

# 64-210 Eingebettete Systeme

–VLSI-Entwurf / Methoden und Werkzeuge–

[http://tams.informatik.uni-hamburg.de/  
lectures/2013ss/vorlesung/es](http://tams.informatik.uni-hamburg.de/lectures/2013ss/vorlesung/es)

Andreas Mäder



Universität Hamburg  
Fakultät für Mathematik, Informatik und Naturwissenschaften  
Fachbereich Informatik

**Technische Aspekte Multimodaler Systeme**

Sommersemester 2013

Die folgenden Folien sind ein Auszug aus den Unterlagen der Vorlesung **64-613 Rechnerarchitekturen und Mikrosystemtechnik** vom Wintersemester 2011/2012.

Das komplette Material findet sich auf den Web-Seiten unter <http://tams.informatik.uni-hamburg.de/lectures/2011ws/vorlesung/ram>

# Gliederung

## 1. Entwurfsmethodik

Motivation

Abstraktion im VLSI-Entwurf

Vorgehensweise

## 2. EDA-Werkzeuge

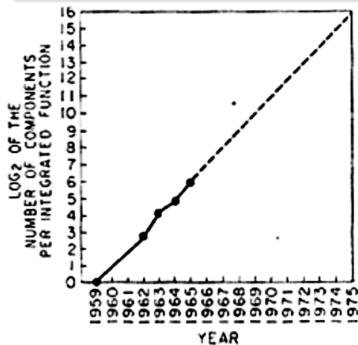
## 3. Entwurfsstile



# Motivation

## Moore's Law

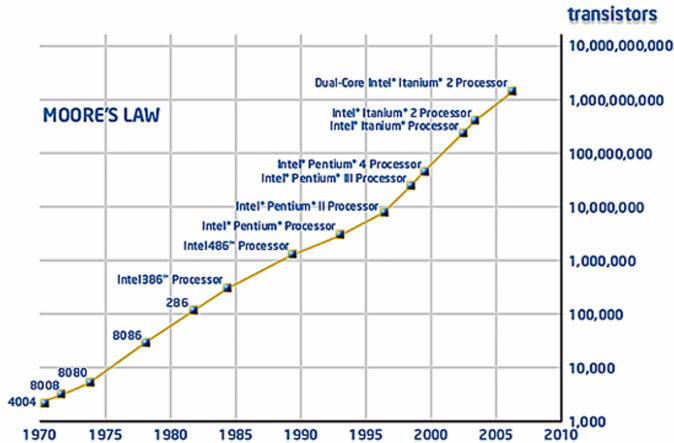
Die Zahl der Transistoren pro IC verdoppelt sich alle 2 Jahre



Gordon Moore 1965:

„Cramming more components onto integrated circuits“

# Motivation (cont.)



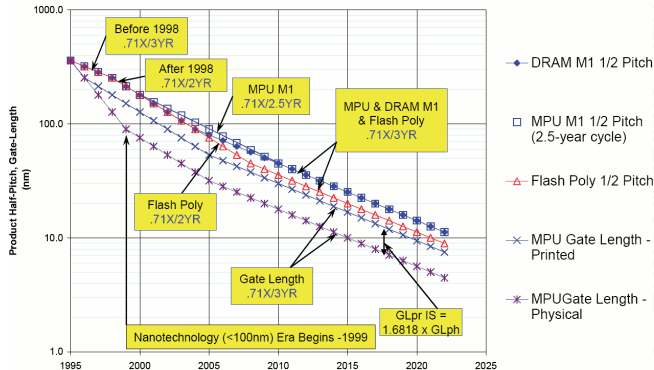
Intel

# Motivation (cont.)

## 1. Technologie

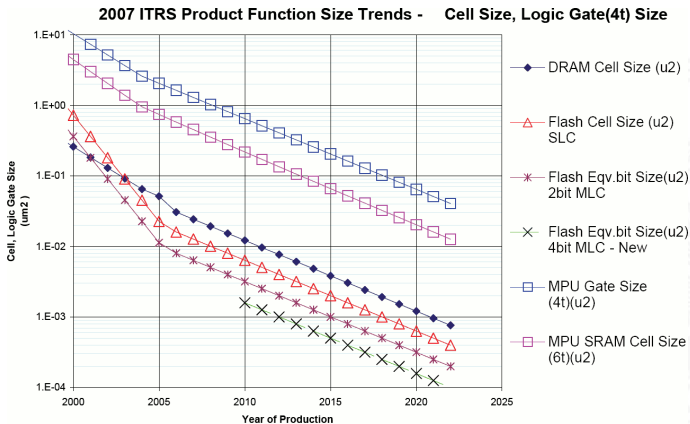
- ▶ Verkleinerung der Strukturbreite
- ▶ Höhere Integrationsdichte

2007 ITRS Product Technology Trends - Half-Pitch, Gate-Length



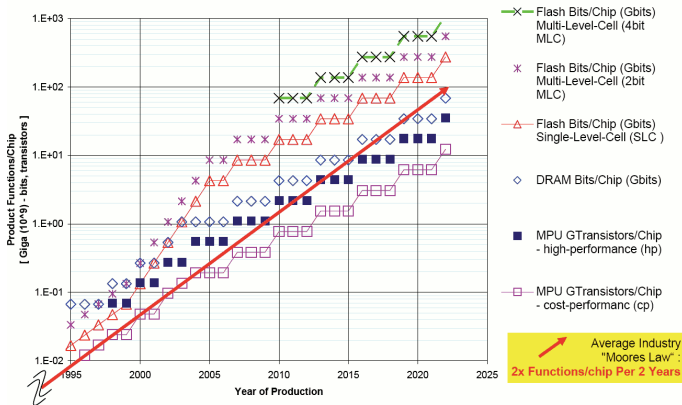
[ITRS07]

# Motivation (cont.)



# Motivation (cont.)

2007 ITRS Product Technology Trends - Functions per Chip



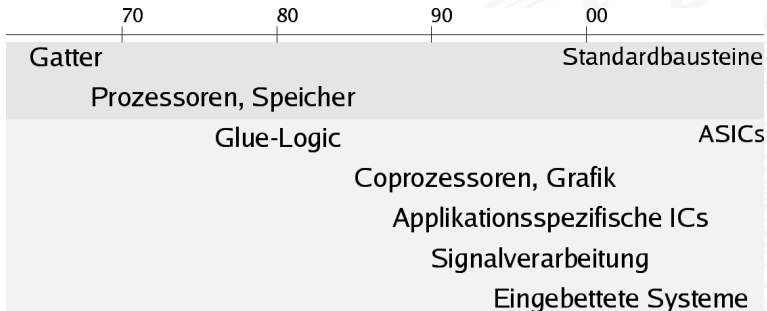
[ITRS07]



# Motivation (cont.)

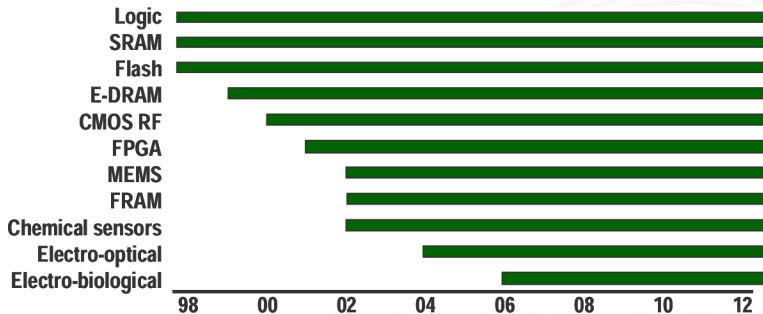
## 2. Applikationen

- ▶ von Standardbausteinen zu ASICs und Systemen
- ▶ Digitale Anwendungen



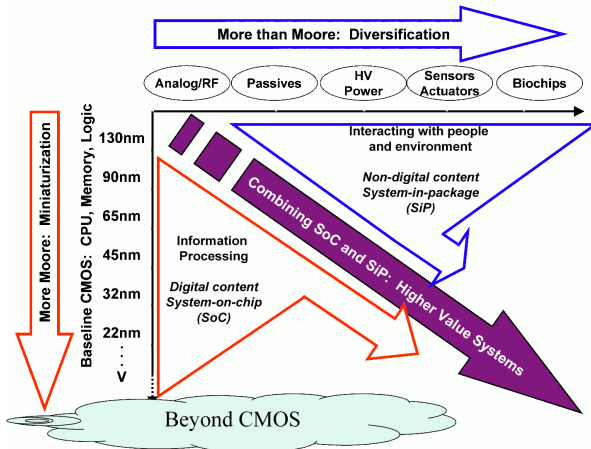
## Motivation (cont.)

- ▶ + Analoge Komponenten
- ▶ + Mikrosysteme



# Motivation (cont.)

## Übergang zu Systemen



[ITRS07]

# Motivation (cont.)

neue Anwendungsfelder

- ▶ „Computing“
- ▶ „Consumer Products“
- ▶ „Automotive“
- ▶ „Telecommunication“
- ▶ „mobile Applications“

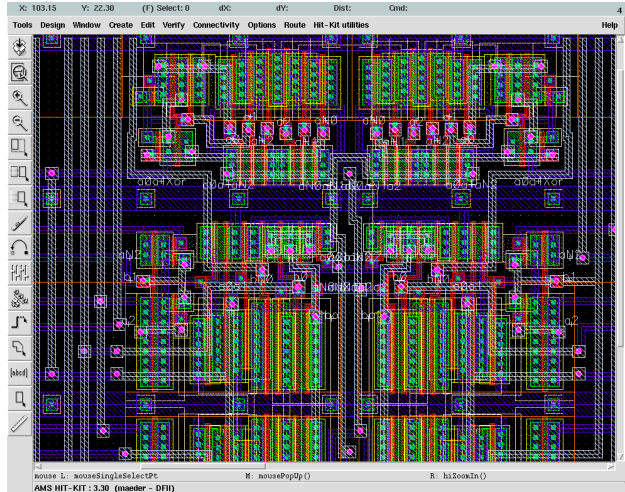
## 3. Methoden und Werkzeuge im Chipentwurf

- ▶ enges Zusammenwirken mit der technischen Entwicklung und den Anforderungen durch die Applikationen



## Wie wird Entworfen?

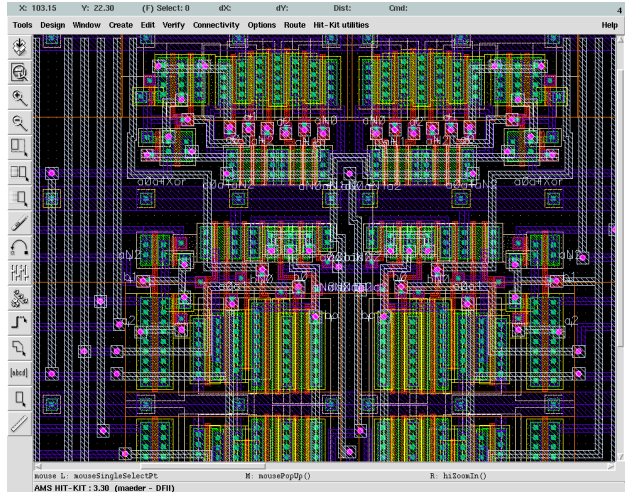
# Hardwareentwurf



# Wie wird Entworfen?

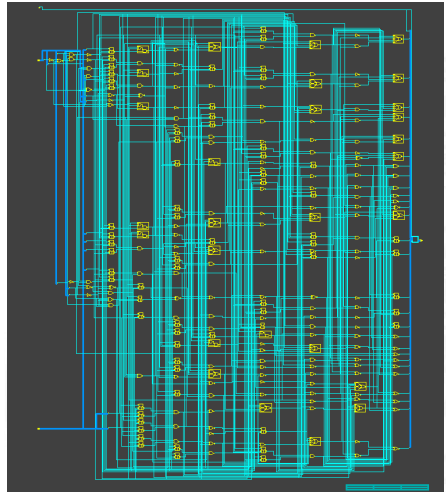
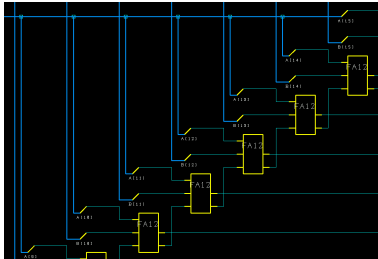
## Hardwareentwurf

...so nicht  
meistens



# Wie wird Entworfen?

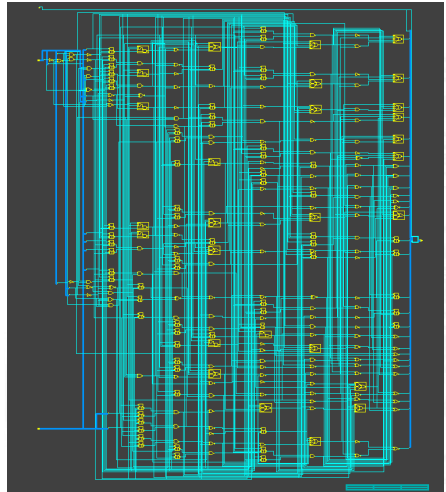
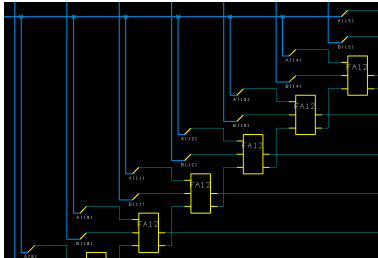
## Hardwareentwurf



# Wie wird Entworfen?

Hardwareentwurf

...so auch nicht

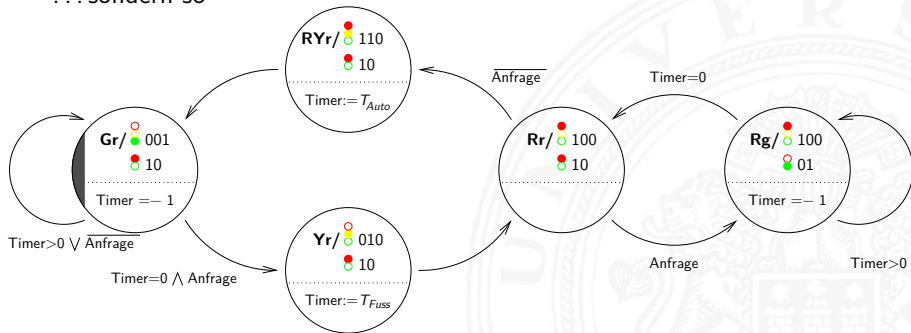




# Wie wird Entworfen?

## Hardwareentwurf

... sondern so



# Wie wird Entworfen? (cont.)

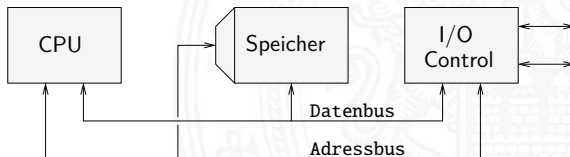
```

mainP: process (clk, rst) is
  type stateTy is (Gr, Yr, Rr, Rg, RYr);
  variable      timer      : integer range 0 to maxWalkC;
  variable      state      : stateTy;
  variable      request    : boolean;
begin
  if rst = '0' then
    liCar      <= "001";      liWalk <= "10";
    state      := Gr;
    timer      := 0;
    request    := false;
  elsif rising_edge(clk) then
    case state is
      when Gr =>
        liCar      <= "001";      liWalk <= "10";
        if (reqWalk = '1') then request := true;
        end if;
        if (timer > 0) then timer := timer - 1;
        elsif request then state := Yr;
        end if;
      when Yr =>
        liCar      <= "010";      liWalk <= "10";
        timer      := maxWalkC-1;
        state      := Rr;
      when Rr =>
    ...
  
```

# Abstraktion im VLSI-Entwurf

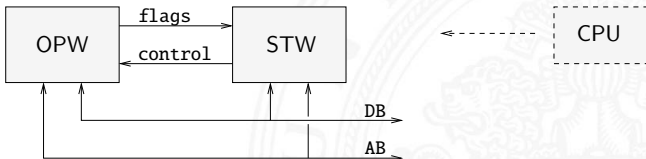
## Abstraktionsebenen

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



## Abstraktion im VLSI-Entwurf (cont.)

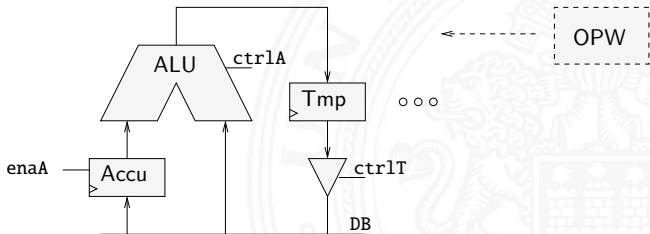
- ▶ Hauptblockebene (Algorithmenebene, funktionale Ebene)
  - ▶ Funktion/Verhalten Algorithmen, formale Funktionsmodelle
  - ▶ Struktur
    - aus Hardwaremodule, Busse...
  - ▶ Nachrichten Protokolle
  - ▶ Geometrie Cluster



# 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



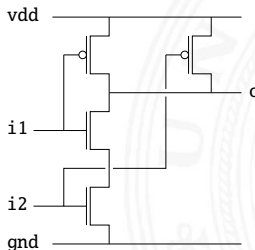
## Abstraktion im VLSI-Entwurf (cont.)

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



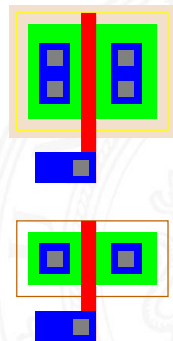
## Abstraktion im VLSI-Entwurf (cont.)

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



## Abstraktion im VLSI-Entwurf (cont.)

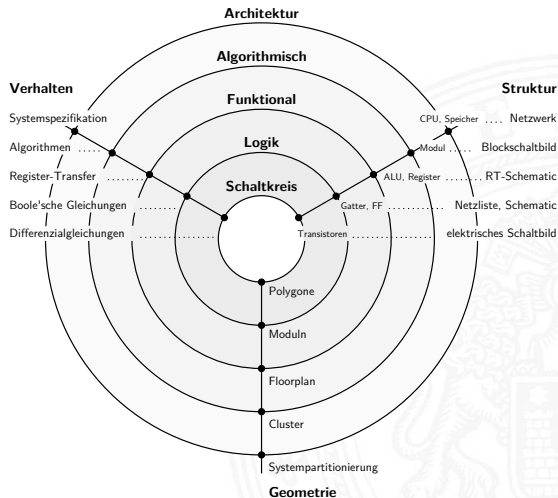
- ▶ physikalische Ebene (geometrische Ebene)
  - ▶ Funktion/Verhalten    partielle DGL
  - ▶ Struktur                Dotierungsprofile





# Abstraktion im VLSI-Entwurf (cont.)

## Y-Diagramm



D. Gajski, R. Kuhn 1983:  
„New VLSI Tools“

# Abstraktion im VLSI-Entwurf (cont.)

## Y-Diagramm / Gajski-Diagramm

- ▶ Visualisiert Abstraktionsebenen
- ▶ Sichtweisen
  - ▶ Funktion / Verhalten
  - ▶ Struktur
  - ▶ Geometrie (historisch, inzwischen überholt)

# Entwurfsvorgehen

- ▶ Unterscheidung von *Struktur* und *Verhalten*
  - ▶ Auf jeder Abstraktionsebene gibt es *elementare Einheiten* mit definiertem Verhalten
  - ▶ Entwurfsaufgabe
    - ▶ ein gegebenes Verhalten in eine Strukturbeschreibung (aus elementaren Einheiten) der jeweiligen Ebene umzusetzen
    - ▶ jede dieser Einheiten ist ihrerseits in der nächst niedrigeren Abstraktionsebene entsprechend zu realisieren
- ⇒ hierarchischer Entwurf, top-down

## Entwurfsvorgehen (cont.)

- ⇒ top-down: typisches Entwurfsvorgehen
- ⇒ bottom-up: Einflüsse auf höhere Abstraktionsebenen
  - ▶ Zeitverhalten
  - ▶ Schaltungstechniken
  - ▶ Arithmetiken
  - ▶ ...
- ▶ Zentrale Bedeutung der Simulation, bzw. der Verifikation
- ▶ Entwurf als iterativer Prozess
  - ▶ Alternativen: „exploring the design-space“
  - ▶ Versionen
  - ▶ Teamarbeit

# Gliederung

1. Entwurfsmethodik
2. EDA-Werkzeuge
  - Hierarchischer Entwurf
  - Werkzeuge
  - Probleme
3. Entwurstile

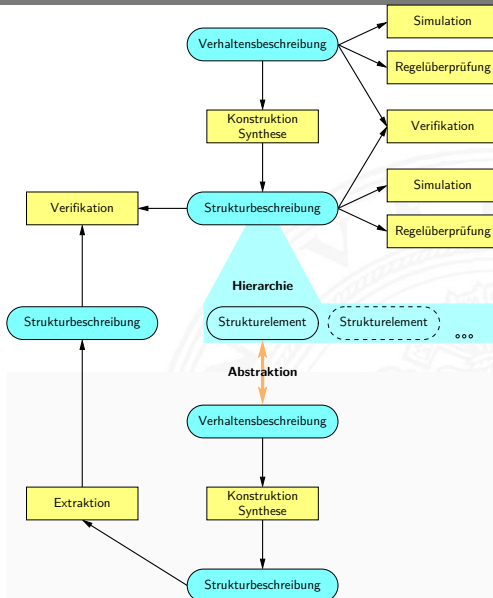


# Hierarchischer Entwurf

Nur durch neue Methoden und Werkzeuge konnte die Produktivität beim Chipentwurf während der letzten Jahre mit Moore's Law mithalten

- ▶ Änderungen in der Entwurfsmethodik
  - Struktur  $\Rightarrow$  Verhalten
  - grafische Eingabe  $\Rightarrow$  Hardwarebeschreibungssprachen
- ▶ Entwurf auf höheren Abstraktionsebenen
- ▶ Automatische Transformationen bis zum Layout
  - ▶ Synthese: Register-Transfer, High-Level
  - ▶ Datenpfad-/Makrozellgenerierung
  - ▶ Zellsynthese
  - ▶ Platzierung & Verdrahtung

# Entwurfswerkzeuge

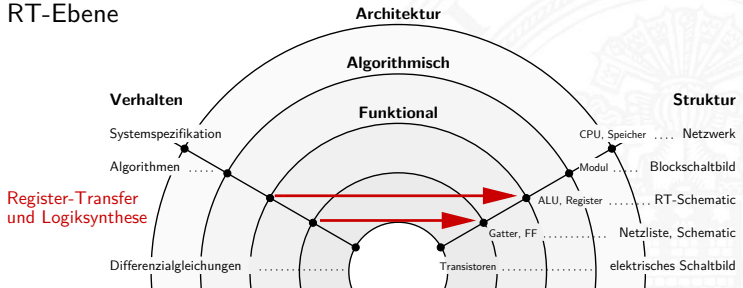


# Entwurfswerkzeuge

## ► Synthese

= automatische Generierung von Strukturbeschreibungen aus Verhaltensmodellen

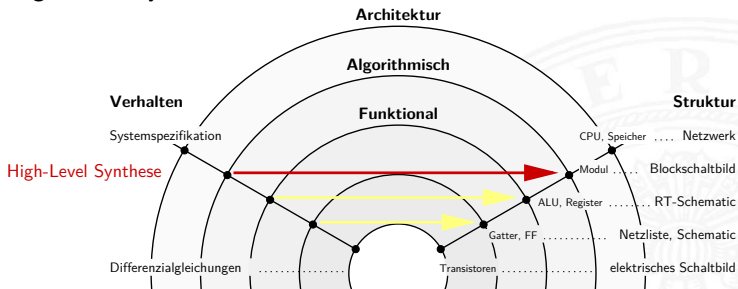
- Trend: IP-Komponenten (**I**ntellectual **P**roperty) und „behavioral Code“
- RT-Ebene





# Entwurfswerkzeuge (cont.)

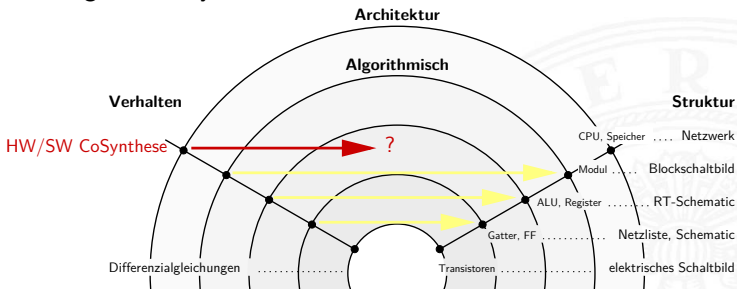
## ► High-Level Synthese



- Einschränkung des „Suchraums“
- spezielle Zielarchitekturen
- spezielle Anwendungsfelder
- Datenflussdominiert      DSPs
- Kontrollflussdominiert    Prozessoren

# Entwurfswerkzeuge (cont.)

## ► CoDesign → CoSynthese



- Partitionierung Hardware / Software ?
- nur manuell möglich

# Entwurfswerkzeuge (cont.)

## ► Simulation

- Trend: wachsender Aufwand, Systemsimulation
- Problem der Simulationsauswertung  $\Rightarrow$  auch dort Abstraktion
  - Programmiersprachen-Schnittstellen (VHPI, Verilog-PLI...)
  - Beispiele: ► Signalverarbeitung ► Bildverarbeitung
- Hardwarebeschleunigung
  - Emulation von Gatternetzlisten durch FPGA-Boards
  - Beispiel: Betriebssystem auf Simulationsmodell vom Mikroprozessor booten (Sun Microsystems)
- gemischte Simulation
  - Hardware- und Software
  - auf verschiedenen Abstraktionsebenen
  - + IP-Modelle
  - + analoge Modelle

## Entwurfswerkzeuge (cont.)

- ▶ Analysewerkzeuge
  - ▶ Leistungsverbrauch
  - ▶ Timing
  - ▶ jeweils: statisch, geschätzt oder in Verbindung mit Simulation
- ▶ Verifikation, wenn möglich
  - = Verifikation: Aussagen gelten *für alle möglichen* Eingaben
  - Simulation: Beschränkung auf Stimuli
  - ▶ formale Methoden, um Eigenschaften zu überprüfen
  - ▶ meist Vergleich verschiedener Modelle
    - ▶ in Verbindung mit Extraktion
    - ▶ Referenzmodell, woher?
  - ▶ Ersatz von Simulationen

## Entwurfswerkzeuge (cont.)

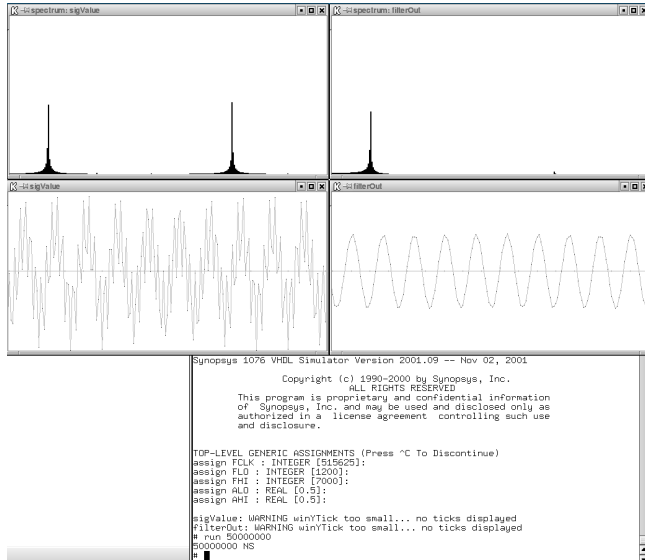
- ▶ Layoutwerkzeuge / Platzierung & Verdrahtung
  - ▶ NP-vollständige Probleme
  - ⇒ Heuristiken
  - ⇒ sehr starke Spezialisierung, z.B. Routing bei Standardzell
- Entwürfen:
  1. Verdrahtung der Spannungsversorgung: Power-Routing
  2. Clock-Tree Synthese / -Routing
  3. zeitkritische Netze bearbeiten: „constraint driven“ Routing
  4. normale Verdrahtung
  5. nachträgliche Optimierung: DRC-Fehler, thermische Modelle...
- ▶ Test des Entwurfs
  - = Testbarkeit: Fertigungsfehler (physikalisch) feststellen
  - Simulation: Überprüfung der Funktion
  - ▶ Ziel: defekte ICs aussortieren, vor Verpackung in Gehäuse

## Entwurfswerkzeuge (cont.)

- ▶ Problem
  - ▶ *alle internen Leitungen/Gatter ansprechen*
  - ▶ *nur die Padzellen sind direkt zugänglich*
- ▶ Fehlermodelle: „stuck-at“, bridging, open. . .
- ▶ Verfahren um Testbarkeit zu gewährleisten
  - ▶ Selbsttest, z.B. BIST (**B**uild **I**n **S**elf **T**est)
  - ▶ Scan-Path: Flipflops als Schieberegister
  - ▶ . . .
- ▶ Dabei wird zusätzliche Logik integriert (bis zu 30%)
  - ▶ (teil-)automatisch bei der Synthese
- ▶ Fehlersimulation: überprüft die Fehlerüberdeckung  
„Wie viele Fehler können erkannt werden?“
- ▶ Testmustergenerierung: erzeugt automatisch Testvektoren

## Beispiel

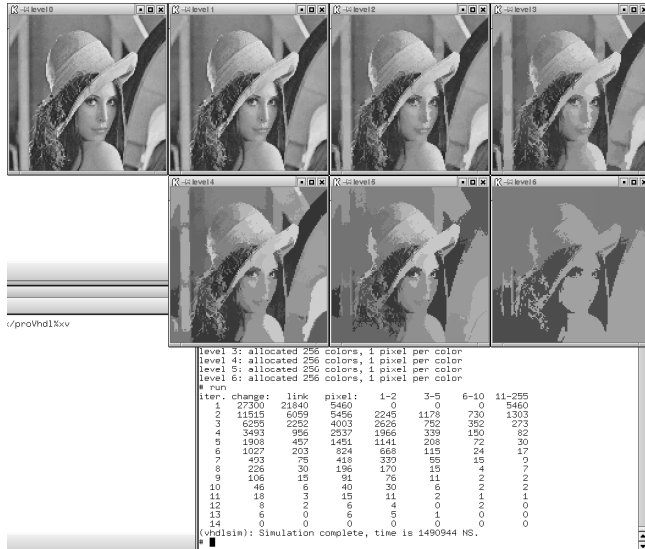
- ▶ Signalverarbeitung
- ▶ digitales Filter



◀ Simulation

## Beispiel

- ▶ Bildverarbeitung
- ▶ Segmentierung



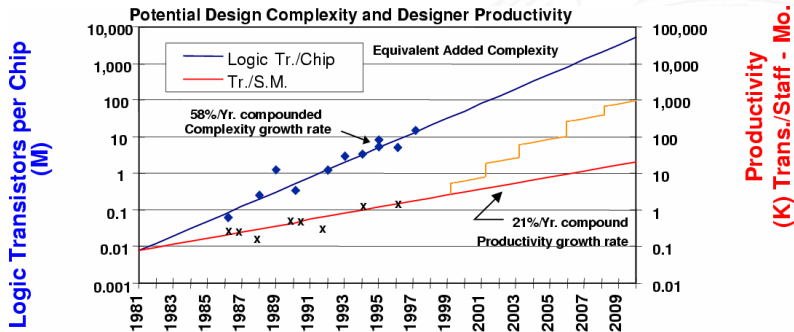
◀ Simulation



# Probleme

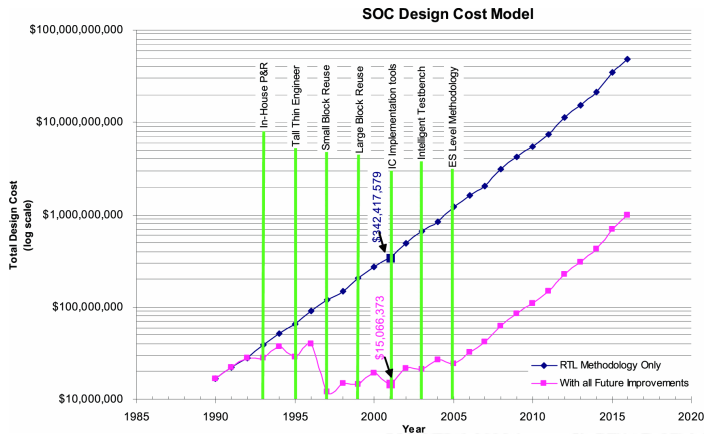
Moore's Law heißt in der Praxis

- ▶ Entwurf immer größerer und komplexerer Systeme
- Produktivitätssteigerungen



# Probleme (cont.)

## — Entwurfskosten



## Probleme (cont.)

- ▶ Geänderte Systemanforderungen
  - ▶ Performance
  - ▶ Größe
  - ▶ ökonomische Randbedingungen
  - ▶ Low-Power: Leistungsaufnahme, Abwärme...
  - ▶ Umgebung: EMV, Temperatur, mechanische Eigenschaften...
- Wie können all diese Anforderungen (formal) spezifiziert werden?

# Gliederung

1. Entwurfsmethodik
2. EDA-Werkzeuge
3. Entwurfstile

Full-Custom

Makro- und Standardzellentwurf

Gate-Array Entwurf

programmierbare Logik: PLDs, FPGAs

Vergleich

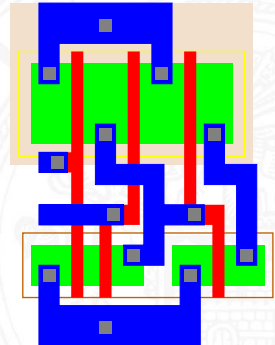
# Entwurfstile

- ▶ mehrere Möglichkeiten Schaltungen zu entwerfen
- ▶ Unterscheidungsmerkmale
  - ▶ Zeitaufwand: Entwurfsdauer, Fertigungszeit
  - ▶ Kosten: Fertigung, pro Stück, EDA-Werkzeuge
  - ▶ IC-Eigenschaften: Größe, Taktfrequenz, Leistungsaufnahme...
- ▶ Entwurfstile
  - ▶ Full-Custom
  - ▶ Standardzell
  - ▶ Gate-Array
  - ▶ FPGA / programmierbare Schaltungen

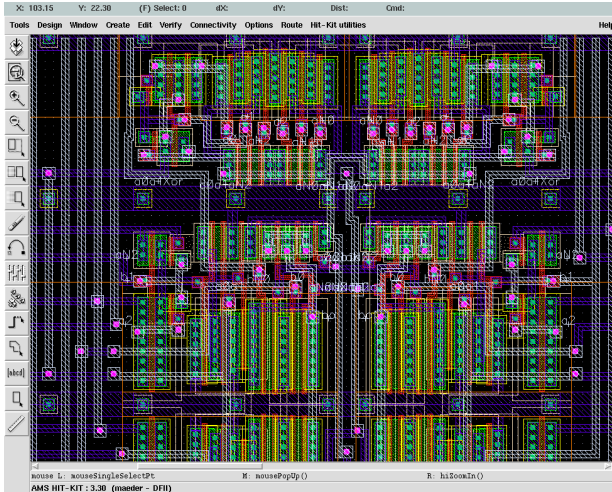
# Full-Custom

## Vollkundenspezifischer Entwurf / Full-Custom

- ▶ Layout aller geometrischer Strukturen
- ▶ viel manuelle Arbeit mit Layout-Editoren
- ▶ optimal kleine, schnelle Entwürfe
- ▶ sehr lange Entwurfsdauer (Effizienz)
- ▶ Ausnutzen von Regularität
- ▶ Teamarbeit nötig, Schnittstellen
- ▶ erfordert erfahrene Entwerfer



# Full-Custom (cont.)

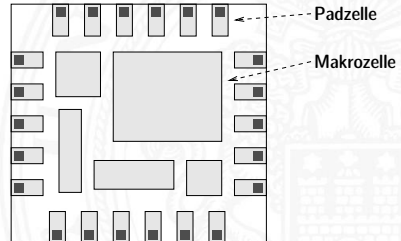


# Makrozellentwurf

## Makrozellentwurf

- ▶ Zellen wie Speicher, ALUs oder Datenpfade werden über Generatoren erzeugt
- ▶ Makrozellen in Full-Custom Qualität
- ▶ meist in Verbindung mit Standardzellentwurf

Chipgröße	variabel
Zellenanzahl	variabel
Zellengröße	variabel
Anschlusslage	variabel
Leiterbahnkanäle	variabel



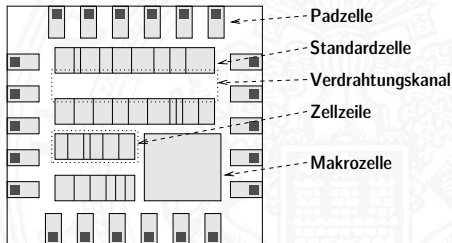


# Standardzellentwurf

## Standardzellentwurf

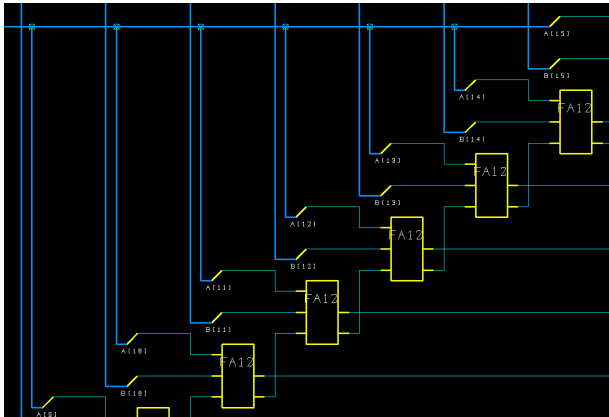
- ▶ vorgefertigte Zellen aus Bibliotheken benutzen
- ▶ Layout der Standardzellen in Full-Custom Qualität
- ▶ schneller flexibler Entwurf
- ▶ meist in Verbindung mit Makrozellgeneratoren

Chipgröße	variabel
Zellenanzahl	variabel
Zellenhöhe	fest
Zellenbreite	variabel
Anschlusslage	variabel
Leiterbahnkanäle	variabel

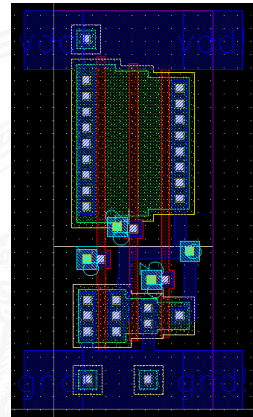


# Standardzellentwurf (cont.)

Schematic

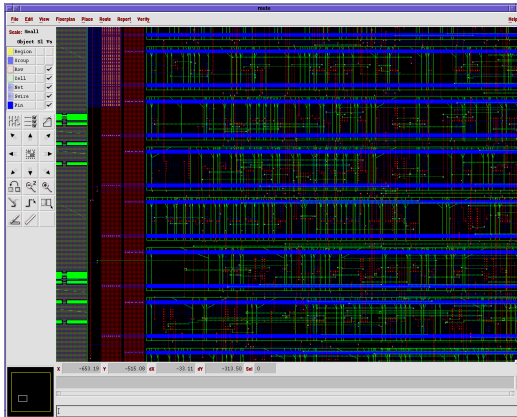


Zell-Layout



# Standardzellentwurf (cont.)

## Standardzell Layout

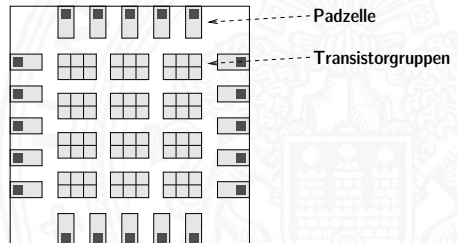


# Gate-Array Entwurf

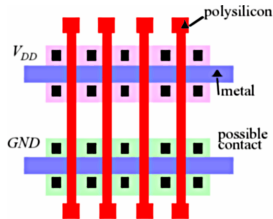
## Gate-Array / Sea-of-Gate Entwurf

- ▶ vorgefertigte Transistoren
- ▶ Layout durch Verbindungsstruktur (Verdrahtung, Kontakte)
- ▶ intra-Zell Verdrahtung aus Zellbibliotheken
- ▶ vorgegebene Master: Komplexität eingeschränkt, Verschnitt
- ▶ schnelle Verfügbarkeit

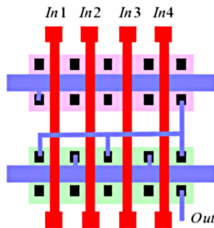
Chipgröße	fest
Zellenanzahl	fest
Zellengröße	fest
Anschlusslage	fest
Leiterbahnkanäle	fest



# Gate-Array Entwurf (cont.)



Uncommitted  
Cell



Committed  
Cell  
(4-input NOR)

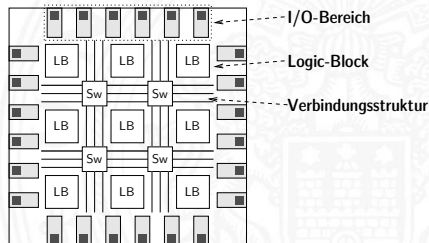
Gate-Array

# programmierbare Schaltungen

programmierbare Schaltungen: FPGA, PLD, LCA...

- ▶ fertig vorgegebene Schaltung: Logik und Verbindungsstruktur
- ▶ Entwurf: Programmierung durch Anwender  $\Rightarrow$  sofort verfügbar
- ▶ Einschränkung durch vorgegebene Struktur
- ▶ Rekonfiguration möglich
- ▶ in-Circuit programmierbar

Chipgröße	fest
Blockanzahl	fest
Anschlusslage	fest
Verbindungsnetz	fest
Blockfunktion	progr.
Verbindungen	progr.



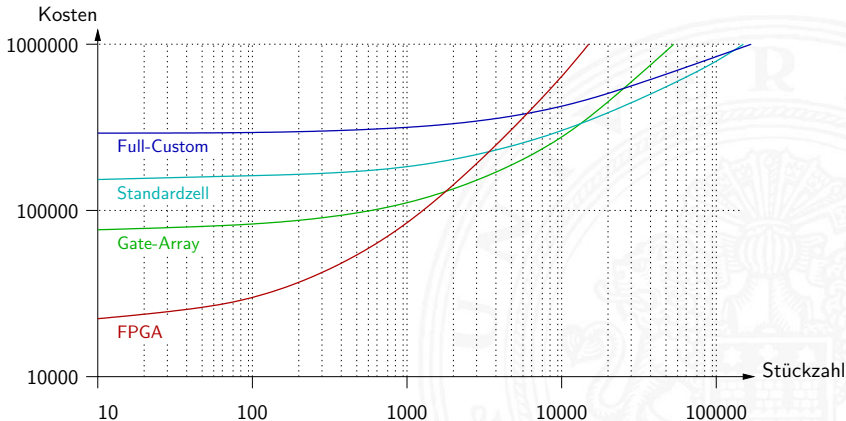
# Vergleich der Entwurfstile

## Tabellarische Übersicht

Stil	Performance	Fläche	Kosten (IC)	Kosten (Design)	time-to-Market	Prozessschritte	Stückzahlen
Full-Custom	+++	+++	+++	---	---	voll	$10^5$
Standard-/Makrozell	++	++	++	--	--	voll	$10^4$
Gate-Array	+	o	+	o	o	4-10	$10^3$
programmierbare Logik	-	--	--	++	+++	0	$< 10^3$

# Vergleich der Entwurfstile (cont.)

## Wirtschaftlichkeit der Entwurfstile





# Vergleich der Entwurfstile (cont.)

## Wahl des Entwurfstils

- ▶ Kostenüberlegungen
  - ▶ Entwurfsdauer: „time-to-Market“
  - ▶ technische Randbedingungen, oft als K.O.-Kriterium
    - ▶ Fläche
    - ▶ Leistungsaufnahme
    - ▶ Sicherheitsaspekte
  - ▶ organisatorische Randbedingungen
    - ▶ vorhandene Werkzeuge
    - ▶ Know-How
    - ▶ „Faktor: Mensch“ (Erfahrungen, Vorlieben)
- ⇒ vielfältige Wechselwirkungen

# Literaturliste

- [BE95] **Abdellatif Bellaouar, Mohamed I. Elmasry:**  
*Low-power digital VLSI design – Circuits and systems.*  
Kluwer Academic Publishers; Boston, MA, 1995.  
ISBN 0-7923-9587-5
- [ITRS07] *International Technology Roadmap for Semiconductors – 2007 Edition.* Semiconductor Industry Association, 2007.  
URL [www.itrs.net/Links/2007ITRS/Home2007.htm](http://www.itrs.net/Links/2007ITRS/Home2007.htm)
- [ITRS11] *International Technology Roadmap for Semiconductors – 2011 Edition.* Semiconductor Industry Association, 2011.  
URL [www.itrs.net/Links/2011ITRS/Home2011.htm](http://www.itrs.net/Links/2011ITRS/Home2011.htm)

## Literaturliste (cont.)

- [MC80] Carver Mead, Lynn Conway:  
*Introduction to VLSI systems.*  
2nd ed. Addison-Wesley; Reading, MA, 1980.  
ISBN 0-201-04358-0
- [She95] Naveed A. Sherwani:  
*Algorithms for VLSI physical design automation.*  
2nd ed. Kluwer Academic Publishers; Boston, MA, 1995.  
ISBN 0-7923-9592-1

## Literaturliste (cont.)

- [T<sup>+</sup>90] Donald E. Thomas [u. a.]:  
*Algorithmic and register-transfer level synthesis*  
 – *The system architect's workbench.*  
 Kluwer Academic Publishers; Boston, MA, 1990.  
 ISBN 0-7923-9053-9
- [WE94] Neil H. E. Weste, Kamran Eshraghian:  
*Principles of CMOS VLSI design*  
 – *A systems perspective.*  
 2nd ed. Addison-Wesley; Reading, MA, 1994.  
 ISBN 0-201-53376-6