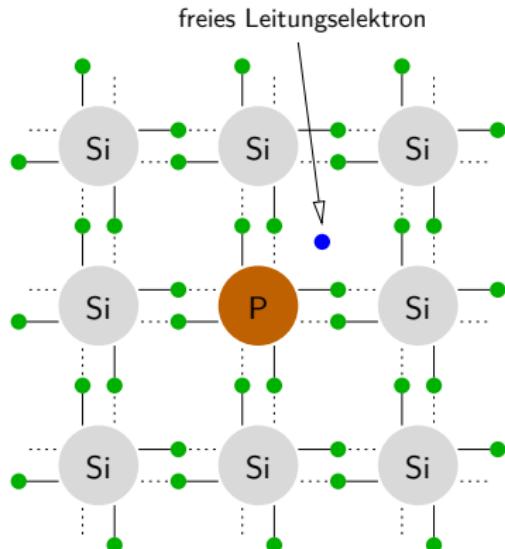
- ▶ es gilt:  $n_i^2 = n_n \cdot n_p$





## Dotierung mit Fremdatomen

Ein kleiner Teil der vierwertigen Atome wird durch fünf- oder dreiwertige Atome ersetzt.





## Dotierung mit Fremdatomen (cont.)

- ▶ Donatoren, Elektronenspender: Phosphor, Arsen, Antimon
- ▶ Akzeptoren: Bor, Aluminium, Gallium, Indium
- ▶ Dotierungsdichten

	Stärke	Fremdatome [ $cm^{-3}$ ]
schwach	$n^-$ , $p^-$	$10^{15} \dots 10^{16}$
mittel	$n$ , $p$	$10^{16} \dots 10^{19}$
stark	$n^+$ , $p^+$	$10^{19} \dots$

- ▶ Beweglichkeit  $\mu$ : materialspezifische Größe

$T = 300^\circ K$		Si	Ge	GaAs	[ $cm^2/(Vs)$ ]
Elektronen	$\mu_n$	1500	3900	8500	
Löcher	$\mu_p$	450	1500	400	

- ▶ Leitfähigkeit: ergibt sich aus Material, Beweglichkeit und Ladungsträgerdichte( $n$ )

$$K = e(n_n\mu_n + n_p\mu_p)$$



## Dotierung mit Fremdatomen (cont.)

- ▶ selbst bei hoher Dotierung ist die Leitfähigkeit um Größenordnungen geringer als bei Metallen

**Si** 1 freier Ladungsträger pro 500 Atome ( $10^{19} / 5 \cdot 10^{22}$ )

**Met** mindestens 1 Ladungsträger pro Atom

- ▶ Majoritätsträger: Ladungsträger in Überzahl (i.d.R. Dotierung)  
Minoritätsträger: Ladungsträger in Unterzahl

$$n_i^2 = n_n \cdot n_p$$



# Herstellung von Halbleitermaterial

Übersicht in: <http://de.wikipedia.org/wiki/Silicium>





## Rohsilizium

- ▶ Siliziumoxid ( $SiO_2$ ): Sand, Kies. . .  
ca. 20 % der Erdkruste
- ▶ Herstellung im Lichtbogenofen: Siliziumoxid + Koks  
 $SiO_2 + 2C \rightarrow Si + 2CO$
- ▶ amorphe Struktur, polykristallin
- ▶ noch ca. 2 % Verunreinigungen (Fe, Al. . .)

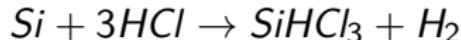




# Solarsilizium

Ziel: Fremdatome aus dem Silizium entfernen

## 1. Chemische Bindung des Siliziums



Reaktion mit Salzsäure erzeugt

$SiHCl_3$  Trichlorsilan

$SiCl_4$  Siliziumchlorid (10%)

$SiH_2Cl_4$  div. andere Chlorsilane/Silane

$FeCl_2, AlCl_3$  div. Metallchloride

## 2. Verschiedene Kondensations- und Destillationschritte trennen

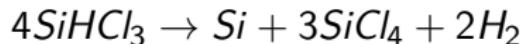
Fremdverbindungen ab, hochreines Trichlorsilan entsteht

< 1ppm Verunreinigungen

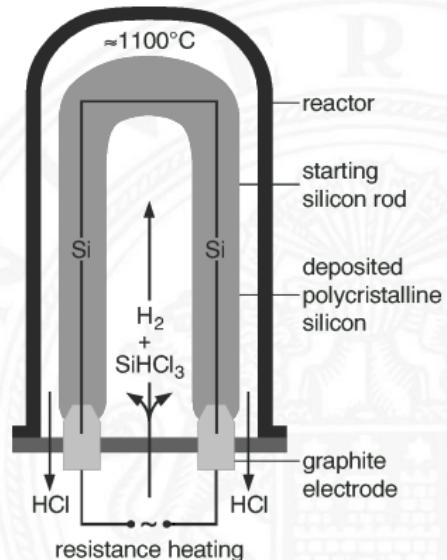


## Solarsilizium (cont.)

3. CVD (Chemical Vapour Deposition) zur Abscheidung des Trichlorsilans zu elementarem Silizium



⇒ polykristallines Silizium  
 $< 0,1\text{ppm}$  Verunreinigungen





# Siliziumeinkristall

## Weitere Ziele

- ▶ Einkristalline Struktur erzeugen
- ▶ Reinheit für Halbleiterherstellung erhöhen  
 $<$ ,  $\ll$  1ppb
- ▶ ggf. Dotierung durch Fremdatome einbringen

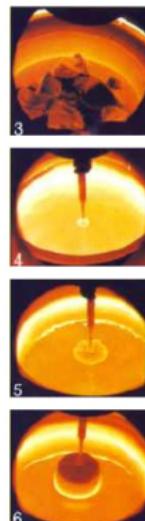
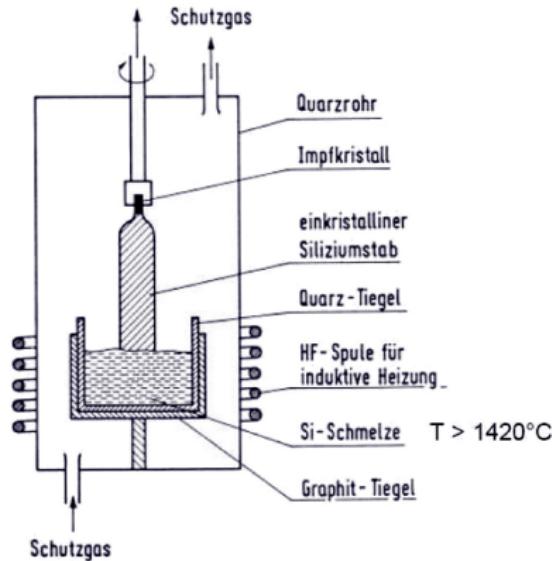
Es gibt dazu mehrere technische Verfahren, bei denen das polykristalline Silizium geschmolzen wird und sich monokristallin an einen Impfkristall anlagert.





# Siliziumeinkristall (cont.)

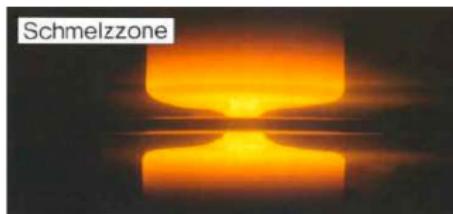
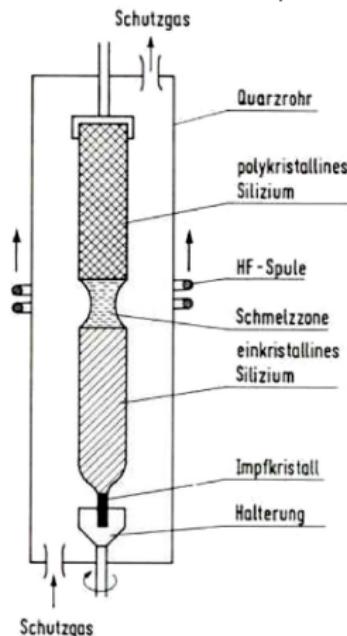
## Czochralski-Verfahren (Tiegelziehverfahren)





## Siliziumeinkristall (cont.)

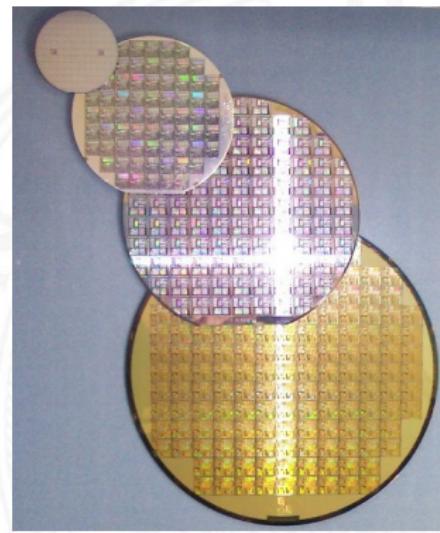
### Zonenschmelz- / Zonenziehverfahren





# Wafer

- ▶ weitere Bearbeitungsschritte:  
zersägen, schleifen, läppen, ätzen, polieren
- ▶ Durchmesser    bis 30 cm  
                      2014: 45 cm
- Dicke            < 1mm
- Rauhigkeit       $\approx$  nm
- ▶ Markieren: Kerben, Lasercodes...  
früher „flats“

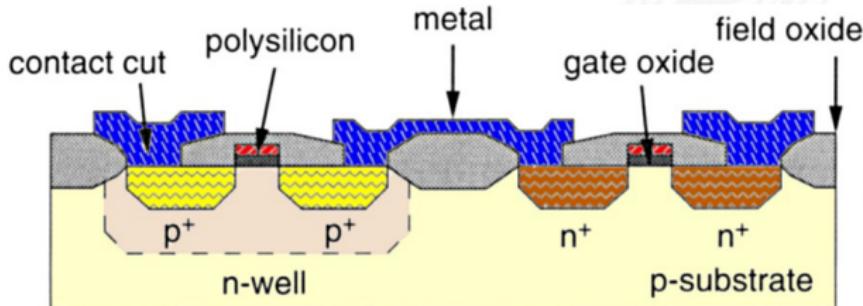




# Technologien

## Technologien zur Erstellung von Halbleiterstrukturen

- ▶ Epitaxie: Aufwachsen von Schichten
- ▶ Oxidation von Siliziumoberflächen:  $SiO_2$  als Isolator
- ▶ Strukturerzeugung durch Lithografie
- ▶ Dotierung des Kristalls durch Ionenimplantation oder Diffusion
- ▶ Ätzprozesse: Abtragen von Schichten





## Technologien (cont.)

### Links

- ▶ <http://www.halbleiter.org>
- ▶ <http://www.siliconfareast.com>
- ▶ <http://www2.renesas.com/fab/en>
- ▶ [http://en.wikipedia.org/wiki/Semiconductor\\_device\\_fabrication](http://en.wikipedia.org/wiki/Semiconductor_device_fabrication)
- ▶ <http://de.wikipedia.org/wiki/Halbleitertechnik>

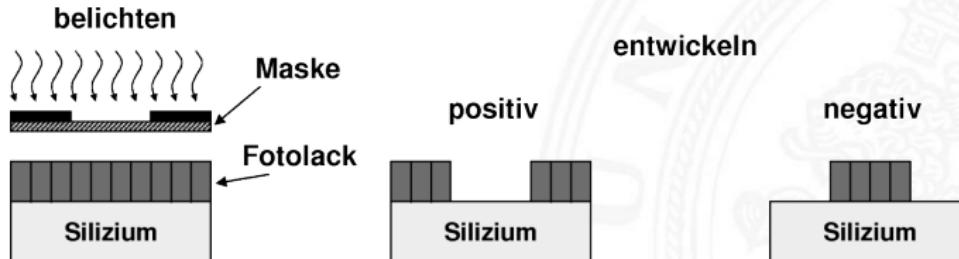
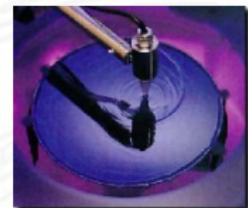


# Lithografie

Übertragung von Strukturen durch einen Belichtungsprozess

## 1. Lack Auftragen (Aufschleudern)

- ▶ Positivlacke: hohe Auflösung  $\Rightarrow$  MOS
- ▶ Negativlacke: robust, thermisch stabil





## Lithografie (cont.)

### 2. „Belichten“

- ▶ Maskenverfahren: 1:1 Belichtung, Step-Verfahren  
UV-Lichtquelle
- ▶ Struktur direkt schreiben: Elektronen- / Ionenstrahl
- ▶ andere Verfahren: Röntgenstrahl- / EUV-Lithografie

### 3. Entwickeln, Härteten, Lack entfernen

- ▶ je nach Lack verschiedene chemische Reaktionsschritte
- ▶ Härtung durch Temperatur

... weitere Schritte des **Planarprozess**

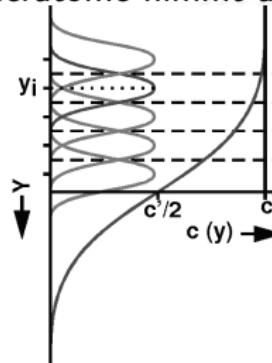
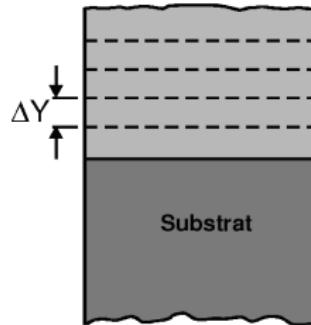


# Dotierung

Fremdatome in den Siliziumkristall einbringen

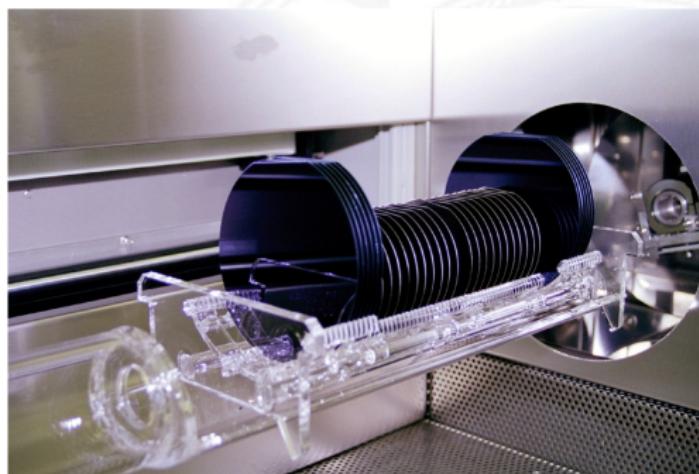
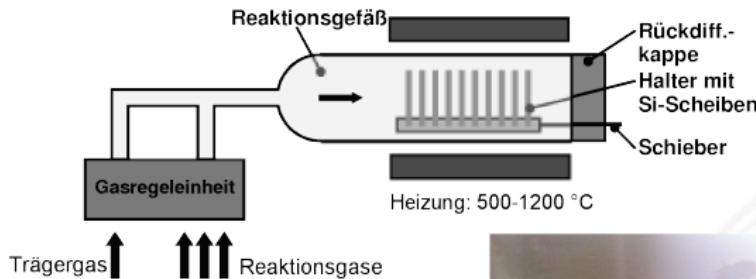
- ▶ Diffusion
  - ▶ Diffusionsofen
  - ▶ gaußförmiges Dotierungsprofil

Konzentration der Dotieratome nimmt ab





## Dotierung (cont.)





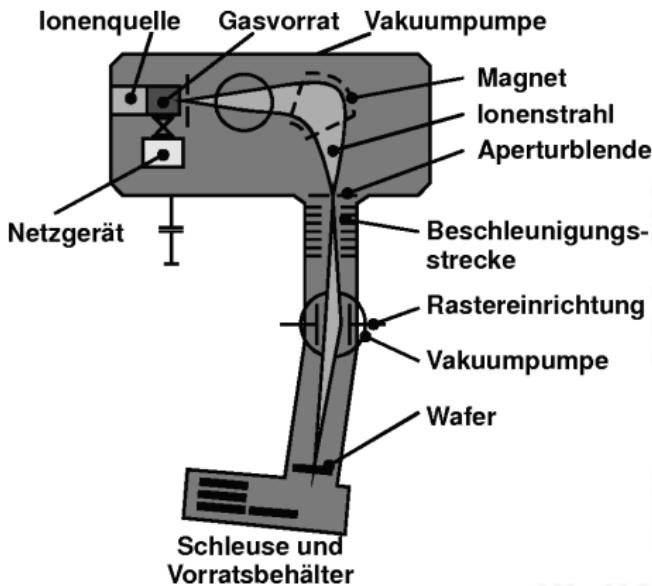
## Dotierung (cont.)

- ▶ Ionenimplantation
  - ▶ „Beschuss“ mit Ionen
  - ▶ Beschleunigung der Ionen im elektrischen Feld
  - ▶ Über die Energie der Ionen kann die Eindringtiefe sehr genau eingestellt werden
  - ▶ „Temperung“ notwendig: Erhitzen des Einkristalls zur Neuorganisation des Kristallgitters



## Dotierung (cont.)

### Ionenimplantation



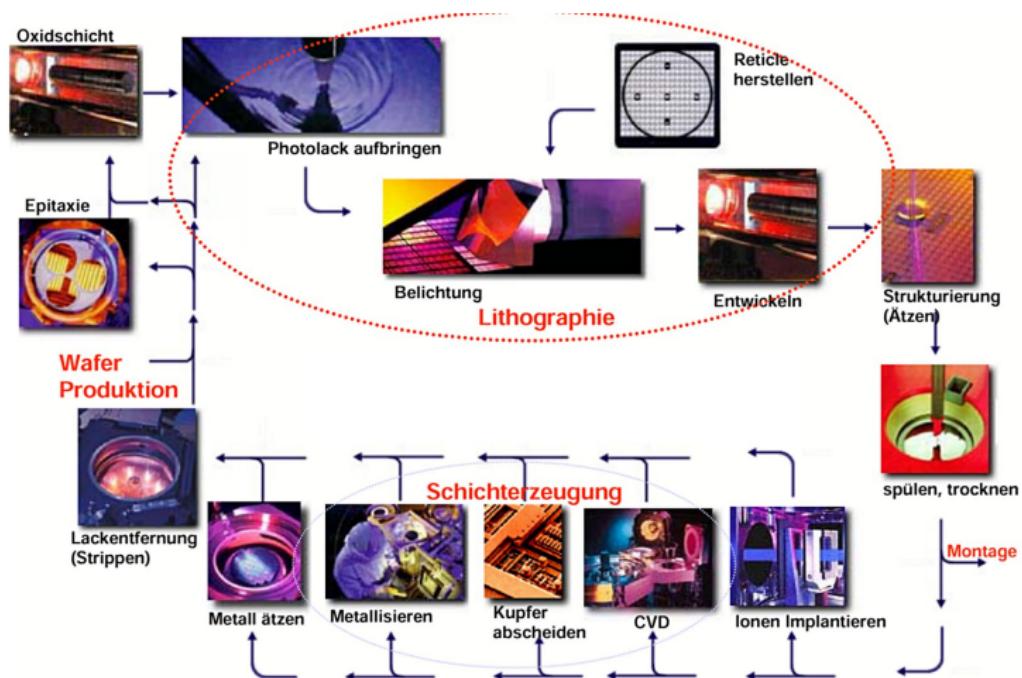


# Planarprozess

- ▶ Der zentrale Ablauf bei der Herstellung von Mikroelektronik
- ▶ Ermöglicht die gleichzeitige Fertigung aller Komponenten auf dem Wafer
- ▶ Schritte
  1. Vorbereiten / Beschichten des Wafers:  
Oxidation, CVD, Aufdampfen, Sputtern...
  2. Strukturieren durch Lithografie
  3. Übertragen der Strukturen durch Ätzprozesse
  4. Modifikation des Materials: Dotierung, Oxidation
  5. Vorbereitung für die nächsten Prozessschritte...



# Planarprozess (cont.)





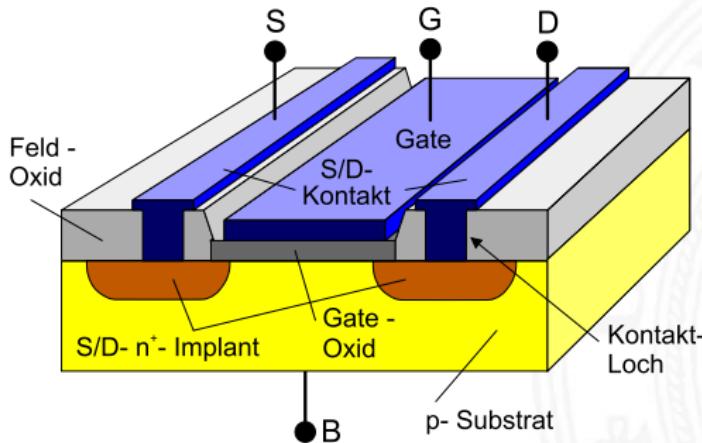
# MOS-Transistor

- ▶ MOS: Metal Oxide Semiconductor  
FET : Feldeffekttransistor
  - ▶ <http://olli.informatik.uni-oldenburg.de/weTEiS/weteis/tutorium.htm>
  - ▶ <http://de.wikipedia.org/wiki/Feldeffekttransistor>
  - ▶ <http://de.wikipedia.org/wiki/MOSFET>
- ▶ unipolarer Transistor: nur eine Art von Ladungsträgern, die Majoritätsträger, ist am Stromfluss beteiligt
  - im Gegensatz zu Bipolartransistoren
  - siehe z.B.: U. Tietze, C. Schenk, *Halbleiter-Schaltungstechnik*



## MOS-Transistor (cont.)

- Anschlüsse:
  - Source** Quelle der Ladungsträger
  - Gate** steuert den Stromfluss
  - Drain** Senke der Ladungsträger
  - Bulk** siehe „Herstellungstechnik“

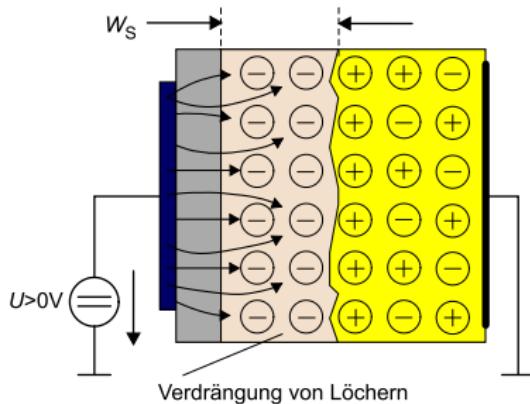


N. Reischneider,  
*CAE-gestützte IC-Entwurfsmethoden*



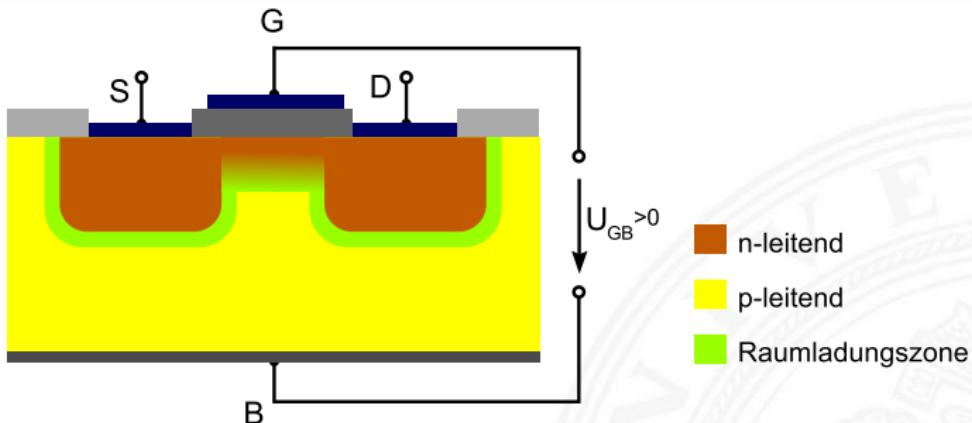
## MOS-Transistor (cont.)

- Funktionsweise: die Ladung des Gates erzeugt ein elektrisches Feld. Durch Inversion werden Ladungsträger unterhalb des Gates verdrängt und ein leitender Kanal zwischen Source und Drain entsteht.





## MOS-Transistor (cont.)

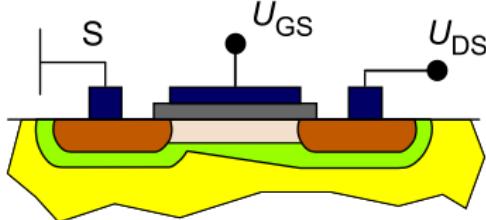


- ▶ Raumladungszone: neutral, keine freien Ladungsträger
  - ▶ Schwellspannung  $U_P$ : abhängig von der Dotierungsdichte, den Parametern des MOS-Kondensators (Dicke und Material der Gate-Isolationsschicht)...
- $U_P$  möglichst klein: 0,3...0,8 V früher: deutlich mehr

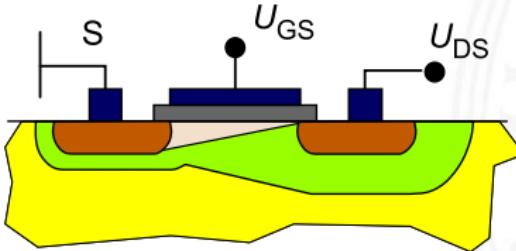


## MOS-Transistor (cont.)

- $U_{DS} \ll U_{GS} - U_P$  normaler Betrieb (Triodenbereich)



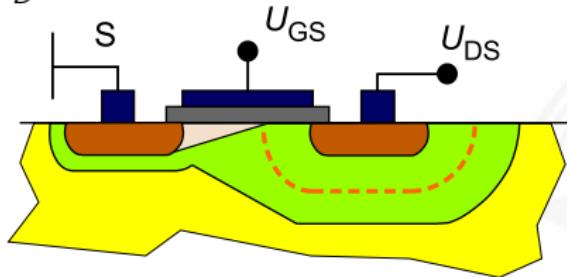
- $U_{DS} = U_{GS} - U_P$  Kanalabschnürung  
Spannungsabfall zwischen S und D durch den Kanalwiderstand





## MOS-Transistor (cont.)

- $U_{DS} > U_{GS} - U_P$  Kanalverkürzung (Sättigungsbereich)  
Der Kanal wird weiter verkürzt, die Spannung  $U_{DS}$  bewirkt ein virtuell größeres Drain durch Inversion.  
 $I_D$  wächst nur noch minimal.

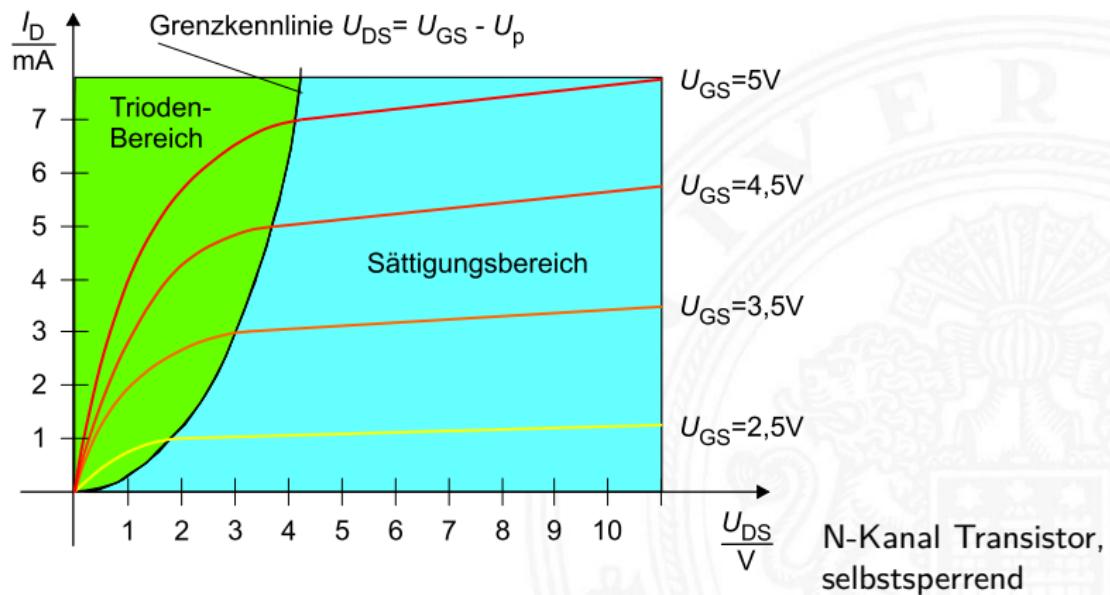


- ⇒ kurze Kanäle aktueller Submikronprozesse können allein durch hohe Spannungen  $U_{DS}$  leitend werden (Durchgreifbetrieb)
- ⇒ einer der Gründe für sinkende Versorgungsspannungen



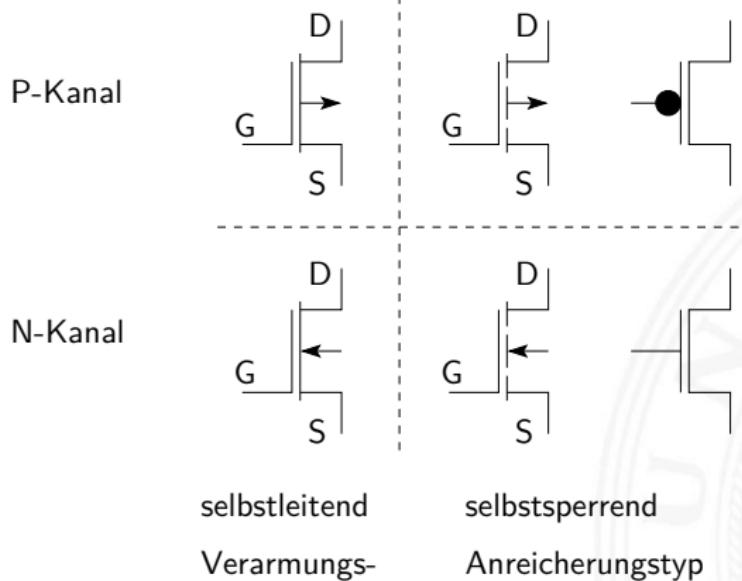
## MOS-Transistor (cont.)

### ► Kennlinienfeld





# MOS-Transistor: Schaltsymbole





# CMOS-Technologie

**Complementary Metal-Oxide Semiconductor:** die derzeit dominierende Technologie für alle hochintegrierten Schaltungen

- ▶ Schaltungsprinzip nutzt n-Kanal und p-Kanal Transistoren
  - ▶ alle elementaren Gatter verfügbar
  - ▶ effiziente Realisierung von *Komplexgattern*
  - ▶ *Transmission-Gate* als elektrischer Schalter
  - ▶ effiziente Realisierung von Flipflops und Speichern
- 
- + sehr hohe Integrationsdichte möglich, gut skalierbar
  - + sehr schnelle Schaltgeschwindigkeit der Gatter
  - + sehr geringer Stromverbrauch pro Gatter möglich
  - + Integration von digitalen und analogen Komponenten



# CMOS: Überblick

- ▶ Schaltungsprinzip
- ▶ Inverter und nicht-invertierender Verstärker
- ▶ NAND, NAND3, NOR (und AND, OR)
- ▶ XOR
  
- ▶ Komplexgatter
- ▶ Transmission-Gate
  
- ▶ Beispiele für Flipflops
- ▶ SRAM



## CMOS: Schaltungsprinzip von „static CMOS“

- ▶ Transistoren werden als Schalter betrachtet
- ▶ zwei zueinander **komplementäre** Zweige der Schaltung
  - ▶ n-Kanal Transistoren zwischen Masse und Ausgang  $y$       1 on
  - ▶ p-Kanal                –"–                Vdd      und Ausgang  $y$       0 on
- ▶ p-Kanal Zweig komplementär („dualer Graph“) zu n-Kanal Zweig:  
jede Reihenschaltung von Elementen wird durch eine  
Parallelschaltung ersetzt (und umgekehrt)
- ▶ immer ein direkt leitender Pfad von entweder Vdd („1“)  
oder Masse / Gnd („0“) zum Ausgang
- ▶ niemals ein direkt leitender Pfad von Vdd nach Masse
- ▶ kein statischer Stromverbrauch im Gatter



# Inverter

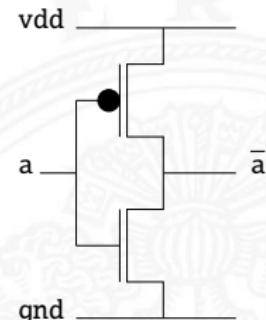


## Funktionsweise

- ▶ selbstsperrende p- und n-Kanal Transistoren
- ▶ komplementär beschaltet
- ▶ Ausgang: Pfad über p-Transistoren zu  $Vdd$   
– " – n-Transistoren zu  $Gnd$
- ▶ genau *einer* der Pfade leitet
- ▶ Eingang  $Trans_P$   $Trans_N$  Ausgang

---

 $a = 0 \rightarrow$  leitet / sperrt  $\rightarrow$  über  $T_P$  mit  $Vdd$  verbunden = 1  
 $a = 1 \rightarrow$  sperrt / leitet  $\rightarrow$  über  $T_N$  mit  $Gnd$  verbunden = 0

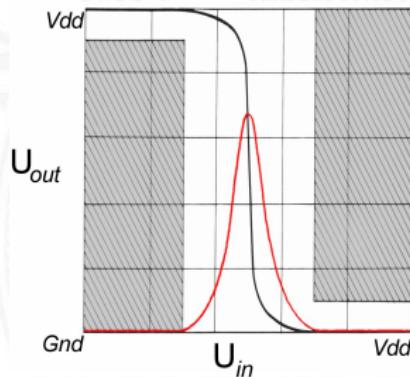
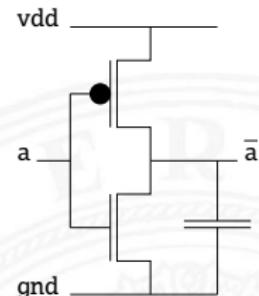




## Inverter (cont.)

### Leistungsaufnahme

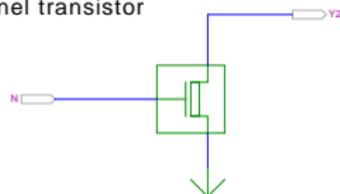
1.  $U_{in} = 0$ , bzw.  $Vdd$ : Sperrstrom, nur  $\mu A$   
 $\Rightarrow$  niedrige statische Leistungsaufnahme
2. Querstrom beim Umschalten:  
 kurzfristig leiten beide Transistoren  
 $\Rightarrow$  Forderung nach steilen Flanken
3. Kapazitive Last: Fanout-Gates  
 Energie auf Gate(s):  $W = \frac{1}{2} C_T Vdd^2$   
 Verlustleistung<sub>(0/1/0)</sub>:  $P = C_T Vdd^2 \cdot f$



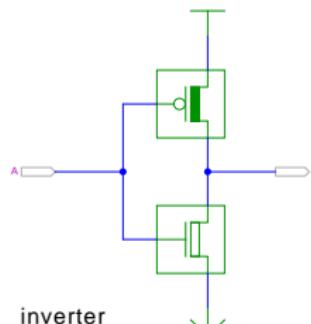
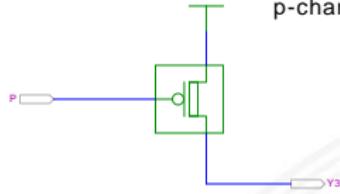


# Hades: n- und p-Kanal Transistor, Inverter, Verstärker

n-channel transistor

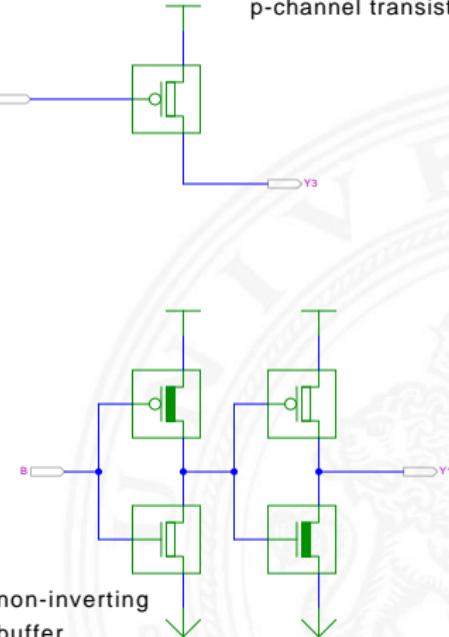


p-channel transistor



inverter

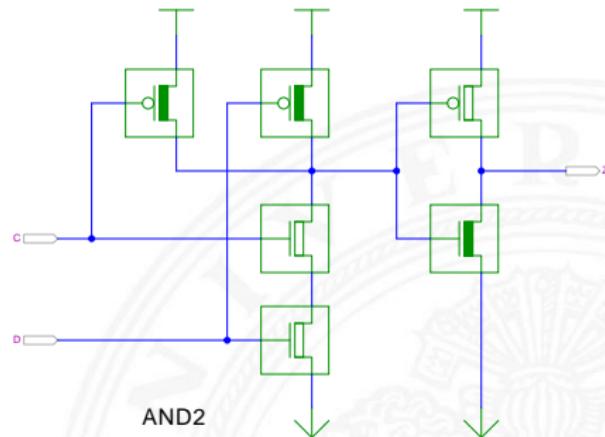
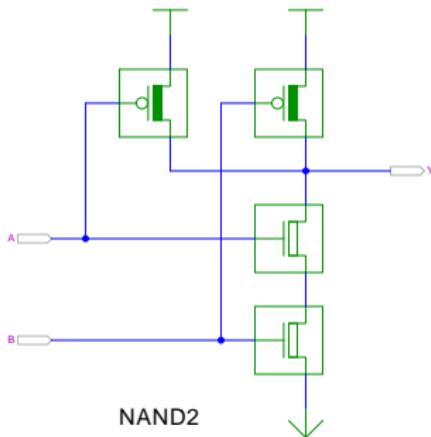
$$Y = \neg A$$

non-inverting  
buffer

$$Y = \neg\neg A = A$$



# NAND- und AND-Gatter

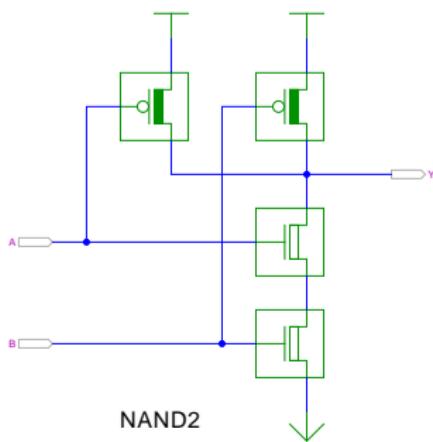


- ▶ NAND: n-Transistoren in Reihe, p-Transistoren parallel
- ▶ AND: Kaskade aus NAND und Inverter

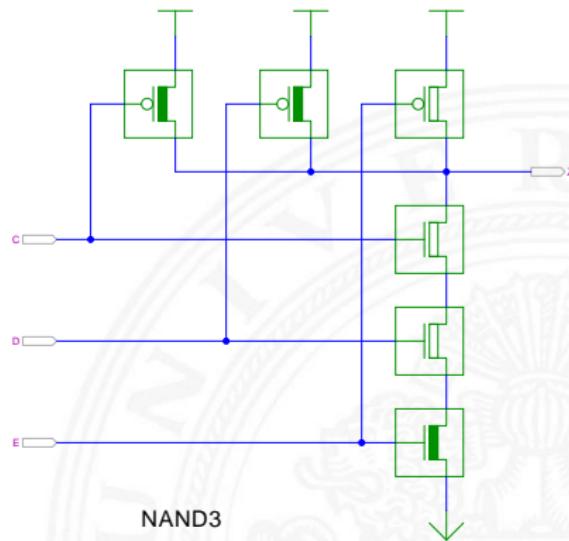
[tams.informatik.uni-hamburg.de/applets/hades/webdemos/05-switched/40-cmos](http://tams.informatik.uni-hamburg.de/applets/hades/webdemos/05-switched/40-cmos)



## NAND- und AND-Gatter (cont.)



NAND2



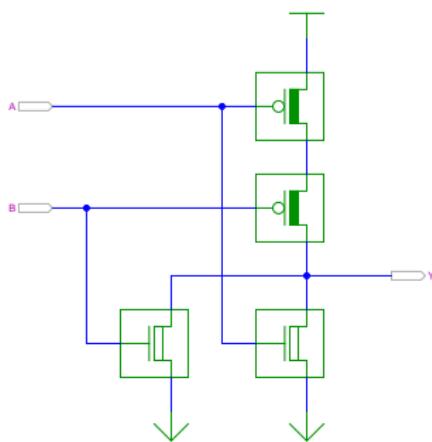
NAND3

- ▶ n-Transistoren in Reihe, p-Transistoren parallel
- ▶ normalerweise max. 4 Transistoren in Reihe (Spannungsabfall)

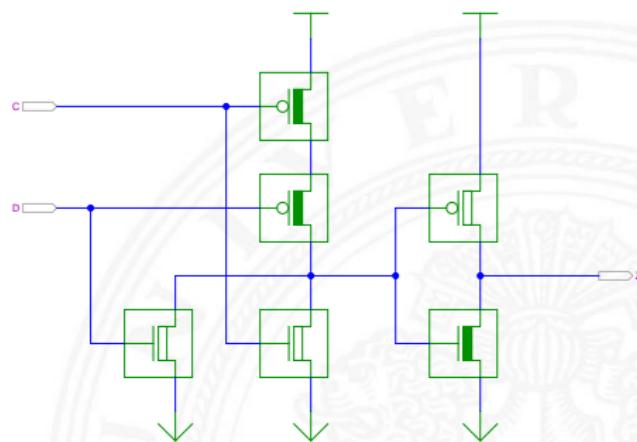


# NOR- und OR-Gatter

NOR2



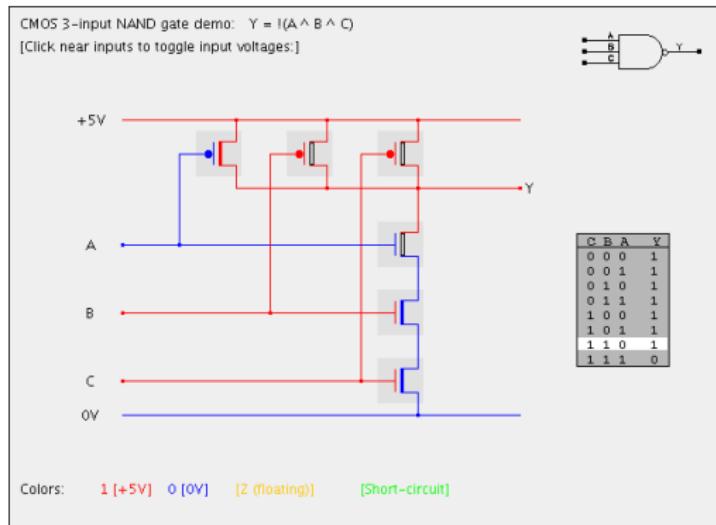
OR2



- ▶ Struktur komplementär zum NAND/AND
- ▶ n-Transistoren parallel, p-Transistoren in Reihe
- ▶ p-Transistoren schalten träge: etwas langsamer als NAND



# CMOS-Technologie: Demos



- ▶ Interaktive Demonstration der CMOS-Grundgatter (Java)  
<http://tams.informatik.uni-hamburg.de/applets/cmos/>



# CMOS: Komplexgatter

## Gatterfunktionen

- ▶ Schaltungen: *negierte monotone boole'sche Funktionen*
- ▶ Beliebiger schaltalgebraischer Ausdruck *ohne Negation*:  $\vee, \wedge$
- ▶ Negation des gesamten Ausdrucks: Ausgang *immer negiert*
- ▶ je Eingang: ein Paar p-/n-Kanal Transistoren
- ▶ Dualitätsprinzip: n- und p-Teil des Gatters

n-Teil      p-Teil      Logik, ohne Negation

---

seriell  $\Leftrightarrow$  parallel  $\equiv \wedge /$  und

parallel  $\Leftrightarrow$  seriell  $\equiv \vee /$  oder



# CMOS: Komplexgatter (cont.)

- ▶ Konstruktion
  1. n-Teil aus Ausdruck ableiten  
beliebige Parallel- und Serienschaltung der n-Transistoren
  2. p-Teil dual dazu entwickeln  
komplementäre Seriell- und Parallelschaltung der p-Transistoren
    - ▶ typischerweise max. 4 Transistoren in Reihe
- ▶ viele invertierende logische Funktionen effizient realisierbar
- ▶ Schaltungslayout automatisch synthetisierbar
- ▶ zwei gängige Varianten
  - ▶ AOI-Gatter („AND-OR-invert“)
  - ▶ OAI-Gatter („OR-AND-invert“)



# Komplexgatter

Beispiel:  $\overline{(a \wedge b \wedge c) \vee d \vee (e \wedge f)}$

„AOI321-Gatter“, AND-OR-INVERT Struktur

- ▶ AND-Verknüpfung von (a,b,c)
- ▶ AND-Verknüpfung von (e,f)
- ▶ NOR-Verknüpfung der drei Terme
  
- ▶ direkte Realisierung hätte  $(6+2)+(0)+(4+2)+6 = 18$  Transistoren
- ▶ Komplexgatter mit 12 Transistoren



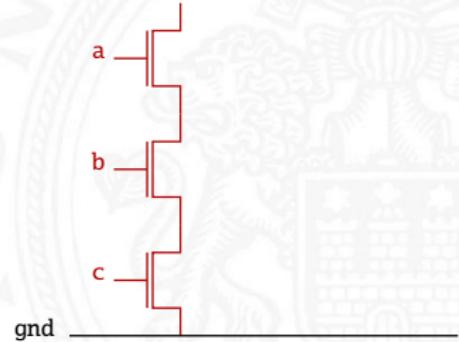
## Komplexgatter (cont.)

Beispiel:  $\overline{(a \wedge b \wedge c) \vee d \vee (e \wedge f)}$



## Komplexgatter (cont.)

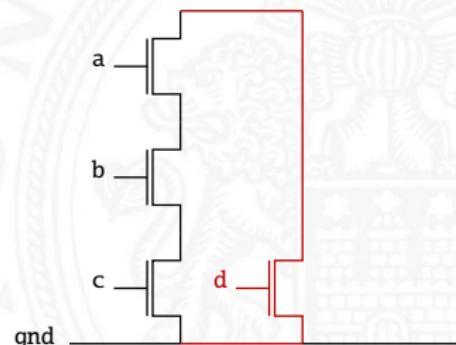
Beispiel:  $\overline{(a \wedge b \wedge c)} \vee d \vee (e \wedge f)$





## Komplexgatter (cont.)

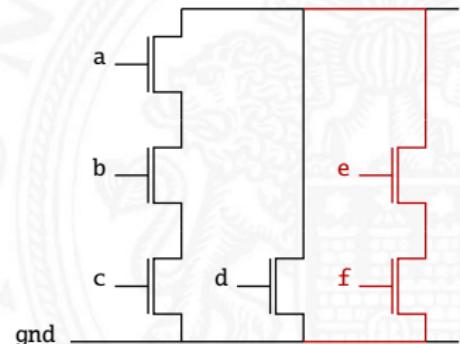
Beispiel:  $\overline{(a \wedge b \wedge c) \vee d \vee (e \wedge f)}$





## Komplexgatter (cont.)

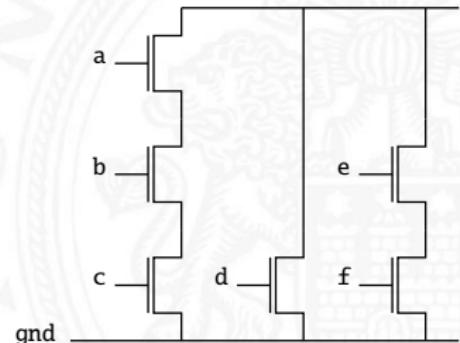
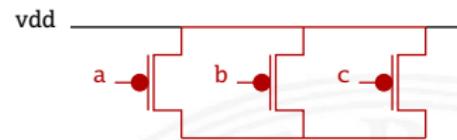
Beispiel:  $\overline{(a \wedge b \wedge c) \vee d} \vee \overline{(e \wedge f)}$





## Komplexgatter (cont.)

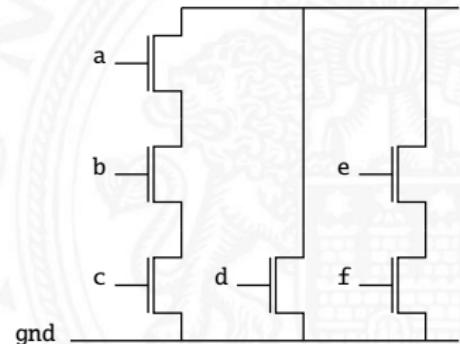
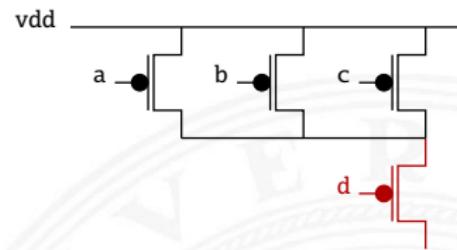
Beispiel:  $\overline{(a \wedge b \wedge c)} \vee d \vee (e \wedge f)$





## Komplexgatter (cont.)

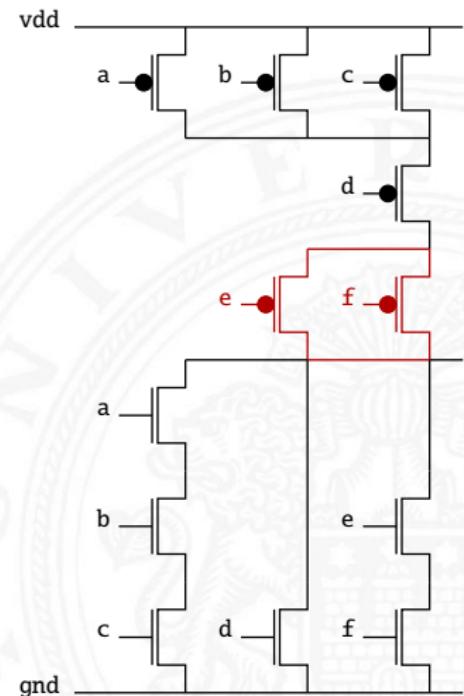
Beispiel:  $\overline{(a \wedge b \wedge c)} \vee d \vee (e \wedge f)$





## Komplexgatter (cont.)

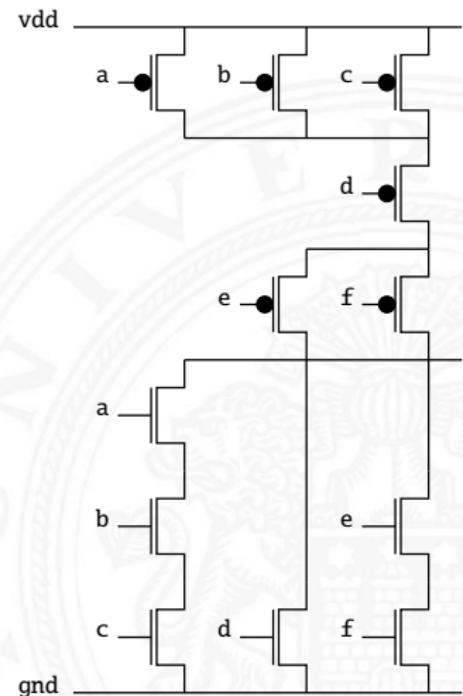
Beispiel:  $\overline{(a \wedge b \wedge c)} \vee d \vee \overline{(e \wedge f)}$





## Komplexgatter (cont.)

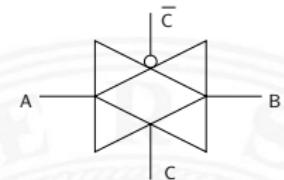
Beispiel:  $\overline{(a \wedge b \wedge c)} \vee d \vee (e \wedge f)$





# Transmission-Gate

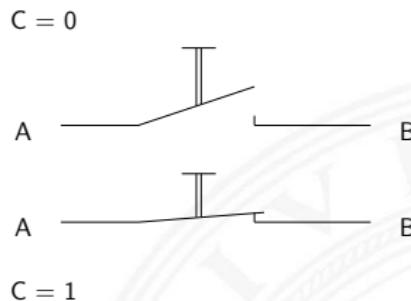
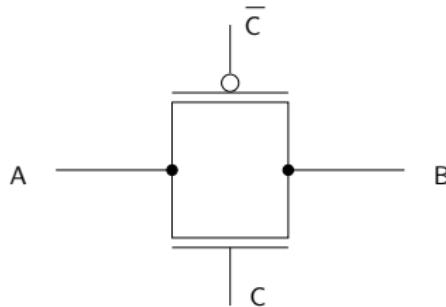
**Transmissions-Gatter** (*transmission gate, t-gate*)



- ▶ Paar aus je einem n- und p-Kanal MOS-Transistor
- ▶ symmetrische Anordnung
  
- ▶ Ansteuerung der Gate-Elektroden mit invertierter Polarität
  - ⇒ entweder beide Transistoren leiten, oder beide sperren
  
- ▶ Funktion entspricht **elektrisch gesteuertem Schalter**
- ▶ effiziente Realisierung vieler Schaltungen

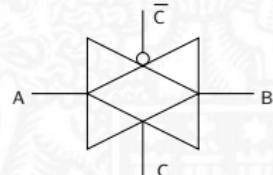


## Transmission-Gate (cont.)



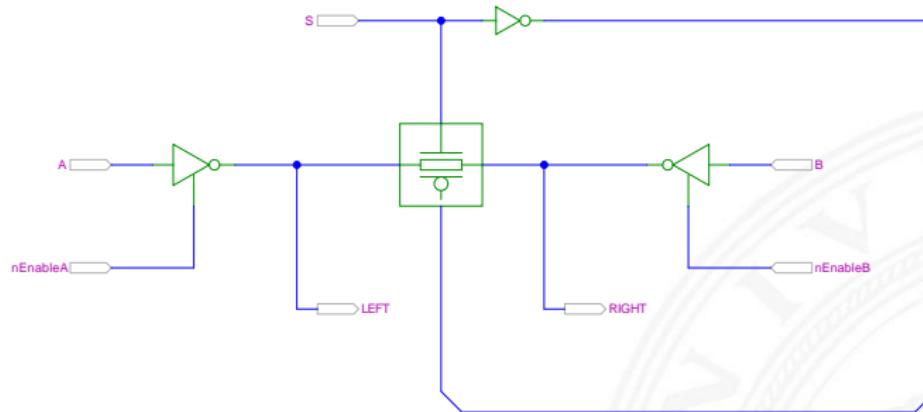
elektrisch gesteuerter Schalter:

- ▶  $C = 0$ : keine Verbindung von A nach B
- ▶  $C = 1$ : leitende Verbindung von A nach B
- ▶ symmetrisch in beide Richtungen





# Transmission-Gate: Demo



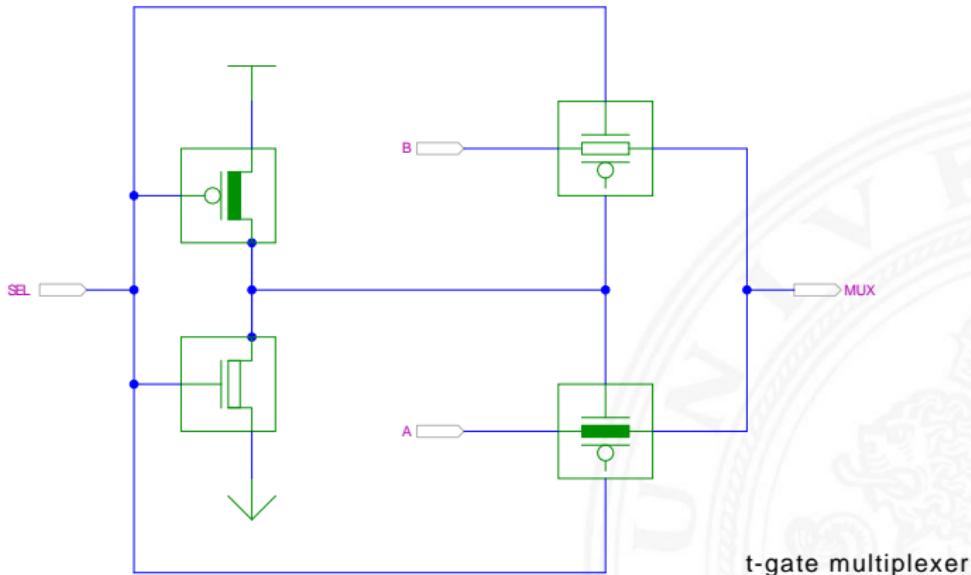
tgate demonstration

- ▶ Werte  $A$  und  $B$  anlegen, Treiber mit enable-Signalen aktivieren
- ▶ Gatter mit  $S$  ein- oder ausschalten

[tams.informatik.uni-hamburg.de/applets/hades/webdemos/05-switched/40-cmos/tgate.html](http://tams.informatik.uni-hamburg.de/applets/hades/webdemos/05-switched/40-cmos/tgate.html)



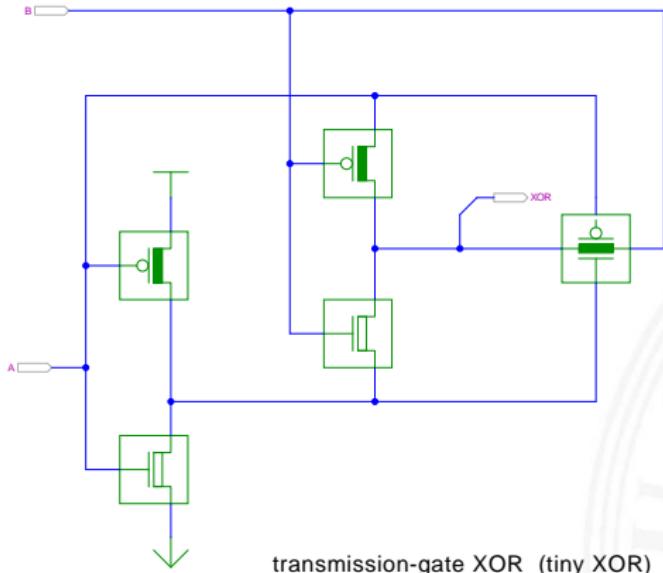
# T-Gate Multiplexer



- ▶ kompakte Realisierung (4 bzw. 6 Transistoren)
- ▶ Eingänge  $a$  und  $b$  nicht verstärkt  $\Rightarrow$  nur begrenzt kaskadierbar



# T-Gate XOR-Gatter



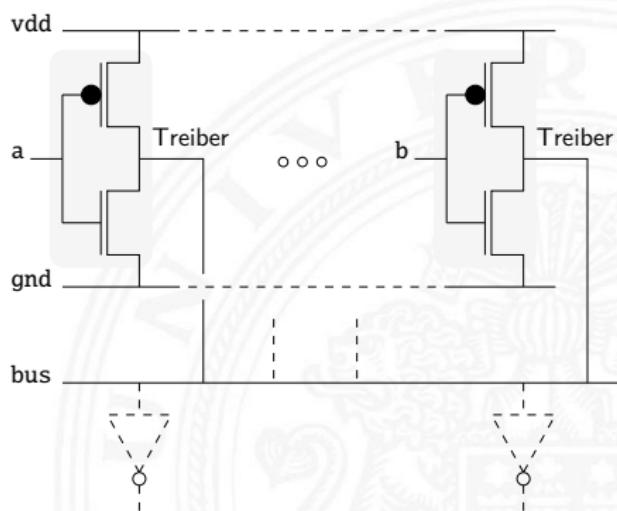
- ▶ kompakte Realisierung des XOR (nur 6 Transistoren)
- ▶ Eingang  $b$  nicht verstärkt  $\Rightarrow$  nur begrenzt kaskadierbar



# Tristate-Treiber

## Bussysteme

- ▶ Quellen: „Bustreiber“
- ▶ Senken : Gattereingänge
- ▶ Probleme
  - ▶ Kurzschluss
  - ▶ offene Eingänge
- ⇒ Tristate



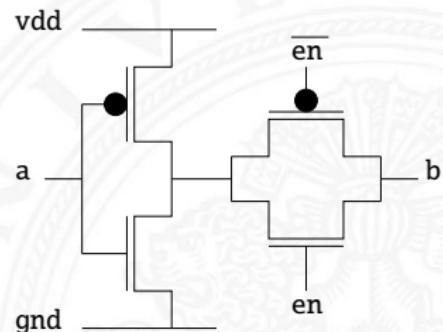
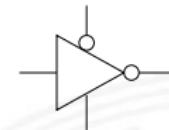


## Tristate-Treiber (cont.)

Beispiel: Tristate-Inverter

Funktionsweise

- ▶ Ausgang elektrisch trennen z.B. mit Transmission-Gate
- ▶ 3-Pegel: 0, 1, Z *hochohmig*



- ▶ Enable Verbindung Ausgang

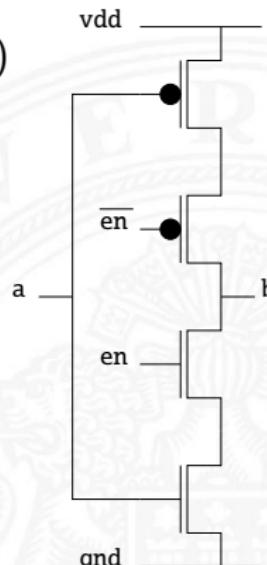
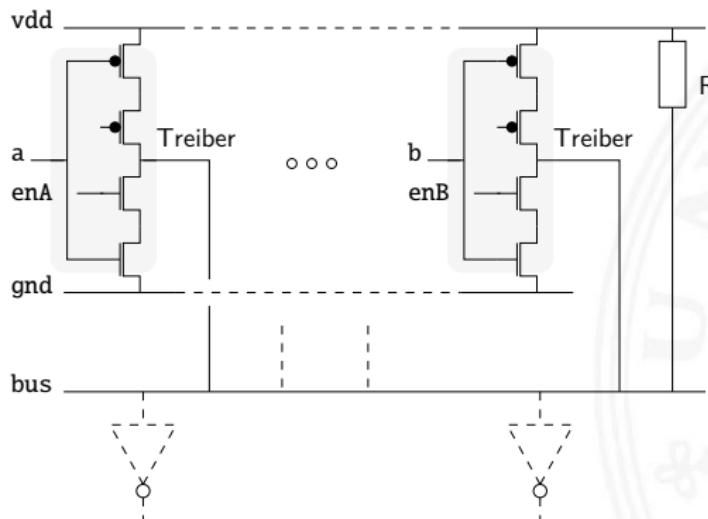
$en = 0 \rightarrow$ getrennt	$\rightarrow bus = Z$	hochohmig
$en = 1 \rightarrow$ geschlossen	$\rightarrow bus = \neg a$	$f(a)$



## Tristate-Treiber (cont.)

### Tristate-Bussystem

- ▶ *pull-up/-down* Widerstand R (offene Eingänge)
- ▶ nur **genau ein** Treiber gleichzeitig aktiv





# Latch / Flipflop: Speichertechnik

## Methoden der Implementation

### 1. statisch

- ▶ Speicherung: Rückkopplung von (statischen) Gattern  
siehe: „Schaltwerke – Flipflops“
  - + taktunabhängig
  - + sicher

### 2. quasi-statisch

- ▶ Speicherung: Rückkopplung von Gattern
- ▶ Transmission-Gates als Multiplexer
  - + taktunabhängig
  - + kleiner



## Latch / Flipflop: Speichertechnik (cont.)

### 3. dynamisch

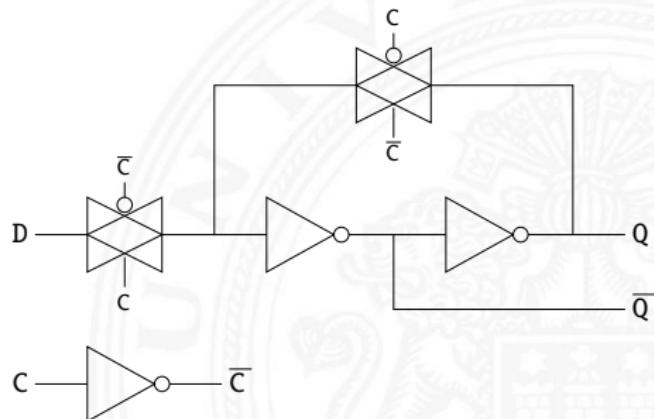
- ▶ Speicherung: Gate-Kapazitäten
- ▶ verschiedene Taktschemata/Schaltungsvarianten
- muss getaktet werden
- schwieriger zu Entwerfen (wegen Taktschema)
- + Integration in Datenpfade (arithmetische Pipelines)
- + sehr hohe Taktfrequenzen
- + sehr klein



## D-Latch: quasi-statisch

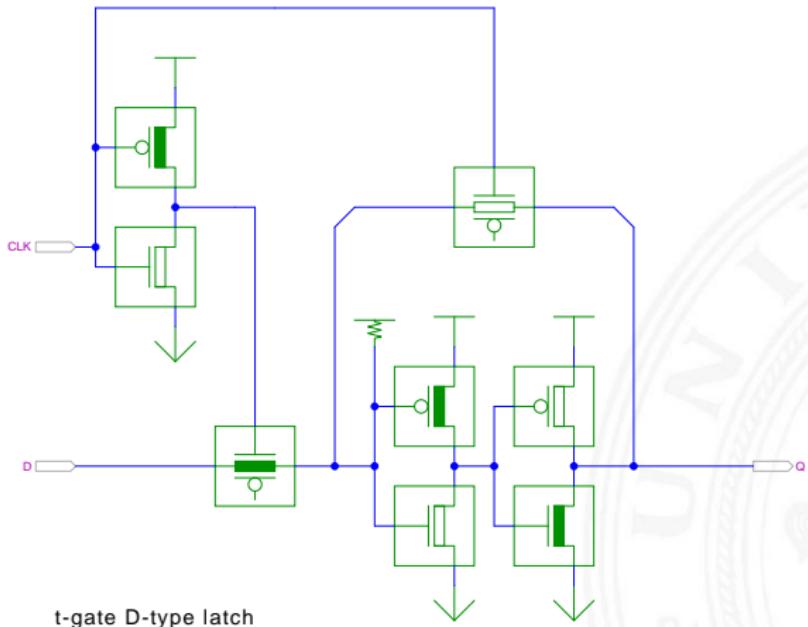
- ▶ Transmission-Gates als Schalter

$C = 1$  Transparent: Eingang über die Inverter zum Ausgang  
 $C = 0$  Speicherung: Rückkopplungspfad aktiv





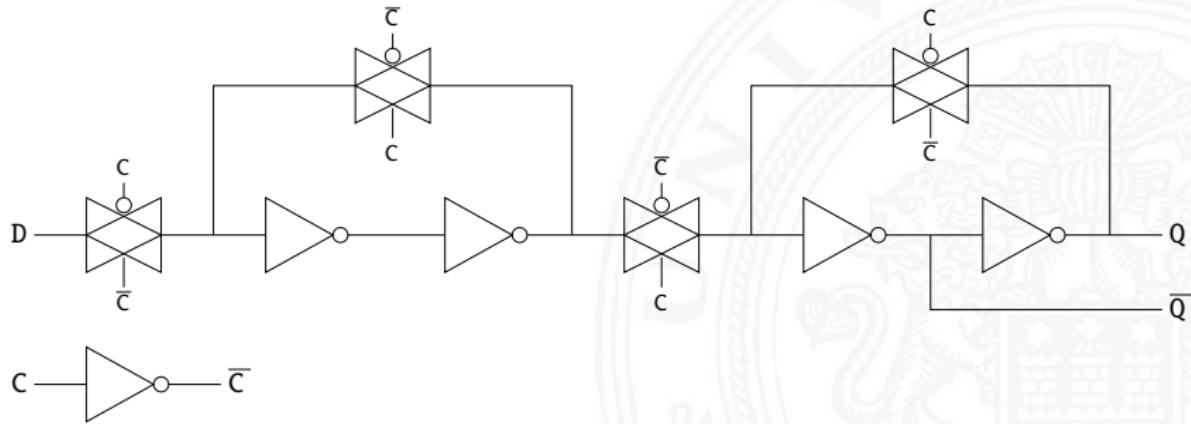
## D-Latch: quasi-statisch (cont.)





## D-Flipflop: quasi-statisch

- ▶ Aufbau aus zwei Latches
  - ▶ Vorderflanke: low-Transparent + high-Transparent
- D-FF, vorderflankengest.

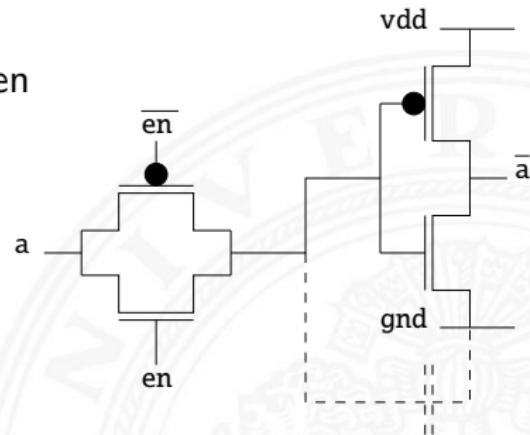




# dynamische Speicherung

## Schaltungsprinzip

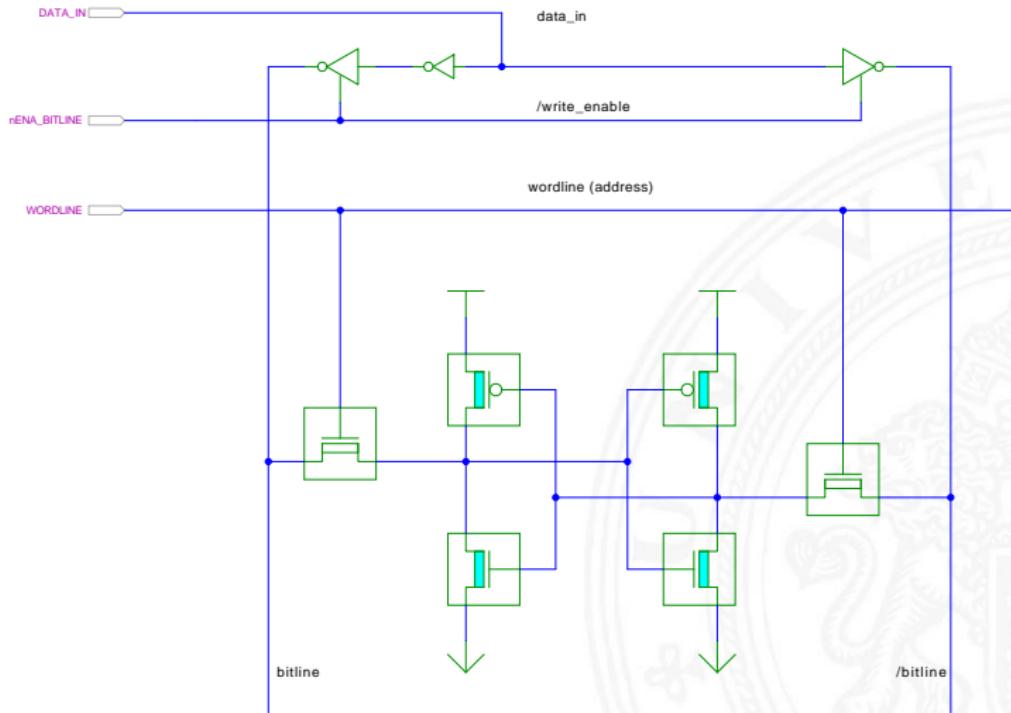
- ▶ Speicherung auf Gate-Kapazitäten



- ▶ viele unterschiedliche Takte / Funktionsweisen
- ▶ Verbindung mit Logikgattern möglich  
⇒ arithmetische Pipelines
- ... aus Zeitgründen nicht weiter vertieft

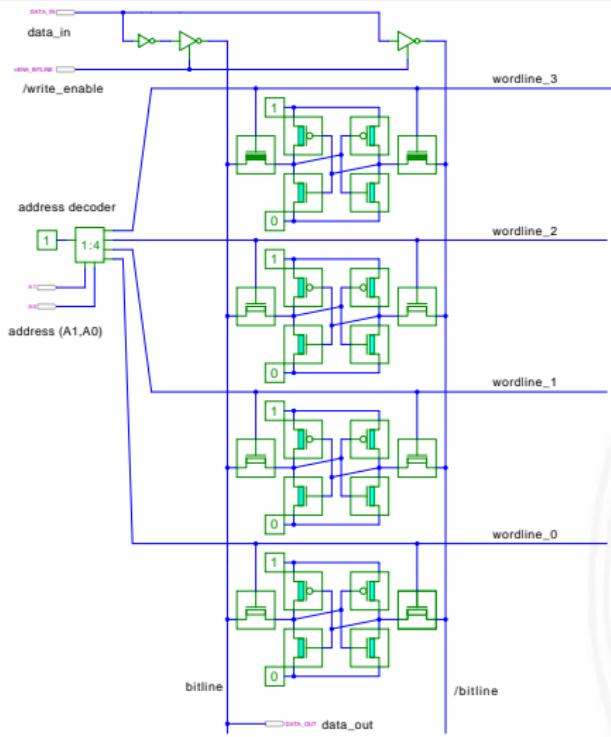


# SRAM: Sechs-Transistor Speicherstelle („6T“)





# Prinzip des SRAM



Hades Webdemos: 05-switched/40-cmos/sram4

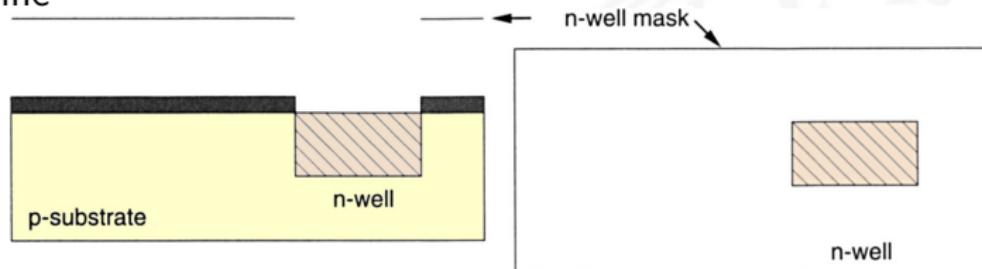


# CMOS Prozessschritte: Inverter

## Ein n-Wannen Prozesses

Weste, Eshragian, *Principles of CMOS VLSI Design*

1. Ausgangsmaterial: p-dotiertes Substrat
2. n-Wanne

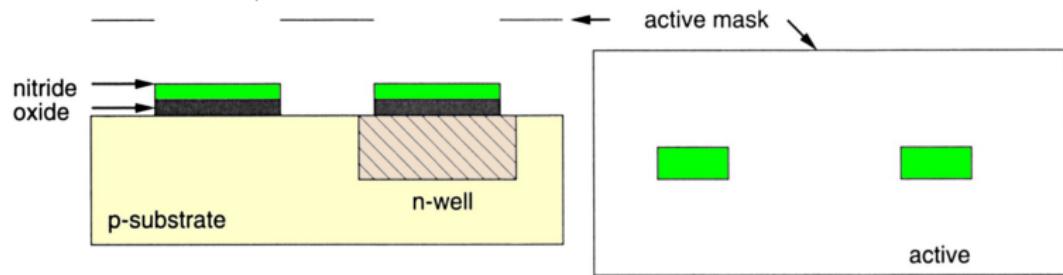


- ▶ Dotierung für p-Kanal Transistoren
- ▶ Herstellung: Ionenimplantation oder Diffusion



## CMOS Prozessschritte: Inverter (cont.)

### 3. „aktive“ Fläche / Dünnoxid

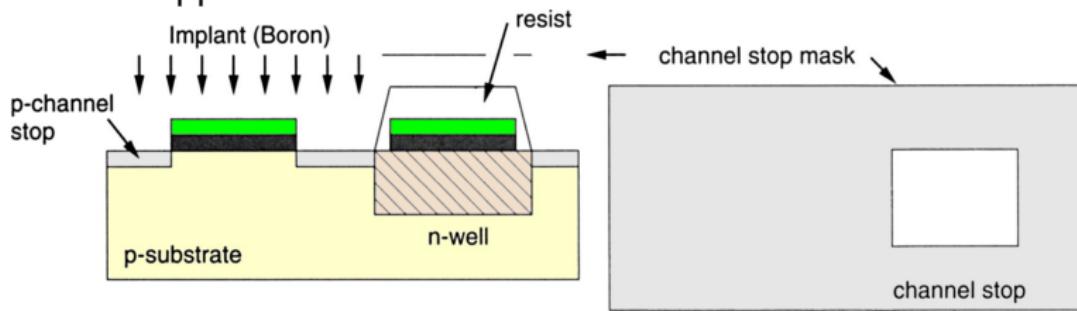


- ▶ Spätere Gates und  $p^+/-n^+$ -Gebiete
- ▶ Herstellung: Epitaxie  $SiO_2$  und Abdeckung mit  $Si_3N_4$



## CMOS Prozessschritte: Inverter (cont.)

### 4. p-Kanalstopp

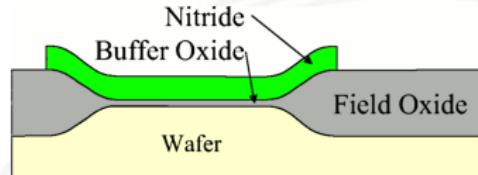
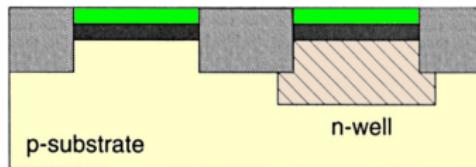


- ▶ Begrenzt n-Kanal Transistoren
- ▶ p-Wannen Maske, bzw.  $\neg$  n-Wanne
- ▶ Maskiert durch Resist und  $Si_3N_4$
- ▶ Substratbereiche in denen keine n-Transistoren sind
- ▶ Herstellung: p<sup>+</sup>-Implant (Bor)
- ▶ n-Kanalstop aktueller Prozesse: analog dazu



## CMOS Prozessschritte: Inverter (cont.)

5. Resist entfernen
6. Feldoxid aufwachsen –  $SiO_2$



- ▶ LOCOS: **L**ocal **O**xidation of **S**ilicon
- ▶ Maskiert durch  $Si_3N_4$
- ▶ Wächst auch lateral unter  $Si_3N_4/SiO_2$  (aktive) Bereiche  
engl. *bird's beak*
- ▶ Der aktive Bereich wird kleiner als vorher maskiert
- ▶ Herstellung: Epitaxie und Oxidation
- ▶ Problem: nicht plane Oberfläche



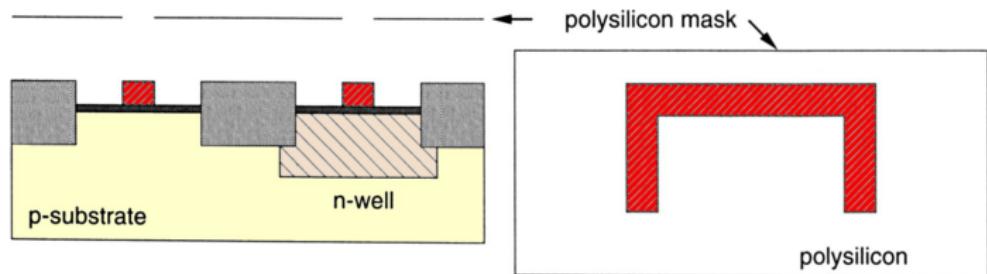
## CMOS Prozessschritte: Inverter (cont.)

7.  $Si_3N_4$  entfernen, Gateoxid bleibt  $SiO_2$
8. Transistor Schwellspannungen „justieren“
  - ▶ Meist wird das Polysilizium zusätzlich n<sup>+</sup> dotiert  
Grund: bessere Leitfähigkeit
  - ▶ Problem:  $U_D(T_N) \approx 0,5 \dots 0,7 \text{ V}$   
 $U_D(T_P) \approx -1,5 \dots -2,0 \text{ V}$
  - ▶ Maske: n-Wanne, bzw. p-Wanne
  - ▶ Herstellung: Epitaxie einer leicht negativ geladenen Schicht an der Substratoberfläche



## CMOS Prozessschritte: Inverter (cont.)

### 9. Polysilizium Gate

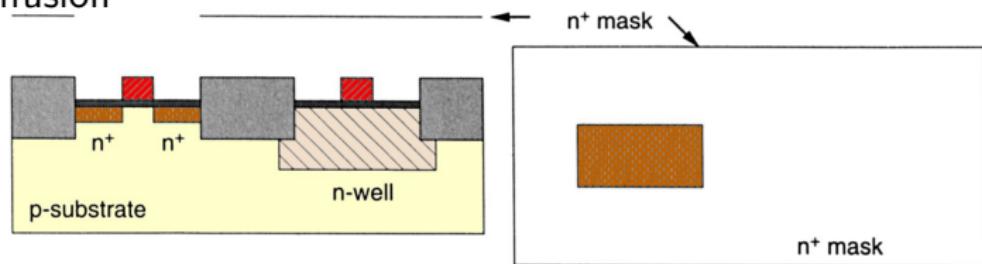


- ▶ Herstellung: Epitaxie von Polysilizium, Ätzen nach Planarprozess



## CMOS Prozessschritte: Inverter (cont.)

### 10. n<sup>+</sup>-Diffusion



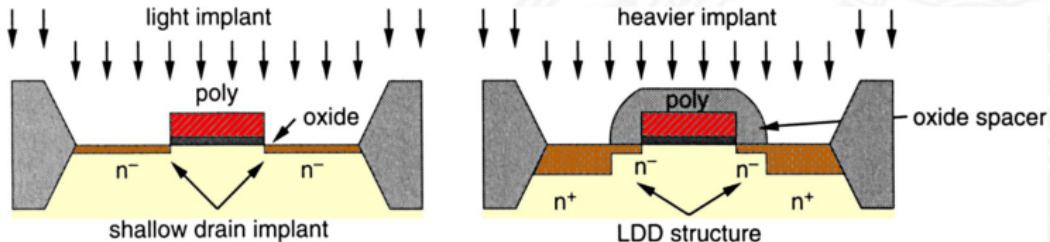
- ▶ Erzeugt Source und Drain der n-Kanal Transistoren
- ▶ Maskiert durch aktiven Bereich, n<sup>+</sup>-Maske und Polysilizium  
⇒ Selbstjustierung
- ▶ Dotiert auch das Polysilizium Gate leicht (s.o.)
- ▶ Herstellung: Ionenimplantation, durchdringt Gateoxid



## CMOS Prozessschritte: Inverter (cont.)

### Zusätzliche Schritte bei der Source/Drain Herstellung

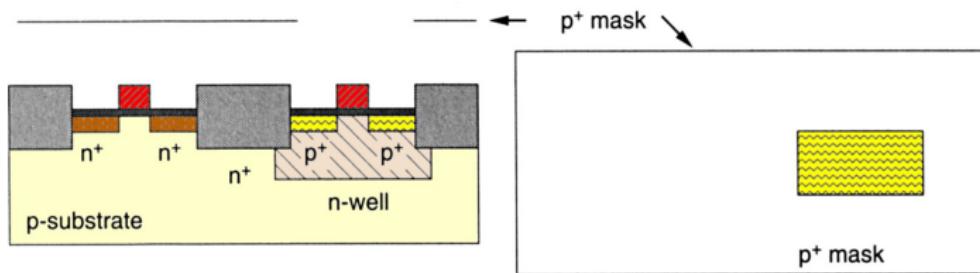
- ▶ Problem „Hot-Carrier“ Effekte (schnelle Ladungsträger):  
Stoßionisation, Gateoxid wird durchdrungen...
- ▶ Lösung: z.B. LDD (Lightly Doped Drain)
  - a. „flaches“ n-LDD Implant
  - b. zusätzliches  $SiO_2$  über Gate aufbringen (*spacer*)
  - c. „normales“  $n^+$ -Implant
  - d. Spacer  $SiO_2$  entfernen





# CMOS Prozessschritte: Inverter (cont.)

## 11. p<sup>+</sup>-Diffusion



- ▶ Erzeugt Source und Drain der p-Kanal Transistoren
- ▶ Maskiert durch aktiven Bereich, p<sup>+</sup>-Maske und Polysilizium  
⇒ Selbstjustierung
- ▶ teilweise implizite p<sup>+</sup>-Maske =  $\neg$  n<sup>+</sup>-Maske
- ▶ wenig schnelle Ladungsträger (Löcher), meist keine LDD-Schritte
- ▶ Herstellung: Ionenimplantation, durchdringt Gateoxid

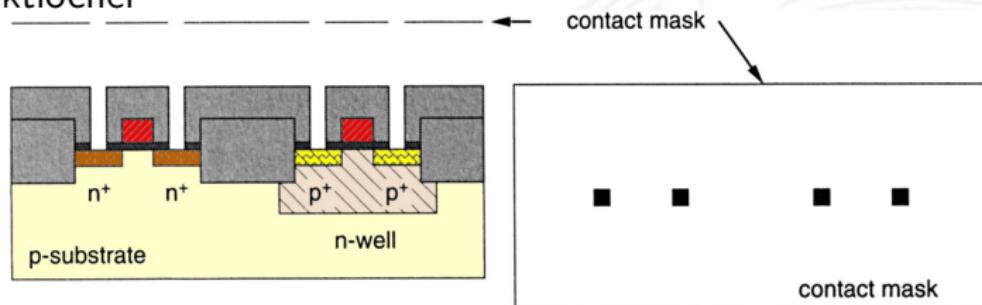


## CMOS Prozessschritte: Inverter (cont.)

### 12. $SiO_2$ aufbringen, Feldoxid

- ▶ Strukturen isolieren
- ▶ Herstellung: Epitaxie

### 13. Kontaktlöcher

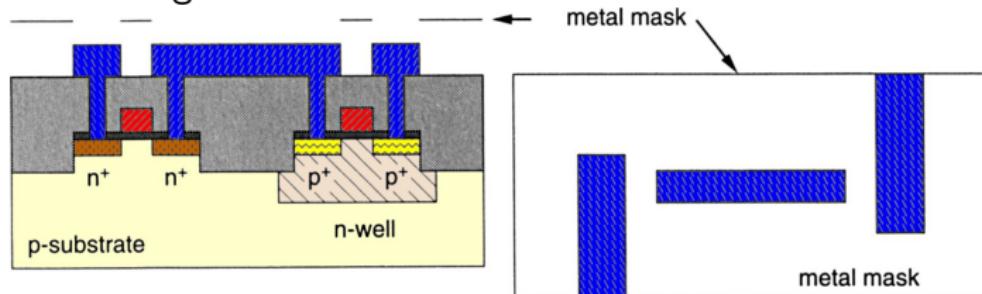


- ▶ Verbindet (spätere) Metallisierung mit Polysilizium oder Diffusion
- ▶ Anschlüsse der Transistoren: Gate, Source, Drain
- ▶ Herstellung: Ätzprozess



## CMOS Prozessschritte: Inverter (cont.)

### 14. Metallverbindung



- ▶ Erzeugt Anschlüsse im Bereich der Kontaktlöcher
- ▶ Herstellung: Metall aufdampfen, Ätzen nach Planarprozess

### 15. weitere Metalllagen

- ▶ Weitere Metallisierungen, bis zu 7 × Metall
- ▶ Schritte: 12. bis 14. wiederholen



## CMOS Prozessschritte: Inverter (cont.)

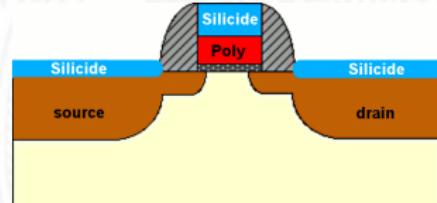
### 16. Passivierung

- ▶ Chipoberfläche abdecken, Plasmanitridschicht

### 17. Pad-Kontakte öffnen

Zahlreiche Erweiterungen für Submikron CMOS-Prozesse

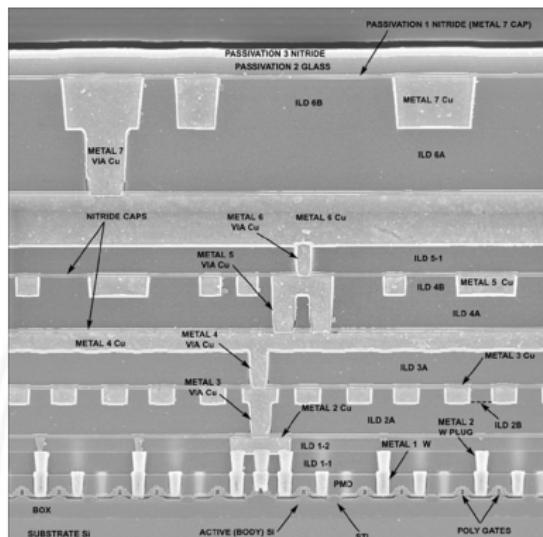
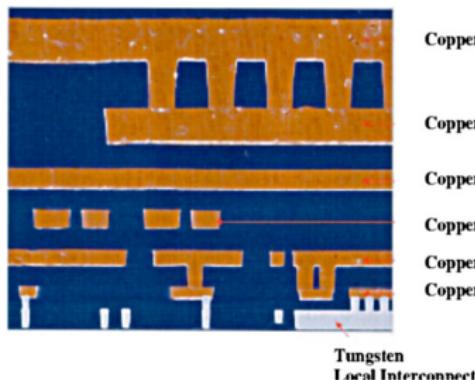
- ▶ „vergrabene“ Layer
  - ▶ verbessern elektrische Eigenschaften
  - ▶ Bipolar-Transistoren
  - ▶ Analog-Schaltungen
- ▶ Gate Spacer, seitlich  $SiO_2$
- ▶ Silizidoberflächen: verringern Kontaktwiderstand zu Metallisierung





# CMOS Prozessschritte: Inverter (cont.)

## ► Kupfer Metallisierung

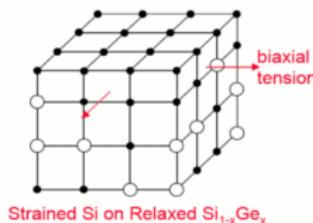


► high-k Dielektrika: Gate-Isolierung dicker, weniger Leckströme

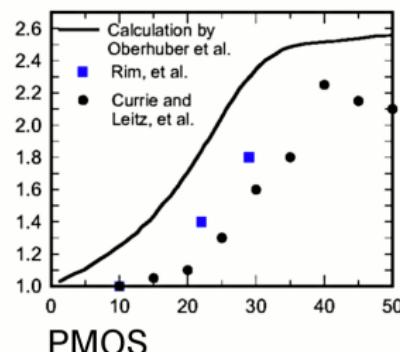
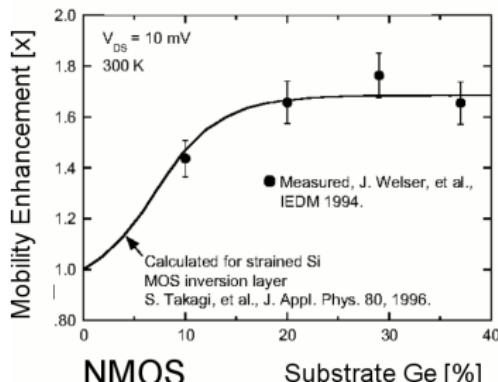
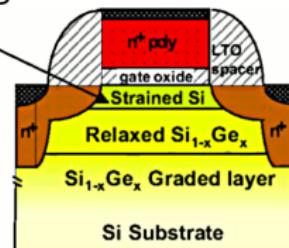


## CMOS Prozessschritte: Inverter (cont.)

- „gestrecktes“ Silizium: höhere Beweglichkeit



Strained Si channel  
with high mobility





# Programmierbare Logikbausteine

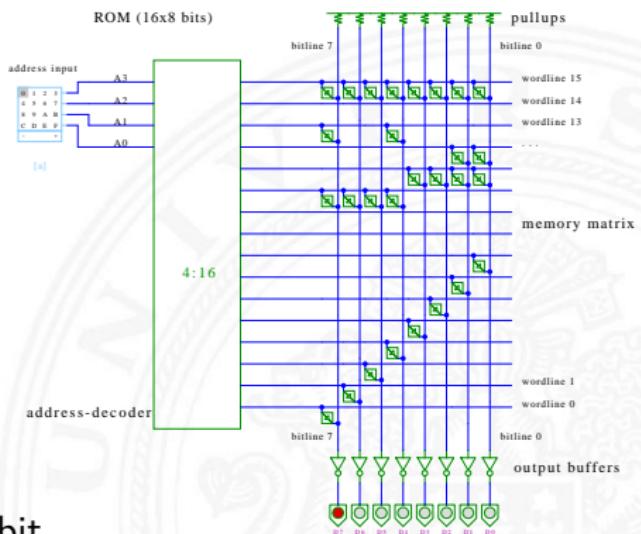
Kompromiss zwischen fest aufgebauter Hardware und Software-basierten Lösungen auf Computern

- ▶ Realisierung anwendungsspezifischer Funktionen und Systeme
  - ▶ gute bis sehr gute Performance
  - ▶ hoher Entwurfsaufwand
  - ▶ vom Anwender (evtl. mehrfach) programmierbar
- ▶ Klassifikation nach Struktur und Komplexität
  - ▶ PROM Programmable Read-Only Memory
  - ▶ PAL Programmable Array Logic
  - ▶ GAL Generic Array Logic
  - ▶ PLA Programmable Logic Array
  - ▶ CPLD Complex Programmable Logic Device
  - ▶ FPGA Field-Programmable Gate Array
  - ▶ ...



# PROM: Programmable Read-Only Memory

- ▶ UND-ODER Struktur
- ▶ UND-Array
  - ▶ fest
  - ▶ voll auskodiert:  $2^n$  Terme
- ▶ ODER-Terme
  - ▶ programmierbar
- ▶ auch: „LUT“ (look-up table)
- ▶ Hades Beispiel:  $n = 4$ ,  $16 \times 8$  bit





# PAL: Programmable Array Logic

- ▶ disjunktive Form: UND-ODER Struktur
- ▶ UND-Ausgänge fest an die ODER-Eingänge angeschlossen
- ▶ Eingänge direkt und invertiert in die UND-Terme geführt
- ▶ Verknüpfungen der Eingänge zu den UND-Termen programmierbar
  
- ▶ heute durch GAL ersetzt (s.u.)

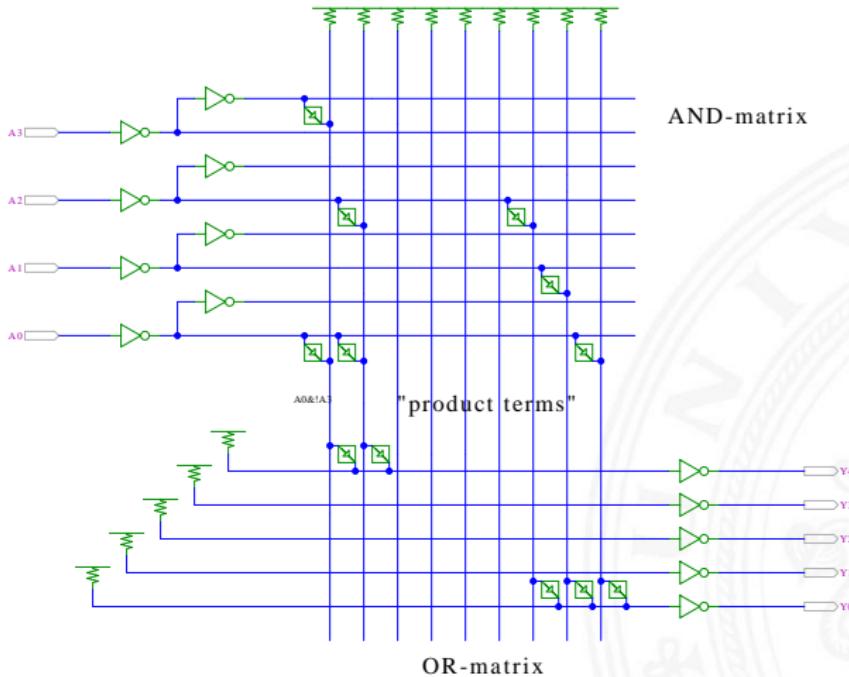


# PLA: Programmable Logic Array

- ▶ disjunktive Form: logische UND-ODER Struktur
  - ▶ Eingänge direkt und invertiert in die UND-Terme geführt
  - ▶ Verknüpfungen Eingänge UND-Terme
  - ▶ Verknüpfungen UND-Ausgänge zu ODER-Eingängen programmierbar
- 
- + in NMOS-Technologie sehr platzsparend realisierbar als NOR-NOR Matrix (de-Morgan Regel)
  - statischer Stromverbrauch
  - in CMOS-Technologie kaum noch verwendet



# PLA: Programmable Logic Array (cont.)



Hades Webdemos: 42-programmable/10-pla/pla.html

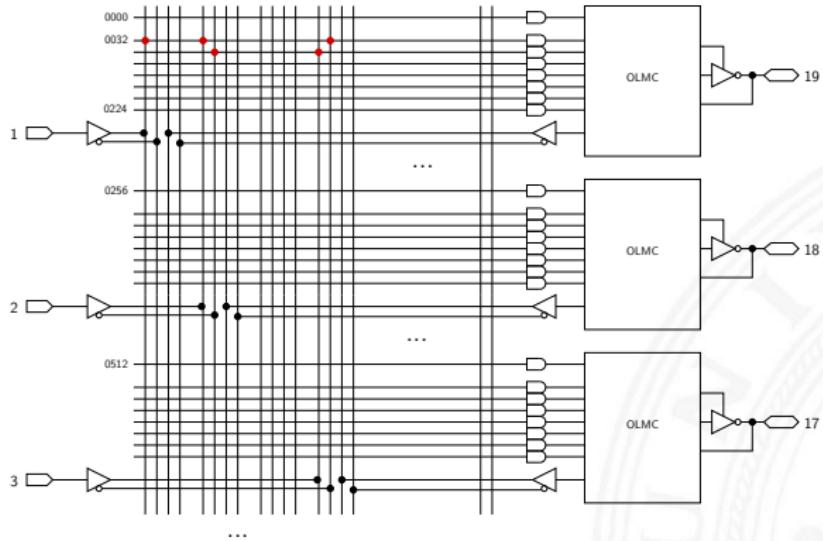


# GAL: Generic Array Logic

- ▶ disjunktive UND-ODER Struktur
- ▶ externe Eingänge und Ausgangswerte direkt/invertiert
- ▶ „Fuses“ verbinden Eingangswerte mit den AND-Termen
  
- ▶ programmierbare Ausgabezellen (OLMC)  
mit je einem D-Flipflop
- ▶ Output-Enable über AND-OR Matrix steuerbar
- ▶ drei Optionen
  - ▶ synchron/kombinatorisch (Flipflop nutzen oder umgehen)
  - ▶ Polarität des Eingangs ( $D$  oder  $\bar{D}$  speichern)
  - ▶ Polarität des Ausgangs ( $Q$  oder  $\bar{Q}$  ausgeben)
- ▶ Beispiel: GAL16V8 mit 8 Ausgabezellen, je 7+1 OR-Terme pro Ausgabezelle, 32 Eingänge pro Term



## GAL: Blockschaltbild (Ausschnitt)

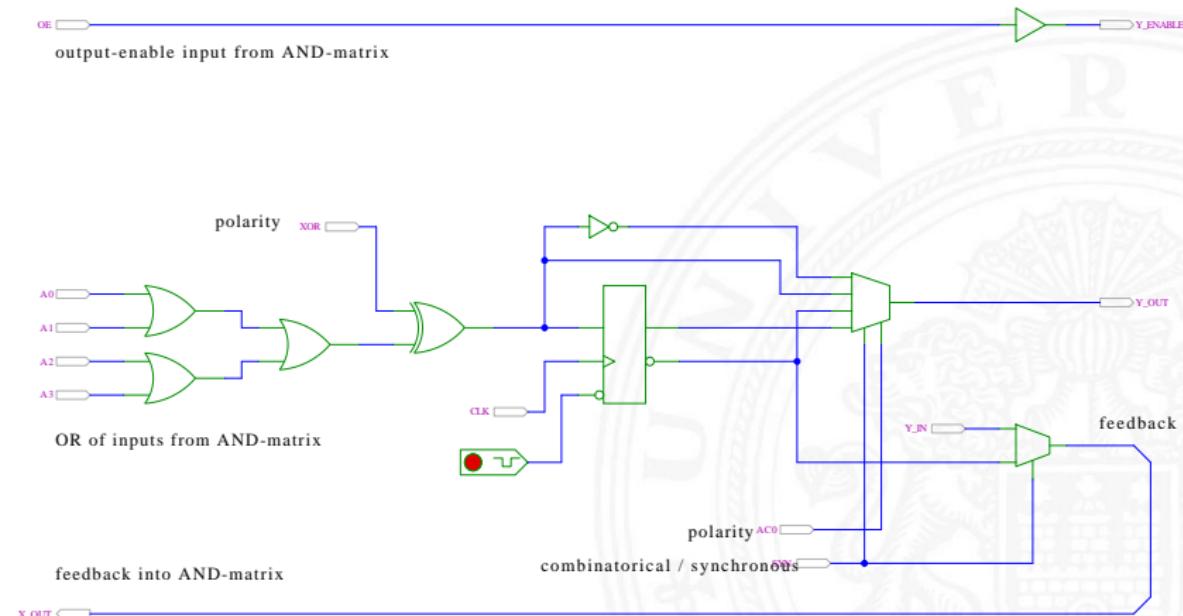


- ▶ programmierbare Sicherungen durchnummeriert
- ▶ kompakte Darstellung der UND-Terme: je eine Zeile
- ▶ Beispiel: zweiter Term (ab 0032)  $y = 1 \vee 2 \vee \bar{3}$



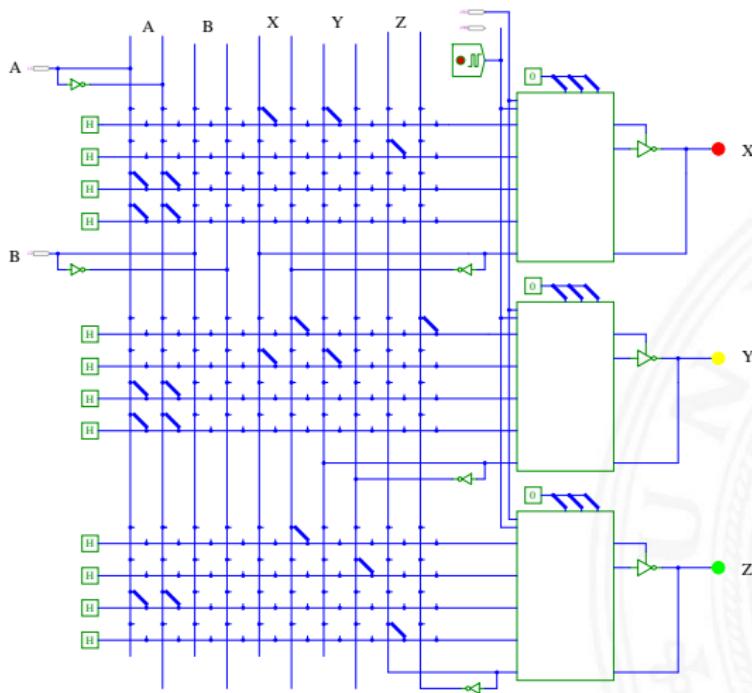
# GAL: Ausgabezelle mit Flipflop

## OLMC: Output-Logic-Macrocell





# GAL: Beispiel Ampel





# FPGA: Field-Programmable Gate-Array

Sammelbegriff für „große“ anwenderprogrammierbare Schaltungen

- ▶ Matrix von kleineren programmierbaren Zellen, beispielsweise
  - ▶ SRAM als Lookup für Funktionen
  - ▶ programmierbare Register
  - ▶ carry-lookahead Logik
- ▶ Multiplexer-Netzwerk als programmierbare Verbindung
- ▶ zusätzliche „Makrozellen“
  - ▶ Multiplizierer
  - ▶ eingebettete Prozessorkerne
- ▶ IO-Zellen
  - ▶ schnelle serielle Kommunikation
  - ▶ PLLs (programmierbare Taktgeneratoren)



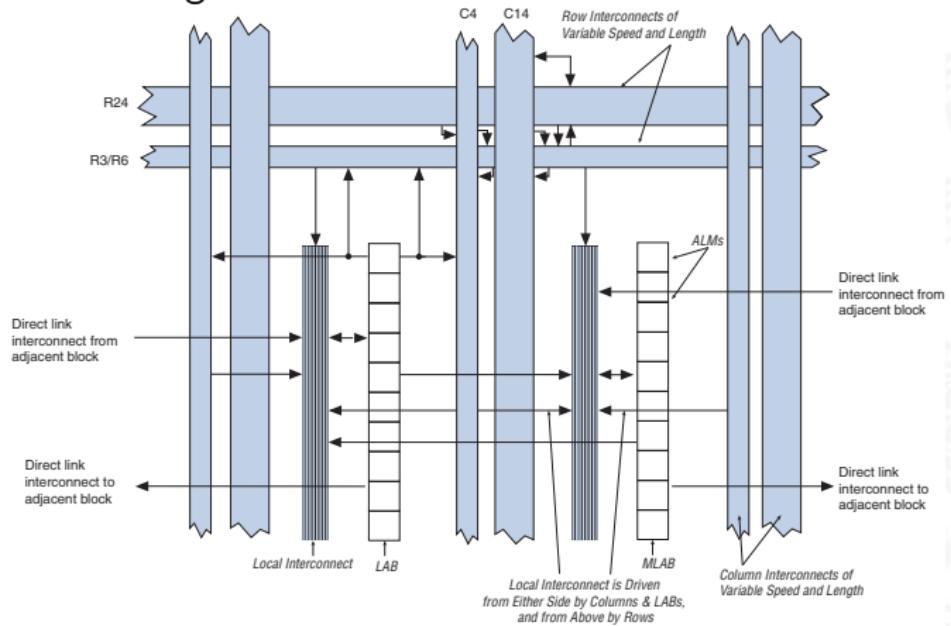
## FPGA: Field-Programmable Gate-Array (cont.)

- ▶ generierte Komponenten: ROM, RAM, FIFO...
- ▶ vorgefertigte IP-Blöcke („*Intellectual Property*“)
  - ▶ Netzwerkprotokolle
  - ▶ Speichercontroller
  - ▶ Bussysteme
  - ▶ ...
- ▶ Komplexität
  - ▶  $\approx 1\,200$  nutzbare I/O
  - ▶  $\approx 15$  Mio. Gatteräquivalente (2 input NOR)
  - ▶  $\approx 1$  GHz
- ▶ Xilinx, Altera, weitere Hersteller



# FPGA: Beispiel Altera

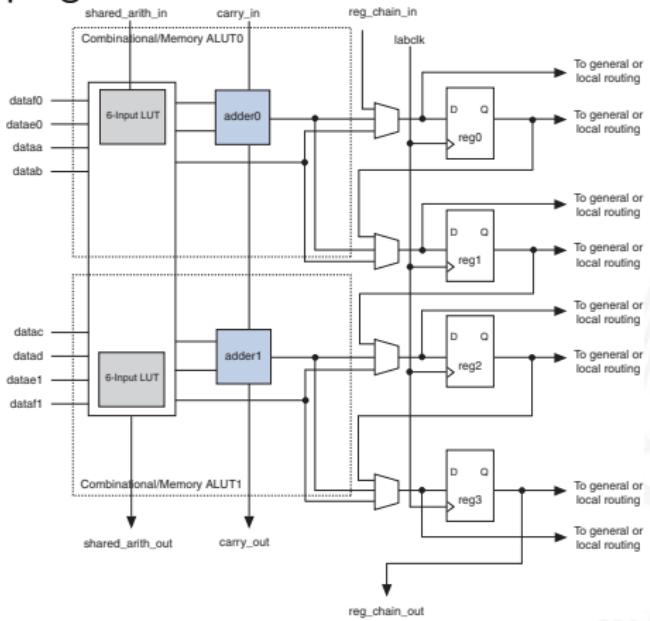
## Verbindungsnetzwerk





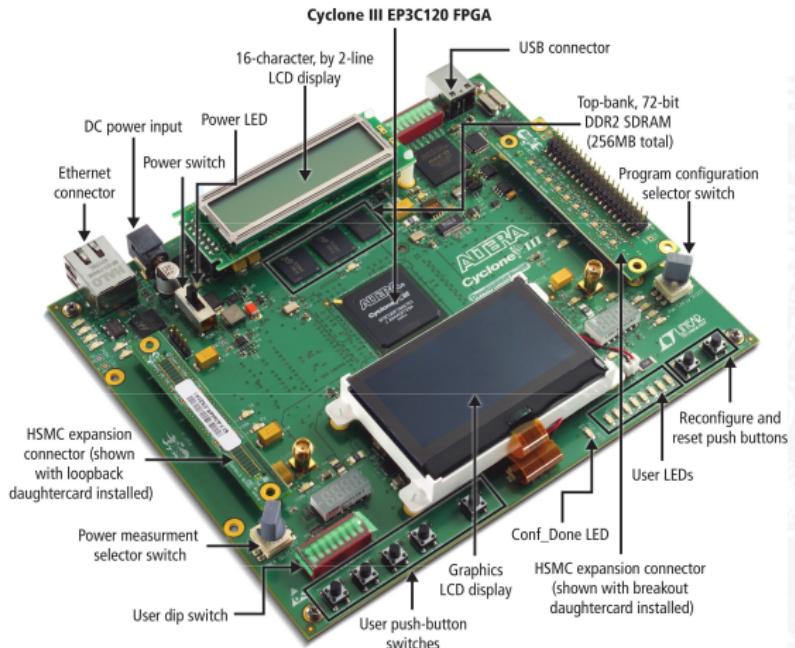
# FPGA: Beispiel Altera (cont.)

## programmierbarer Block





# FPGA: Beispiel Altera (cont.)



Prototypenplatine



# Entwurf Integrierter Schaltungen

besonders anspruchsvoller Bereich der Informatik

- ▶ Halbleiterfertigung benötigt vorab sämtliche Geometriedaten
- ▶ spätere Änderungen eines Chips nicht möglich
- ▶ Durchlauf aller Fertigungsschritte dauert Wochen bis Monate
- ▶ Entwürfe müssen komplett fehlerfrei sein
  
- ▶ spezielle Hardware-/System-Beschreibungssprachen
- ▶ Simulation des Gesamtsystems
- ▶ Analyse des Zeitverhaltens
- ▶ ggf. Emulation/Prototyping mit FPGAs
  
- ▶ Kombination von Hardware- oder Softwarerealisierung von Teilfunktionen, sog. **HW/SW-Codesign**



# Entwurfsablauf

## Wasserfallmodell

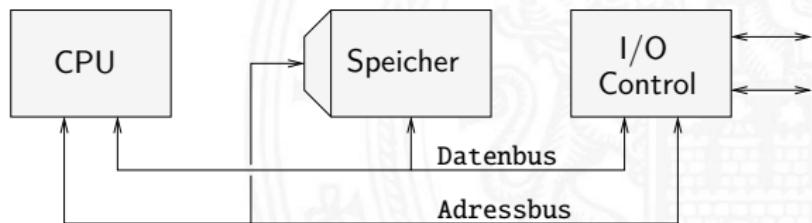
- ▶ Lastenheft
- ▶ Verhaltensmodell (Software)
- ▶ Aufteilung in HW- und SW-Komponenten
- ▶ funktionale Simulation/Emulation und Test
- ▶ Synthese oder manueller Entwurf der HW, Floorplan
- ▶ Generieren der „Netzliste“ (logische Struktur)
- ▶ Simulation mit Überprüfung der Gatter-/Leitungslaufzeiten
- ▶ Generieren und Optimierung des Layouts („Tapeout“)



# Abstraktion im VLSI-Entwurf

## Abstraktionsebenen

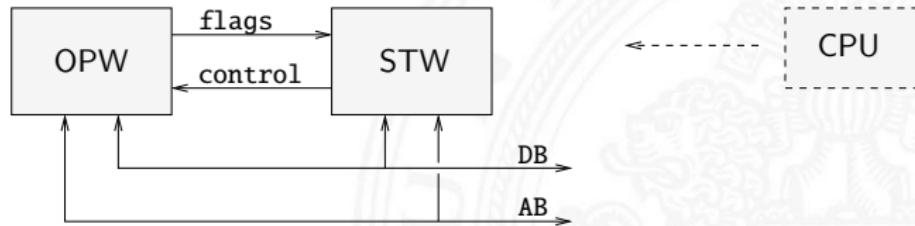
- keine einheitliche Bezeichnung in der Literatur
- ▶ Architekturebene
  - ▶ Funktion/Verhalten Leistungsanforderungen
  - ▶ Struktur Netzwerk
    - aus Prozessoren, Speicher, Busse, Controller...
  - ▶ Nachrichten Programme, Prokolle
  - ▶ Geometrie Systempartitionierung





## Abstraktion im VLSI-Entwurf (cont.)

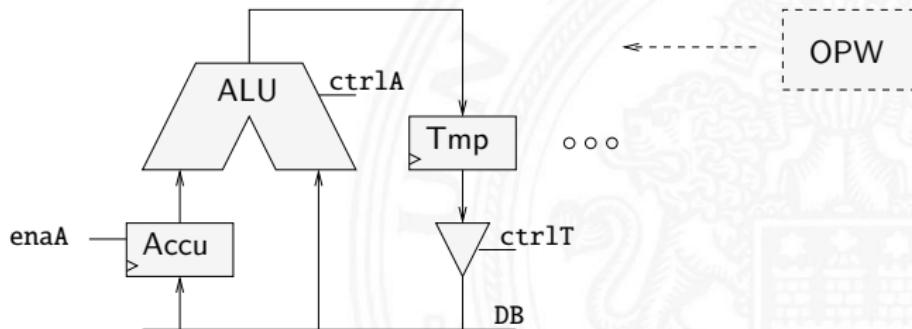
- ▶ Hauptblockebene (Algorithmenebene, funktionale Ebene)
  - ▶ Funktion/Verhalten Algorithmen, formale Funktionsmodelle
  - ▶ Struktur Blockschaltbild aus Hardwaremodule, Busse. . .
  - ▶ Nachrichten Prokolle
  - ▶ Geometrie Cluster





## Abstraktion im VLSI-Entwurf (cont.)

- ▶ Register-Transfer Ebene
  - ▶ Funktion/Verhalten Daten- und Kontrollfluss, Automaten...
  - ▶ Struktur RT-Diagramm
  - aus Register, Multiplexer, ALUs...
  - ▶ Nachrichten Zahlencodierungen, Binärworte...
  - ▶ Geometrie Floorplan





## Abstraktion im VLSI-Entwurf (cont.)

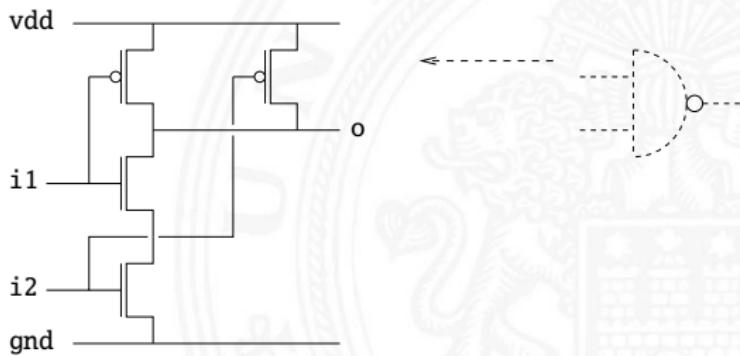
- ▶ Logikebene (Schaltwerkebene)
  - ▶ Funktion/Verhalten Boole'sche Gleichungen
  - ▶ Struktur Gatternetzliste, Schematic
    - aus Gatter, Flipflops, Latches...
  - ▶ Nachrichten Bit
  - ▶ Geometrie Moduln





## Abstraktion im VLSI-Entwurf (cont.)

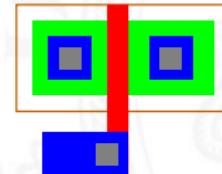
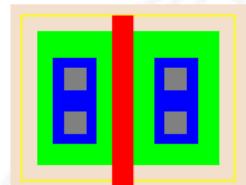
- ▶ elektrische Ebene (Schaltkreisebene)
  - ▶ Funktion/Verhalten Differentialgleichungen
  - ▶ Struktur elektrisches Schaltbild
  - ▶ Nachrichten Transistoren, Kondensatoren...
  - ▶ Geometrie Ströme, Spannungen
  - ▶ Geometrie Polygone, Layout → physikalische Ebene





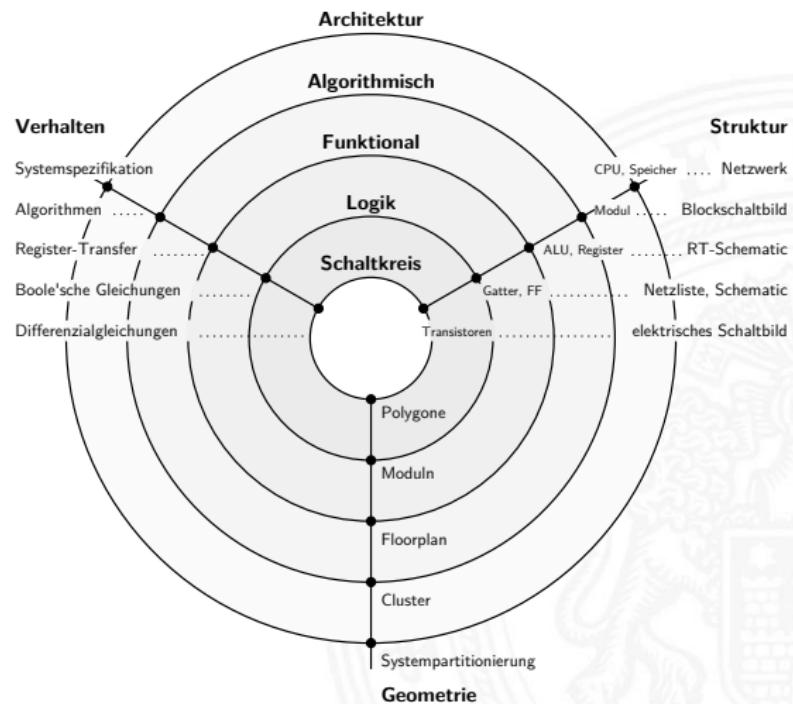
## Abstraktion im VLSI-Entwurf (cont.)

- ▶ physikalische Ebene (geometrische Ebene)
  - ▶ Funktion/Verhalten partielle DGL
  - ▶ Struktur Dotierungsprofile





# Y-Diagramm





## Y-Diagramm (cont.)

drei unterschiedliche Aspekte/Dimensionen:

- 1 Verhalten
- 2 Struktur (logisch)
- 3 Geometrie (physikalisch)

- ▶ Start möglichst abstrakt, z.B. als Verhaltensbeschreibung
- ▶ Ende des Entwurfsprozesses ist vollständige IC Geometrie für die Halbleiterfertigung (Planarprozess)
- ▶ Entwurfsprogramme („EDA“, *Electronic Design Automation*) unterstützen den Entwerfer: setzen Verhalten in Struktur und Struktur in Geometrien um



# Entwurfsstile

Was ist die „beste“ Realisierung einer gewünschten Funktionalität?

- ▶ mehrere konkurrierende Kriterien
  - ▶ Performance, Chipfläche, Stromverbrauch
  - ▶ Stückkosten vs. Entwurfsaufwand und Entwurfskosten
  - ▶ Zeitbedarf bis zur ersten Auslieferung und ggf. für Designänderungen
  - ▶ Schutz von Intellectual-Property
  - ▶ ...
- ▶ vier gängige Varianten
  - ▶ Full-custom Schaltungen
  - ▶ Semi-custom Bausteine: Standardzellen, Gate-Arrays
  - ▶ Anwenderprogrammierbare Bausteine: FPGA, PAL/GAL, ROM
  - ▶ Software auf von-Neumann Rechner: RAM, ROM

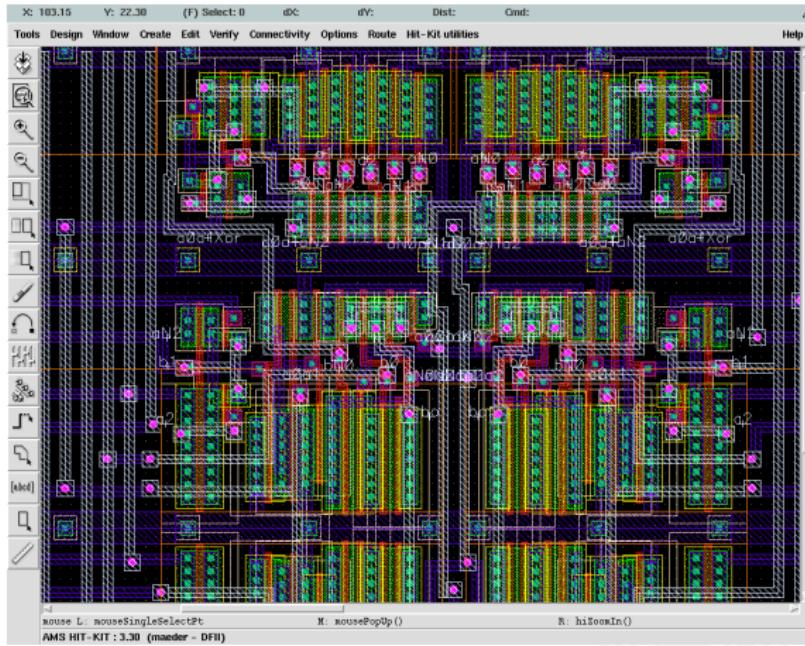


## Full-custom / „Vollkunden-Entwurf“

- ▶ vollständiger Entwurf der gesamten Geometrie eines Chips
  - ▶ jeder Transistor einzeln „massgeschneidert“ und platziert
  - ▶ vorgegeben sind lediglich die Entwurfsregeln (*design-rules*) des Herstellungsprozesses (Strukturbreite, Mindestabstände, usw.)
  - ▶ oft Verwendung von Teilschaltungen/Makros des Herstellers
- 
- ▶ minimale Fläche, beste Performance, kleinster Stromverbrauch
  - ▶ geringste Stückkosten bei der Produktion
  - ▶ aber höchste Entwurfs- und Maskenkosten
  - ▶ erste Prototypen erst nach Durchlaufen aller Maskenschritte
- 
- ▶ nur bei Massenprodukten wirtschaftlich  $\gg 100\,000$  Stück  
z.B. Speicherbausteine (SRAM, DRAM), gängige Prozessoren



# Full-custom / „Vollkunden-Entwurf“ (cont.)





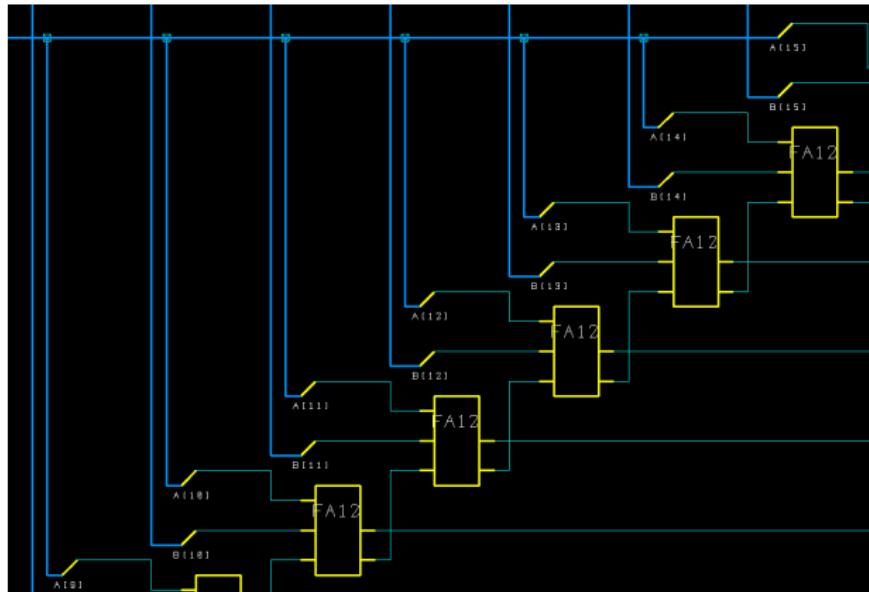
## Semi-custom: Standardzell-Entwurf

- ▶ Entwurf der Schaltung mit vorhandenen Grundkomponenten
  - ▶ Basisbibliothek mit Gattern und Flipflops
  - ▶ teilweise (konfigurierbare) ALUs, Multiplizierer
  - ▶ Generatoren für Speicher
- ▶ Entwurfsregeln sind der Bibliothek berücksichtigt
- ▶ Platzierung der Komponenten und Verdrahtung
  
- ▶ kleine Chipfläche, gute Performance, niedriger Stromverbrauch
- ▶ geringe Stückkosten
- ▶ hohe Maskenkosten (alle Masken erforderlich)
- ▶ erste Prototypen erst nach Durchlaufen aller Maskenschritte
  
- ▶ nur bei größeren Stückzahlen wirtschaftlich  $\gg 10\,000$  Stück

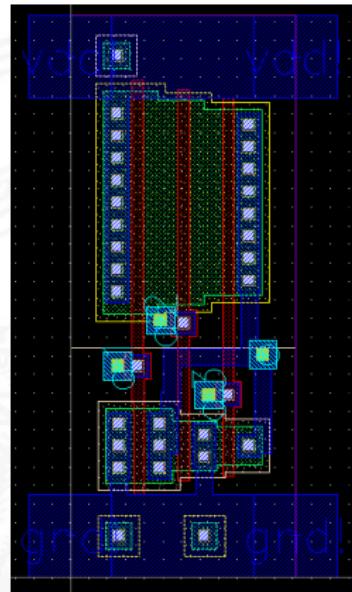


# Semi-custom: Standardzell-Entwurf (cont.)

Schematic



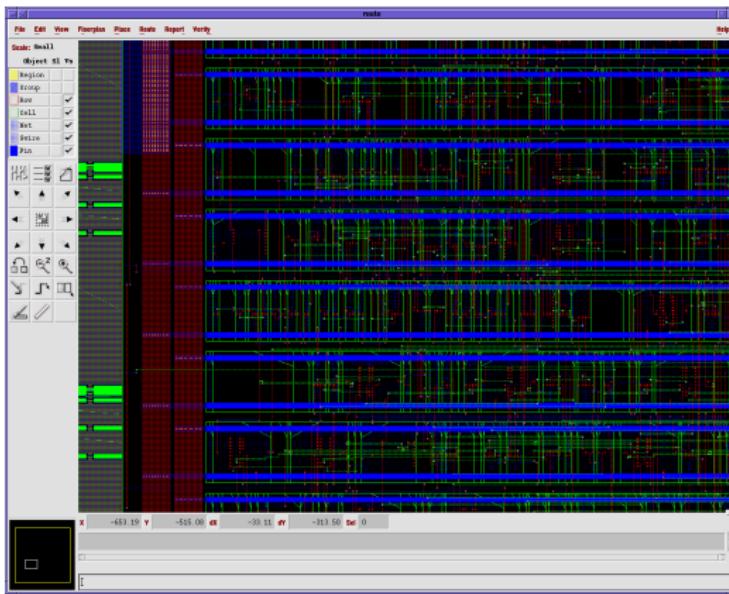
Zell-Layout





# Semi-custom: Standardzell-Entwurf (cont.)

## Standardzell Layout





## Semi-custom: Gate-Arrays

- ▶ Schaltung mit Gattern/Transistoren an festen Positionen
  - ▶ Entwurf durch Verdrahten der vorhandenen Transistoren
  - ▶ überzählige Transistoren werden nicht angeschlossen
- 
- ▶ mittlere Chipfläche, Performance und Stromverbrauch
  - ▶ mittlere Stückkosten
  - ▶ mittlere Maskenkosten (nur Verdrahtung kundenspezifisch)
  - ▶ Prototypen schnell verfügbar (nur Verdrahtung)
- 
- ▶ ab mittleren Stückzahlen wirtschaftlich  $> 1\,000$  Stück
  - ▶ werden von großen FPGAs verdrängt



# FPGA: Field-Programmable Gate-Arrays

- ▶ Hunderte/Tausende von konfigurierbaren Funktionsblöcken
- ▶ Verschaltung dieser Blöcke vom Anwender programmierbar
- ▶ Entwurfsprogramme setzen Beschreibung des Anwenders auf die Hardware-Blöcke und deren Verschaltung um
- ▶ derzeit bis ca. 15 Mio. Gatter-Äquivalente möglich
- ▶ Taktfrequenzen bis max. GHz, typisch 100 MHz-Bereich
- ▶ zwei dominierende Hersteller: Xilinx, Altera
  
- ▶ nicht benutzte Blöcke liegen brach
- ▶ Schaltung kann in Minuten neu programmiert/verbessert werden
  
- ▶ optimal für geringe Stückzahlen, ca. 10...1 000 Stück



# FPGA selbstgemacht: Projekt 64-189

Ideen für einen Mikrochip? Zum Beispiel für Bildverarbeitung, 3D-Algorithmen, Parallelverarbeitung, usw.

- ▶ Hereinschnuppern: **Projekt 64-189 *Entwurf eines Mikrorechners***
- ▶ eigenen Prozessor mit Befehlssatz etc. entwerfen und auf FPGA Prototypenplatine realisieren
  
- ▶ Demo-Boards von Altera und Xilinx und Entwurfsssoftware sind bei uns am Fachbereich verfügbar
- ⇒ einfach bei TAMS oder TIS vorbeischauen



# Literatur: Quellen für die Abbildungen

- ▶ Andreas Mäder,  
*Vorlesung: Rechnerarchitektur und Mikrosystemtechnik*,  
Universität Hamburg, FB Informatik, 2010  
[tams.informatik.uni-hamburg.de/lectures/2010ws/vorlesung/ram](http://tams.informatik.uni-hamburg.de/lectures/2010ws/vorlesung/ram)
- ▶ Norbert Reifschneider,  
*CAE-gestützte IC-Entwurfsmethoden*,  
Prentice Hall, 1998
- ▶ Neil H. E. Weste, Kamran Eshragian,  
*Principles of CMOS VLSI Design — A Systems Perspective*,  
Addison-Wesley Publishing, 1994



## Literatur: Vertiefung

- ▶ Reiner Hartenstein,  
*Standort Deutschland: Wozu noch Mikro-Chips*,  
IT-Press Verlag, 1994 (vergriffen)
- ▶ Gabriela Nicolescu, Pieter J. Mosterman,  
*Model-Based Design for Embedded Systems*, CRC Press, 2010
- ▶ Carver Mead, Lynn Conway,  
*Introduction to VLSI Systems*, Addison-Wesley, 1980
- ▶ Giovanni de Micheli,  
*Synthesis and Optimization of Digital Circuits*,  
McGraw-Hill, 1994
- ▶ Ulrich Tietze, Christoph Schenk,  
*Halbleiter-Schaltungstechnik*, Springer-Verlag, 2009