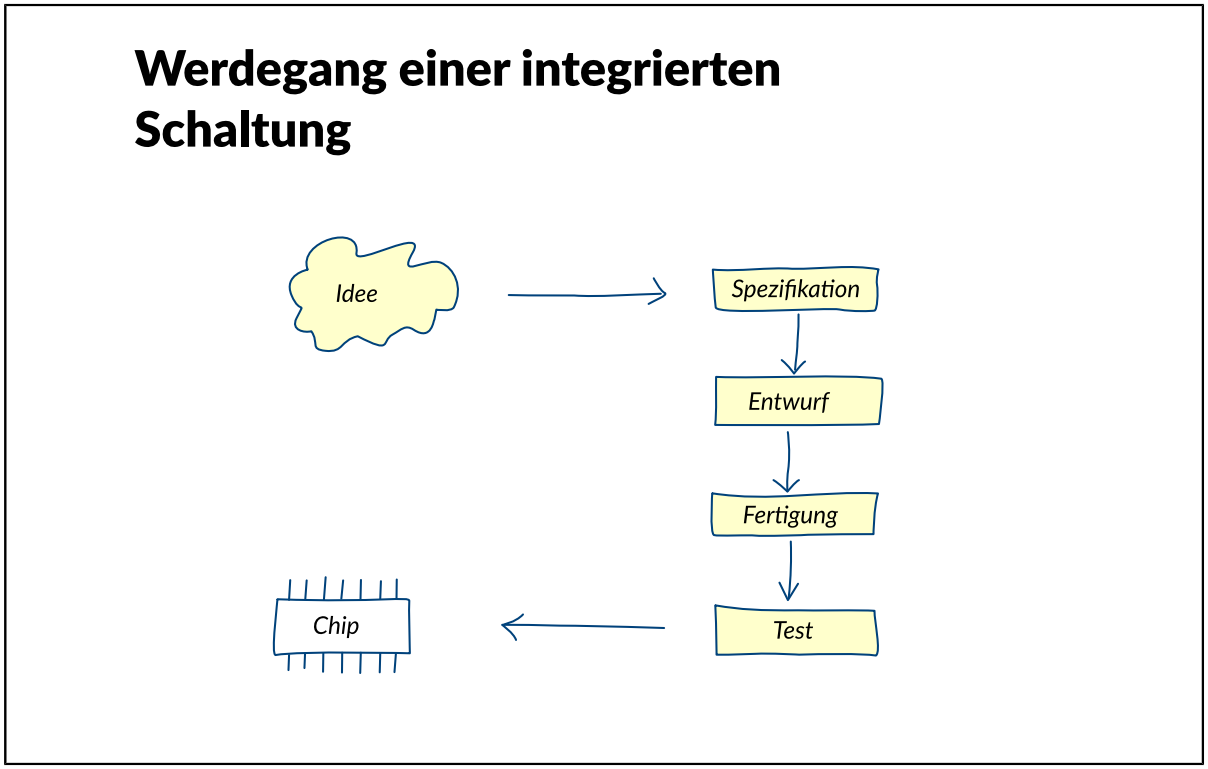
Electronic Design Automation (EDA)

Entwurfsprozess

Entwurfsprozess: Werdegang einer integrierten Schaltung

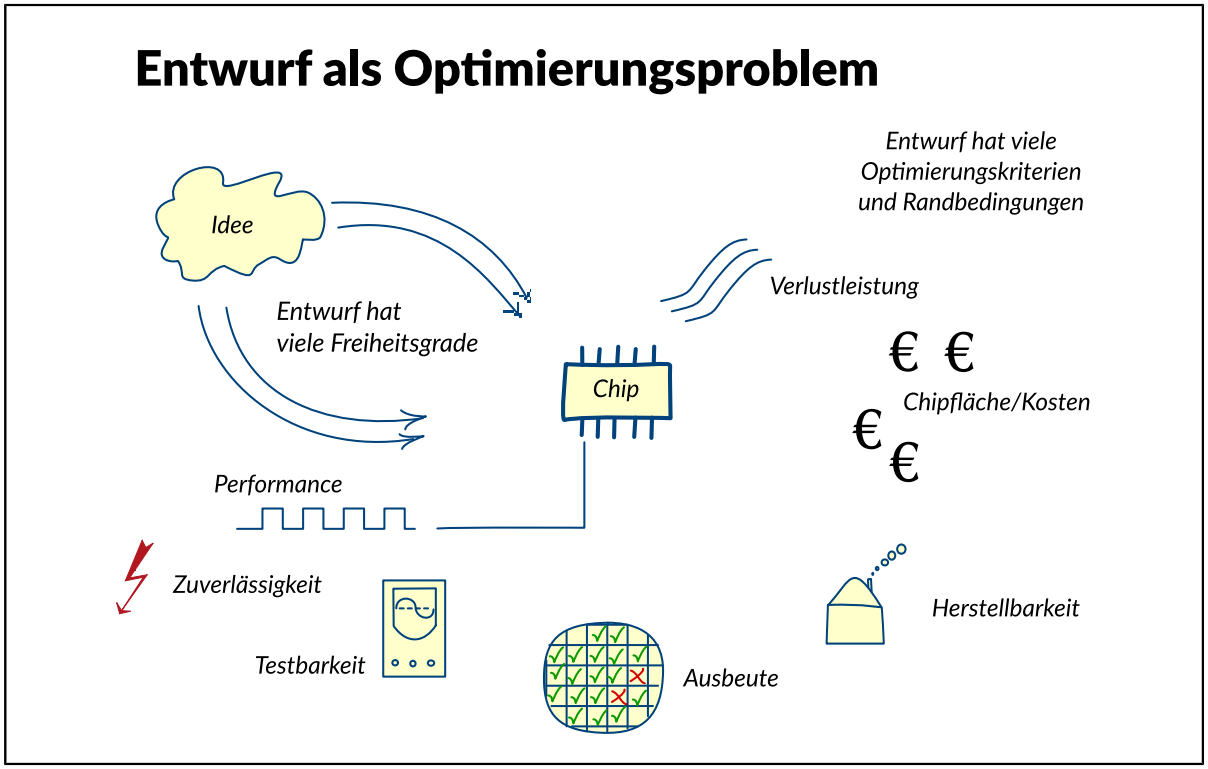


Im Kapitel Systementwurf haben wir bereits die Spezifikation als Ausgangspunkt des Entwurfs kennengelernt.

Wie bei jedem anderen Produkt folgt auf den Entwurfsprozess die Fertigung der integrierten Schaltung. Wie wir jedoch später sehen werden, beeinflussen die Eigenschaften des Fertigungsprozesses - anders als bei vielen anderen Produkten - den Entwurf in vielfältiger Weise. Insbesondere schränken sie die nahezu unendlich großen Entwurfsfreiheiten z.B. durch die Zahl der zur Verfügung stehenden Verdrahtungsebenen oder durch so genannte Entwurfsregeln wie Mindestbreiten oder Mindestabstände, mit denen im Entwurf die Toleranzen des Fertigungsprozesses berücksichtigt werden, wesentlich ein.

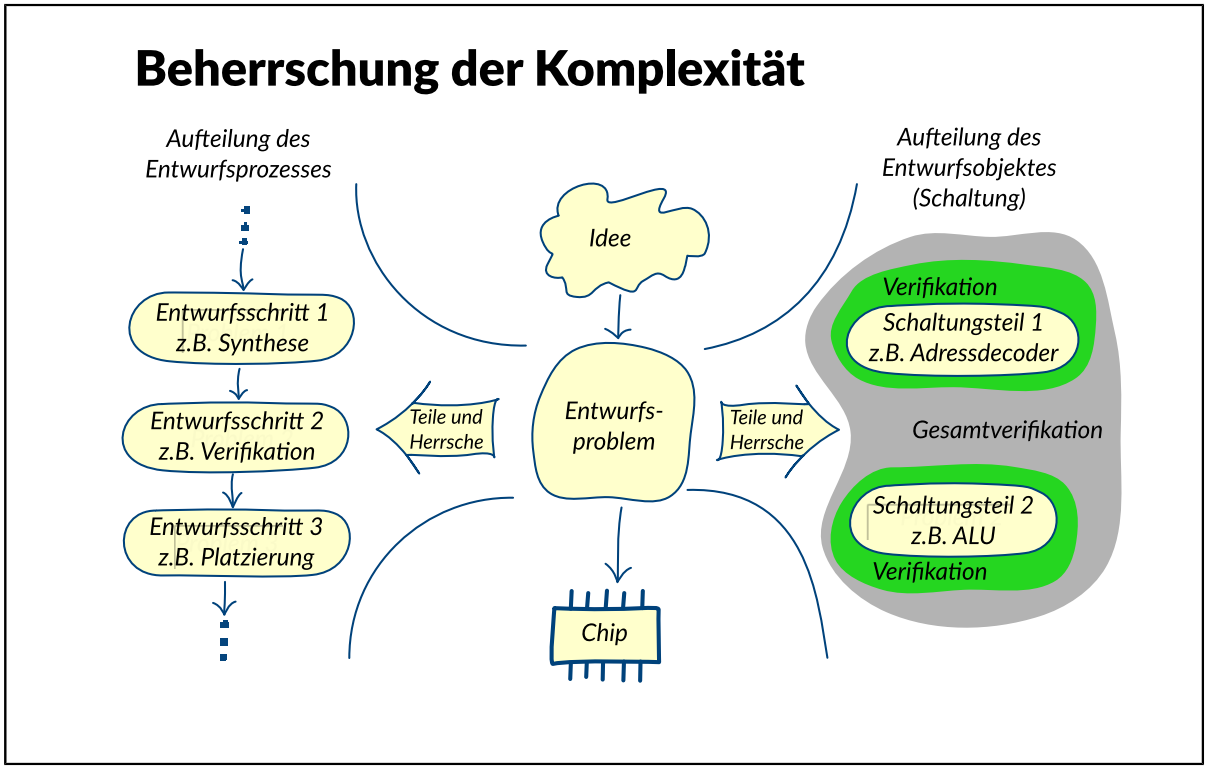
Auf die Fertigung folgt die so genannte Endprüfung, die bei integrierten Schaltungen auch als Test bezeichnet wird. Der Test hochkomplexer Schaltungen stellt eine besondere Herausforderung dar. Es ist deshalb erforderlich, bereits während des Entwurfs Maßnahmen vorzusehen, die einen späteren Test sinnvoll ermöglichen. Dazu gehört die Berechnung geeigneter Testmuster und Maßnahmen zur Erhöhung der Testbarkeit (Design for Testability, DFT).

Entwurfsprozess: Entwurf als Optimierungsproblem



Ziel des Entwurfsprozesses ist es zunächst, eine Schaltung zu entwerfen, die die Spezifikation erfüllt und diese Schaltung in einen Satz von so genannten Maskenbeschreibungen (Layout) umzusetzen, mit der die Schaltung gefertigt werden kann. Wegen der zahlreichen Freiheitsgrade im Entwurf wird es eine nahezu unendlich große Menge von Layouts geben, deren zugehörige Schaltungen die Spezifikation erfüllen. Es ist deshalb wichtig, den Entwurfsprozess als Optimierungsvorgang zu verstehen, der aus den zahlreichen Entwurfsalternativen die bestmögliche auswählt. Zu den Optimierungskriterien zählen dabei die Performance und die Chipfläche, die bereits bei der Erläuterung des Entwurfsprozesses erwähnt wurden, aber auch die Verlustleistung, deren Minimierung besondere Bedeutung zukommt, die Robustheit gegen Fertigungsschwankungen, die Testbarkeit und viele andere. Bei der Besprechung der einzelnen Entwurfswerkzeuge werden wir genauer erkennen, wie dieser Optimierungsvorgang verläuft. Sehr häufig werden wir auf gegenläufige Optimierungsziele stoßen, so dass ein geeigneter Kompromiss zu finden ist. Dies gilt in besonderem Maße für die grundlegenden Optimierungsziele Performance und Fläche.

Entwurfsprozess: Beherrschung der Komplexität

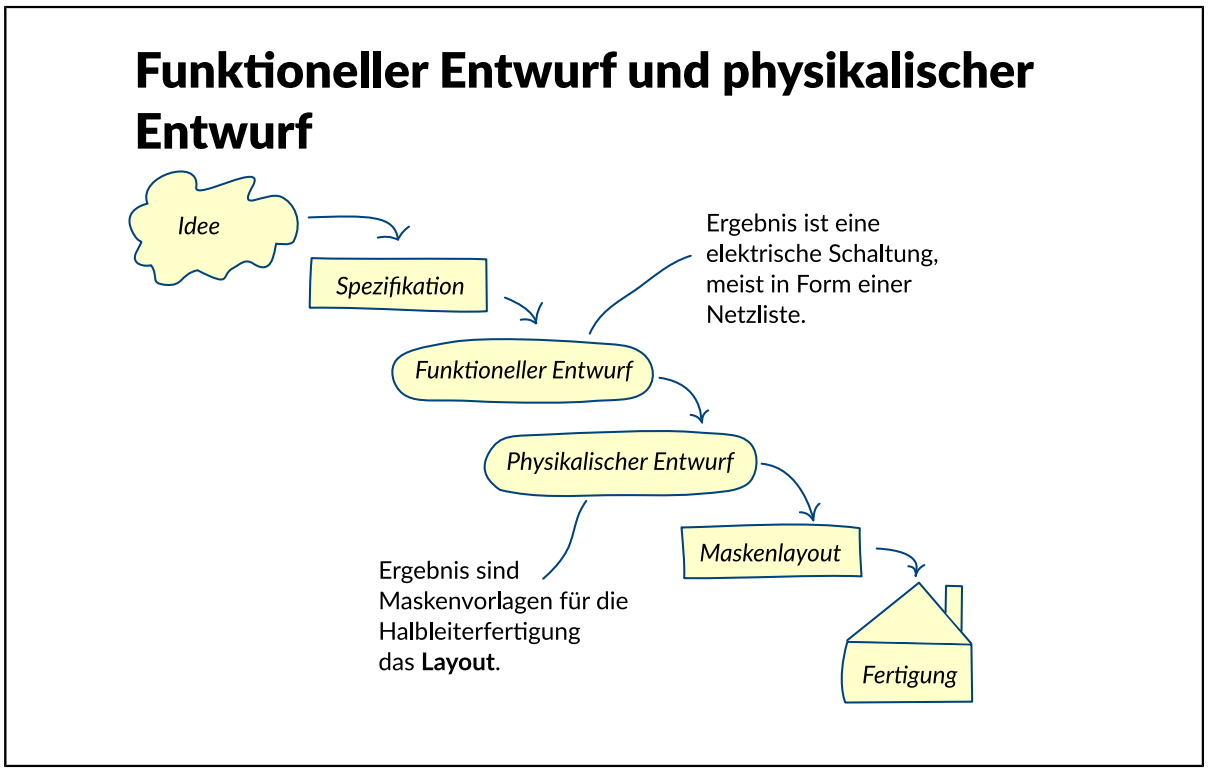


Beim Entwurf integrierter Schaltungen handelt es sich um einen Vorgang, der die Verwaltung und die systematische Behandlung von Millionen bzw. Milliarden einzelner Objekte betrifft. Es ist unschwer einzusehen, dass dieser Vorgang selbst ein äußerst komplexes Unternehmen darstellt, das nur durch eine systematische Durchführung beherrschbar ist. Daher ist es das Ziel von EDA, eine vollständige Entwurfsautomatisierung zu erreichen, d.h. eine weitestgehend ohne Benutzeraktion durchgeführte Transformation einer möglichst abstrakten Entwurfsspezifikation in eine Maskengeometrie.

Da ein vollständig automatischer Entwurf mit gleichzeitiger Garantie der Korrektheit des Entwurfsergebnisses zumindest heute utopisch erscheint, haben sich in der Praxis Entwurfsstrategien herausgebildet, die auf dem Stand der Technik das Komplexitätsproblem beherrschbar machen. Die beiden wichtigsten Strategien sind:

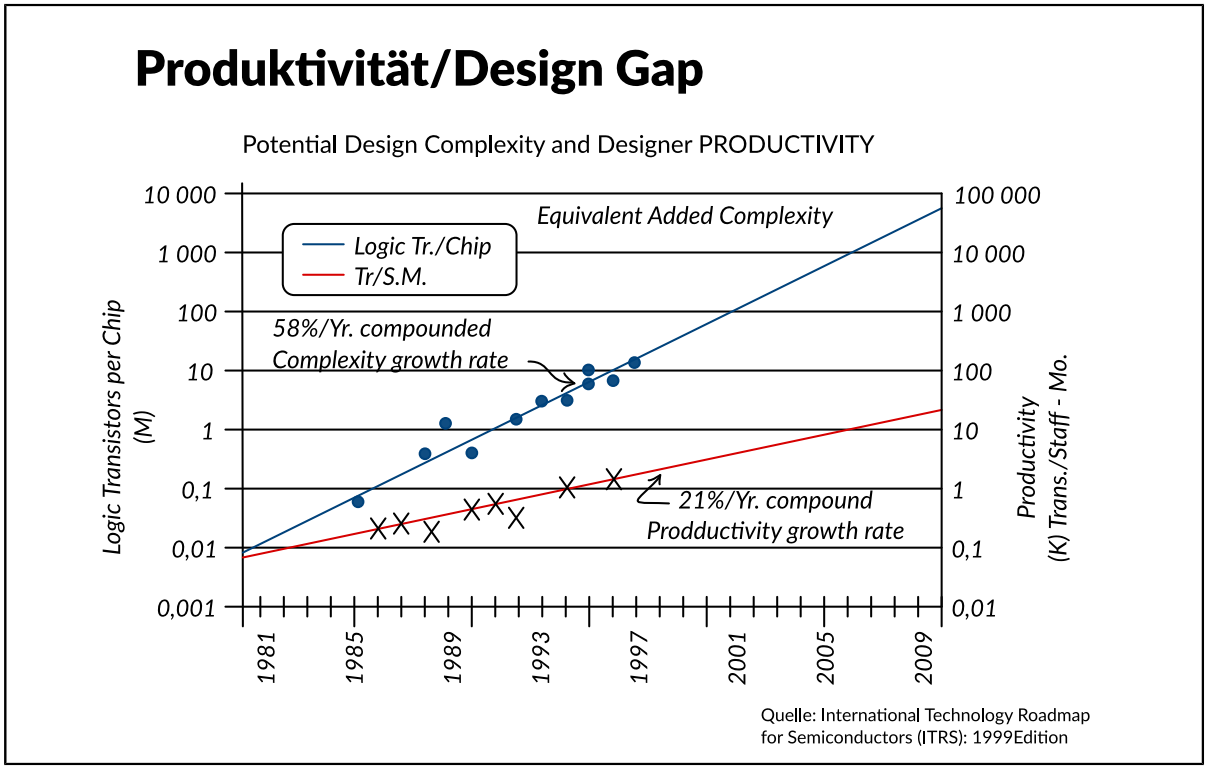
- Aufteilung des Entwurfsprozesses in einzelne Entwurfsschritte (Teile und Herrsche-Prinzip, divide and conquer). Dieses Prinzip wird einerseites bei der Aufteilung des gesamten Entwurfsvorgangs in einzelne Entwurfsschritte und andererseites bei der Aufteilung einer großen Schaltung in kleinere Teile mit sequentieller Abarbeitung verwendet.
- Einbau von Prüfschritten im Anschluss an jeden Entwurfsschritt (Verifikation). Der Entwurfsprozess kann als kontinuierliche Folge von synthetisierenden und analysierenden Schritten (Entwurfs- und Prüfschritte) angesehen werden.

Entwurfsprozess: Funktioneller und physikalischer Entwurf



In der Praxis hat sich eine Aufteilung des Entwurfsprozesses in zwei Phasen, den so genannten funktionellen und den physikalischen Entwurf bewährt. Der von der Spezifikation ausgehende funktionelle Entwurf endet mit dem Vorliegen einer elektrischen Schaltung, meist in Form einer so genannten Netzliste, im physikalischen Entwurf wird diese dann in ihre geometrische Repräsentation, die für die Fertigung erforderlichen Maskenvorlagen, übersetzt. Dabei sind – in wesentlich stärkerem Umfang als im funktionellen Entwurf – die Randbedingungen der zugrundegelegten Herstellungstechnologie zu beachten.

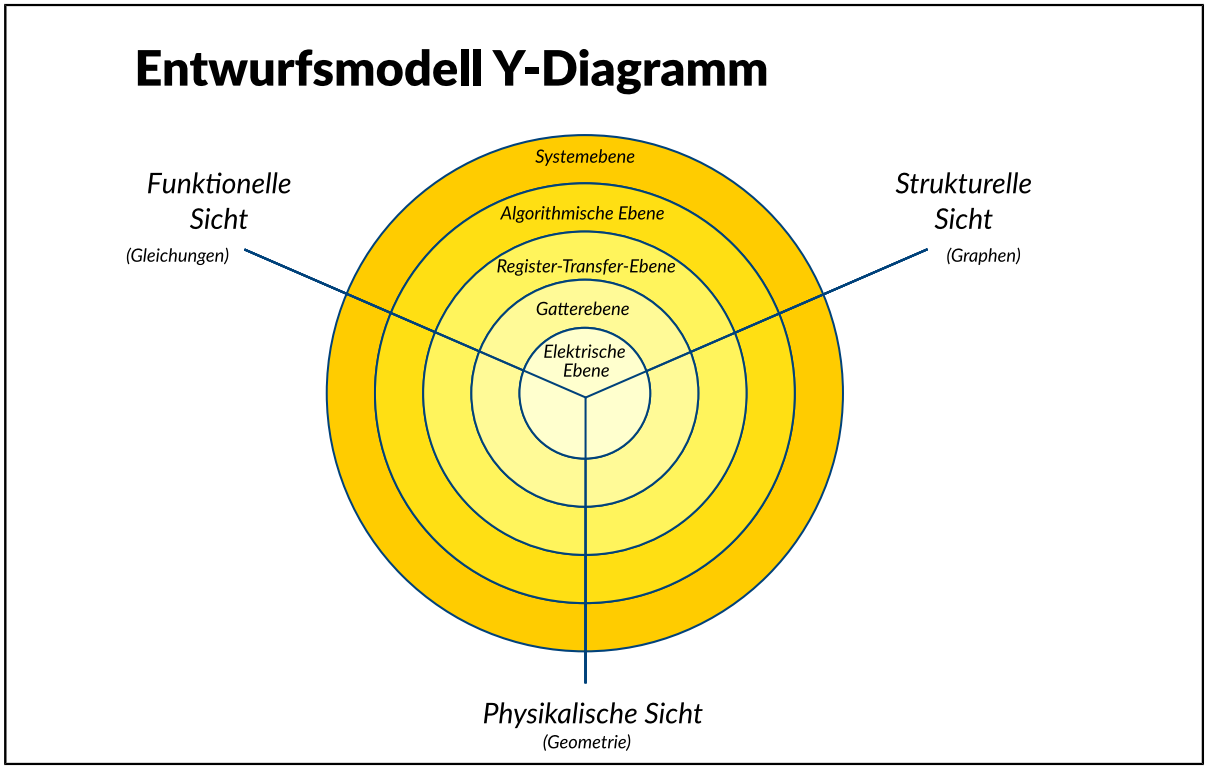
Entwurfsprozess: Produktivität



Kernaufgabe der EDA-Werkzeuge im Entwurfsprozess ist die Beherrschung der Komplexität des Problems (Machbarkeit). Daneben stellen sie ein wesentliches Hilfsmittel zur Steigerung der Entwurfsproduktivität (Effizienz) dar. Leider ist eine quantitative Ermittlung der Produktivität im Entwurf schwierig; alle Rechenmodelle dazu sind stark umstritten. Ein primitiver Ansatz wählt als Produktivitätsmaß die Zahl der von einem Entwickler pro Tag entworfenen Transistoren. Zwar ist es unstrittig, dass dieser Wert durch EDA Werkzeuge stark beeinflusst wird; gleichzeitig sind jedoch auch andere Effekte (Technologie, Designstil, Wiederverwendung von Schaltungsteilen, Fähigkeit der Entwickler u.a.) wirksam, die es nahezu unmöglich machen, den EDA-Einfluss zahlenmäßig nachzuweisen.

In industriellen Studien wurde gezeigt, dass die Entwurfsproduktivität in den letzten zehn Jahren deutlich langsamer gewachsen ist als die Problemgröße (gemessen in der Anzahl der Transistoren pro IC). Die sich damit öffnende Schere (Design Gap) wieder zu schließen, erfordert gewaltige Anstrengungen (und Aufwendungen) für die Entwurfsmethodik und die EDA-Werkzeuge. Diese Anstrengungen müssen vor allem den Entwurfsschritten gelten, bei denen der Automatisierungsgrad heute noch relativ gering ist.

Entwurfsprozess: Entwurfsmodell Y-Diagramm

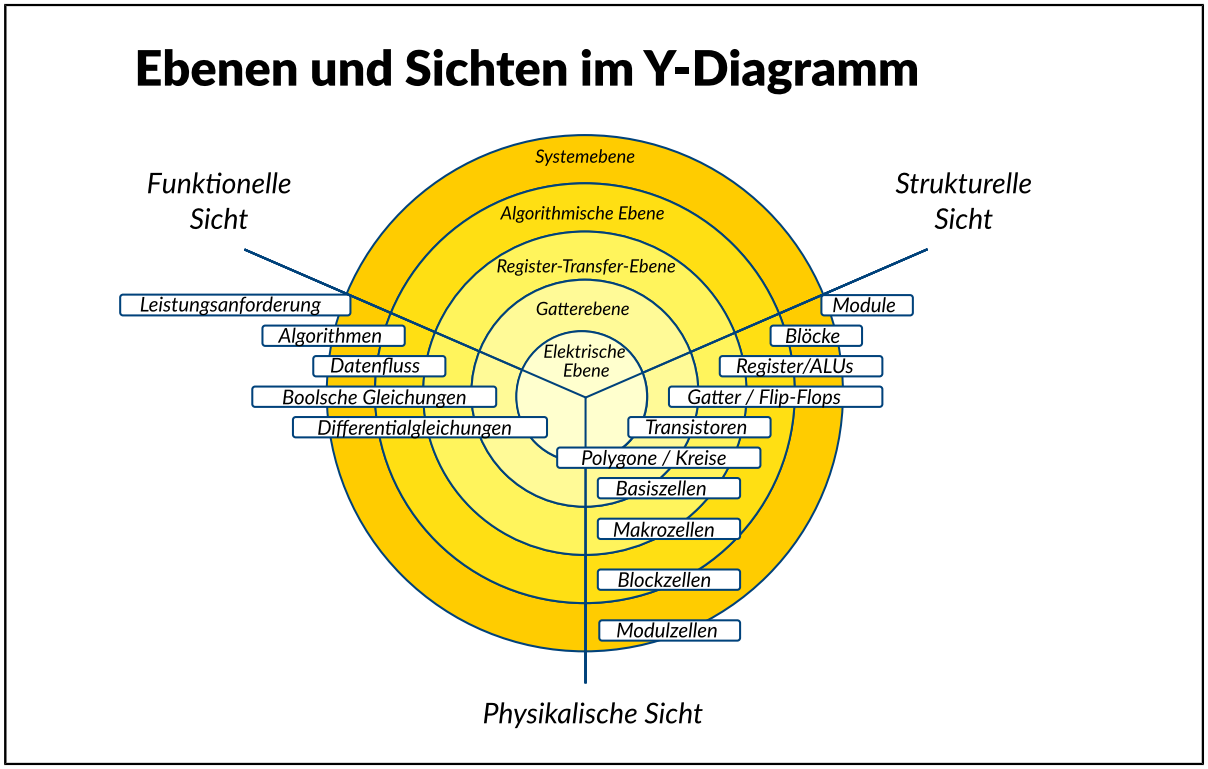


Es gibt viele Ansätze, die verschiedenen Dimensionen des Entwurfs in systematischer Weise zu einem Entwurfsmodell miteinander zu verknüpfen. Das am weitesten verbreitete und wohl auch – zumindest für das Grundverständnis – beste ist das 1983 von Gajski und Kuhn für digitale Schaltungen vorgestellte Y-Modell, das von Walker und Thomas 1985 weiter verfeinert wurde. Es bietet neben der Einfachheit und Eleganz der Darstellung den Vorteil, dass sich die einzelnen Entwurfsstile – wie wir später sehen werden – in übersichtlicher Weise graphisch darstellen lassen.

Im Y-Modell ist der Abstraktionsgrad eines Entwurfs in so genannten Entwurfsebenen dargestellt. Die verschiedenen Repräsentationen eines Entwurfs auf einer Ebene sind als drei prinzipiell unterschiedliche Sichten (funktionell, strukturell, physikalisch) dargestellt.

Es sei an dieser Stelle angemerkt, dass das hier verwendete Modell, in der konkret gewählten Terminologie und in der Bedeutung einzelner Begriffe nicht exakt mit dem Original übereinstimmt (was in unserem Falle schon wegen der Übersetzung nicht möglich ist), aber dem Grundgedanken in keiner Hinsicht widerspricht.

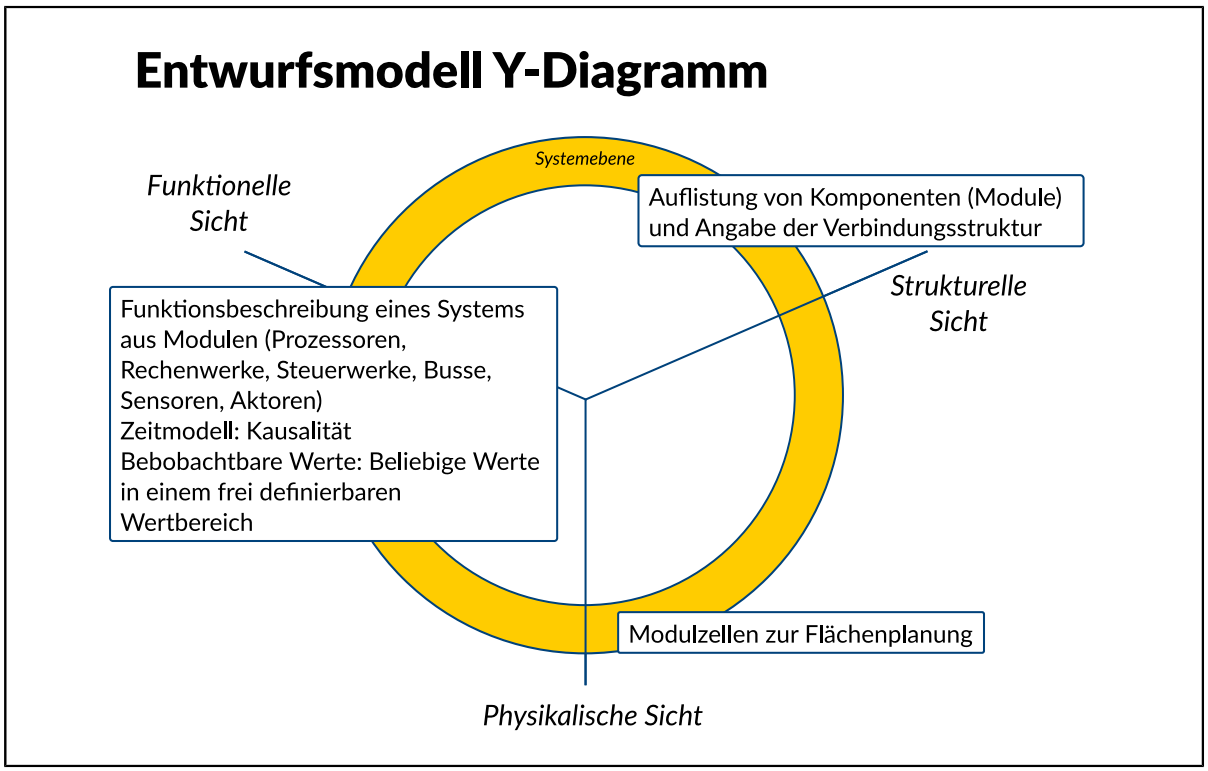
Entwurfsprozess: Ebenen und Sichten im Y-Diagramm



Das Modell zeigt drei Achsen, die in Form eines Y angeordnet sind. Sie gehen vom Zentrum einer Reihe konzentrischer Kreise aus. Jede dieser Achsen repräsentiert einen speziellen Aspekt des Entwurfsgegenstands, eine Sicht. Jede der Sichten beschreibt bestimmte Eigenschaften, die den Entwurfsgegenstand charakterisieren. Die Gesamtheit aller Sichten liefert eine vollständige Beschreibung aller relevanten Aspekte des Entwurfsgegenstands. Die funktionelle Sicht beschreibt alle Aspekte des Verhaltens des Entwurfsgegenstands. Dies beinhaltet Operationen, die vom Entwurfsgegenstand ausgeführt werden, sowie sein Zeitverhalten. Typisches Beschreibungsmittel sind mathematische Gleichungen. Die strukturelle Sicht beschreibt die logische Struktur bzw. die abstrakte Implementierung des Entwurfsgegenstands in Form der topologischen Anordnung von Komponenten und deren Verbindungen. Das dazu passende mathematische Modell sind Graphen, die häufig auf Netzlisten abgebildet werden. Die physikalische Sicht beschreibt die konkrete Implementation des Entwurfsgegenstands, das heißt die Realisierung der (strukturellen) Komponenten mit Hilfe realer physikalischer Objekte. Dies beinhaltet die exakte geometrische Ausdehnung und Anordnung aller Komponenten und Verbindungsstrukturen, mathematische Beschreibungsform ist also die Geometrie.

Es gibt Versionen dieses Entwurfsmodells, in denen weitere Aspekte des Entwurfs, wie Zeitverhalten oder Testbarkeit in weiteren Sichten repräsentiert werden. Hier werden solche Informationen jedoch als in den drei allgemeinen Sichten verteilt betrachtet. In jeder der drei Sichten besteht die Entwurfsbeschreibung aus mehreren Dokumenten, die den Entwurfsgegenstand jeweils unterschiedlich abstrakt und detailliert repräsentieren. Der jeweilige Abstraktions- bzw. Detailliertheitsgrad einer Beschreibung wird durch die Entwurfsebenen charakterisiert. In unserem Modell unterscheiden wir sechs Entwurfsebenen.

Entwurfsprozess: Systemebene



Funktionelle Sicht:

- Modellierungskonzept: Ein System, bestehend aus Modulen wie z.B. Prozessoren, Rechenwerke, Steuerwerke, Busse, Sensoren, Aktoren. Die jeweiligen Module sind charakterisiert durch ihre Funktionalität (z.B. dem Instruktionssatz), Leistungskriterien (z.B. Taktfrequenz), Kommunikationsprotokolle
- Zeitmodell: Kausalität
- Beobachtbare Werte: Beliebige Werte in einem frei definierbaren Wertebereich

Strukturelle Sicht:

- Auflistung von Komponenten (Module) und Angabe der Verbindungsstruktur

Physikalische Sicht:

- Modulzellen, mit denen Flächenplanung im weiteren Sinn durchgeführt werden kann

Entwurfsprozess: ... Modellierungskonzept

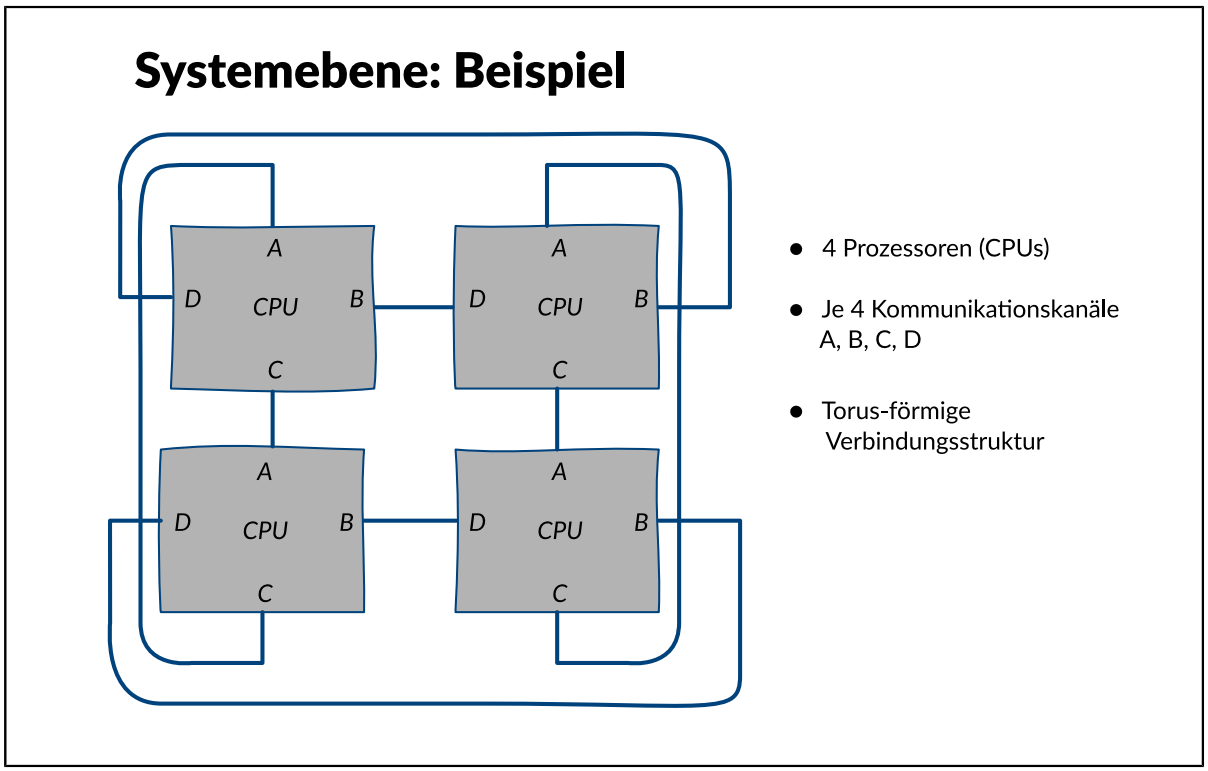
Systemebene: Modellierungskonzept

- System kooperierender Module
- Abstrakte Datentypen (ADT)
- Aus Softwaresicht: Objektorientierte Programmierung
- Moduleigenschaften:
  - Grundlegende Funktionalität (z.B. Syntax und Semantik des Instruktionssatzes eines Prozessors)
  - Leistungsrestriktionen (z.B. Taktfrequenz)
  - Syntax und Semantik der globalen Kommunikationsstruktur (Bsp. Protokolle)

Das Modellierungskonzept auf dieser Ebene für die funktionelle Sicht (Verhalten) ist durch ein System kooperierender Module gegeben. Aus einer theoretischen Sichtweise sind abstrakte Datentypen (ADT) gut geeignet, als konzeptionelles Modell für diese Ebene zu dienen. Wenn man sich aus dem Bereich der Software nähert, so passt das Konzept der objektorientierten Programmierung sehr gut. Drei hauptsächliche Eigenschaften müssen pro beteiligtem Modul spezifiziert werden:

- Für jedes Modul muss die grundlegende Funktionalität angegeben werden. Dies bedeutet z.B. bei einem Prozessor nichts anderes, als Syntax und Semantik seines Instruktionssatzes zu definieren.
- Die grundlegenden Restriktionen auf dieser Ebene sind solche bezüglich der Leistung. Sie können global oder pro beteiligtem Modul spezifiziert sein, z.B. die Taktfrequenz.
- Schließlich müssen Syntax und Semantik der globalen Kommunikationsstruktur angegeben werden. Dies geschieht durch Definition von Protokollen für jede existierende Kommunikationsverbindung (Bus).

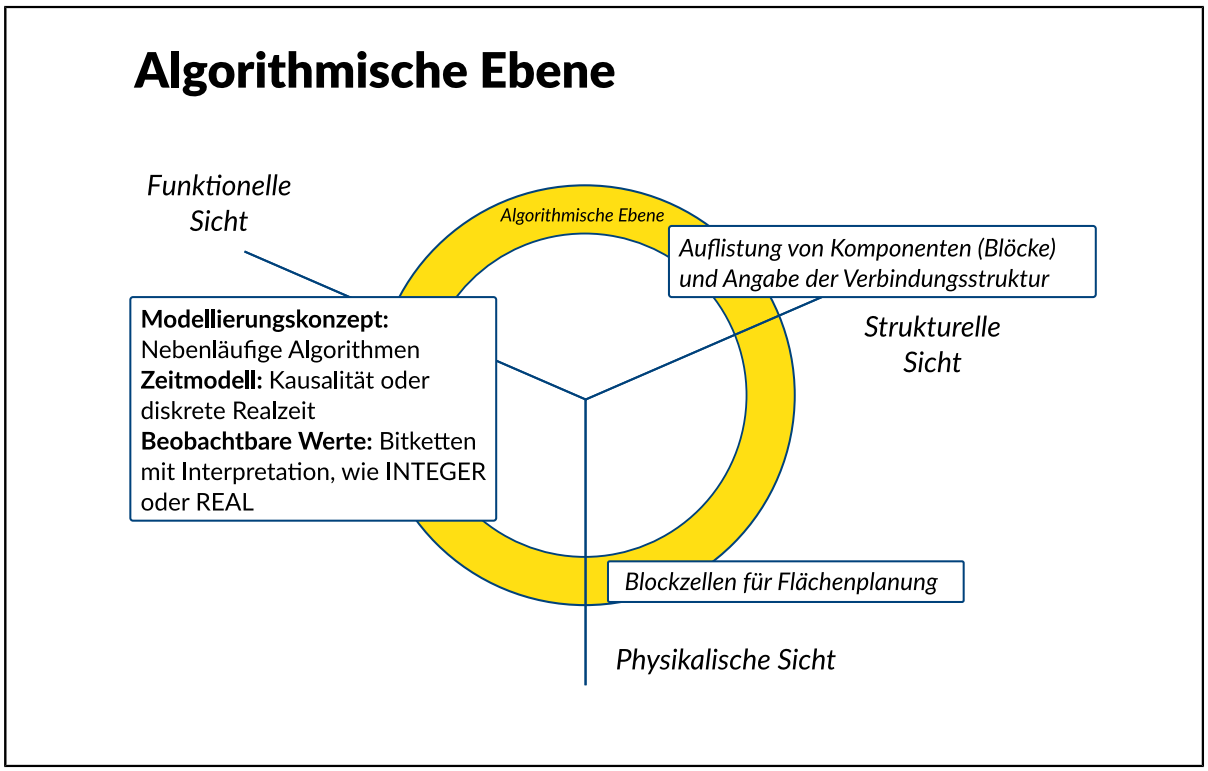
Entwurfsprozess: ... Beispiel



Die Struktur auf dieser Ebene ist durch einfaches Auflisten der beteiligten Komponenten und durch Angabe der Verbindungsstruktur gegeben. Diese Verbindungen (zumeist Busse) können auf dieser Ebene ebenfalls als Komponenten angesehen werden. Ihnen werden in der Verhaltenssicht die Kommunikationsprotokolle zugeordnet. Neben dieser statischen Struktur kann zusätzlich eine dynamische existieren. Sie gibt pro Modul an, von welchen anderen Modulen es Dienste anfordert und für welche es Dienste anbietet.

Auf dieser Ebene existiert sehr wenig geometrische Information. Andererseits finden grundlegende mechanische Entscheidungen auf dieser Ebene statt. So kann auf dieser Ebene beispielsweise die dreidimensionale Anordnung der Komponenten bestimmt werden.

Entwurfsprozess: Algorithmische Ebene

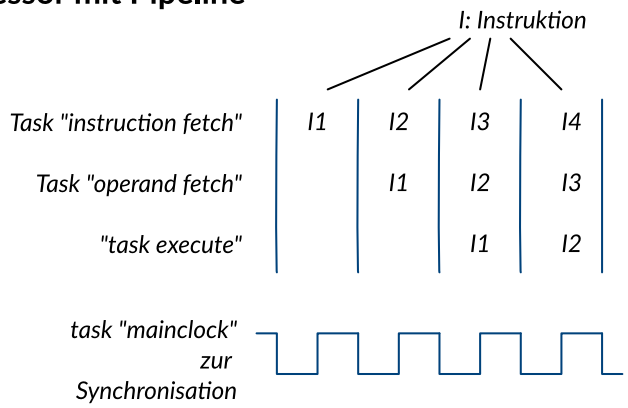


Algorithmen sind üblicherweise hochgradig nebenläufig. Daher erscheinen Modellierungskonzepte wie Petri-Netze oder Datenflussgraphen für diesen Zweck gut geeignet. Bezüglich des Zeitmodells interessiert man sich auf dieser Ebene in den meisten Fällen weiterhin nur für Kausalitäten. Allerdings wird in manchen Fällen ein diskretes Zeitmodell angenommen, wobei man ein bestimmtes Taktschema im Auge hat. Die beobachtbaren Werte sind konkreter als auf der Systemebene. Ihre Eigenschaft als Bitketten ist nun in den meisten Fällen sichtbar. Doch ist dies weiterhin von geringem Interesse, man konzentriert sich auf die typspezifische Interpretation (z.B. als Integer). Bezüglich der Struktur müssen wir zwischen der Komposition einer Kontrollstruktur aus Komponenten wie "While-Schleife" oder "Fork/Join" (Kontrollpfad) und der Struktur des Operationsteils des Algorithmus (Datenpfad) unterscheiden. Im letzteren Fall werden auf dieser Ebene bereits Hardware-Komponenten wie z.B. ALUs benutzt. Die geometrischen Informationen sind auf dieser Ebene immer noch recht abstrakt, jedoch findet hier bereits auf der Basis von Blockzellen eine konkrete Flächenplanung statt.

Entwurfsprozess: ... Beispiel

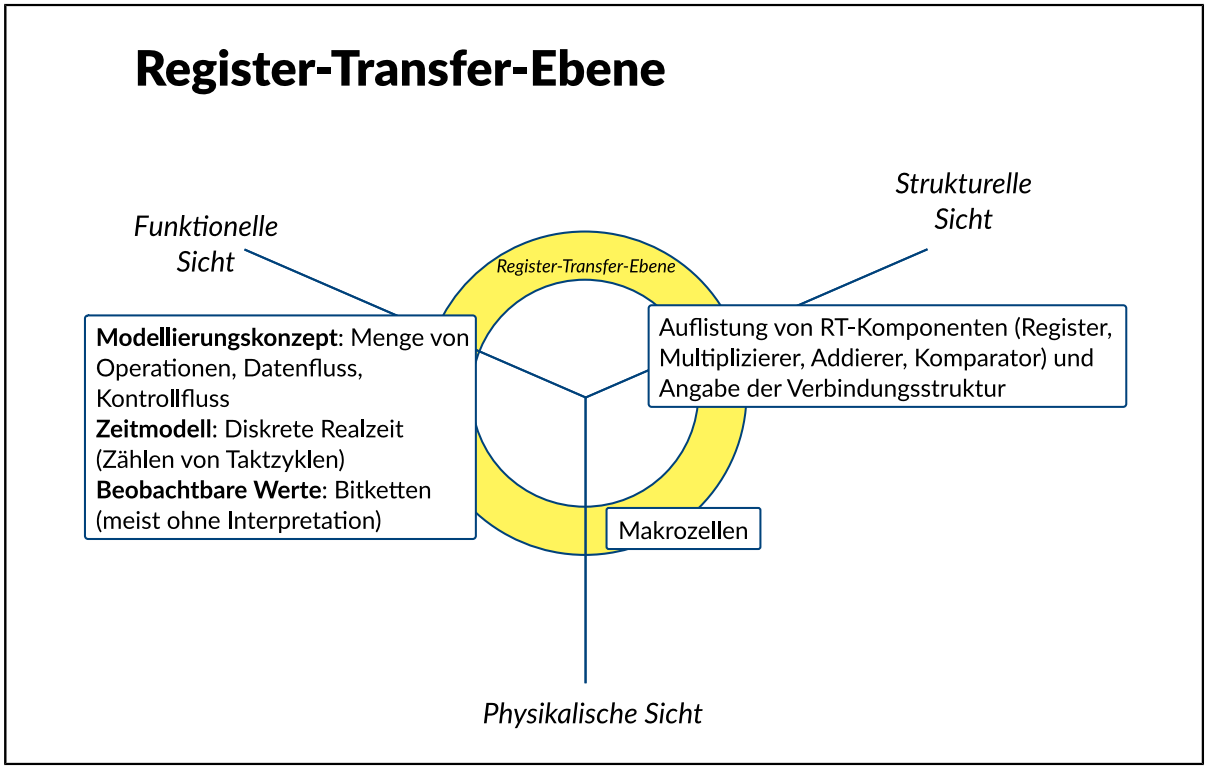
Algorithmische Ebene: Beispiel

Prozessor mit Pipeline



Angenommen wird ein gewöhnlicher Prozessor vom von-Neumann-Typ. Er habe einen Interpretationszyklus, bestehend aus "instruction fetch", "operand fetch" und "execute". Weiterhin wird angenommen, dass diese drei Aktivitäten im Pipelining ablaufen, also nebenläufig. Sie sollen mittels eines Taktes, genannt "mainclock", synchronisiert werden.

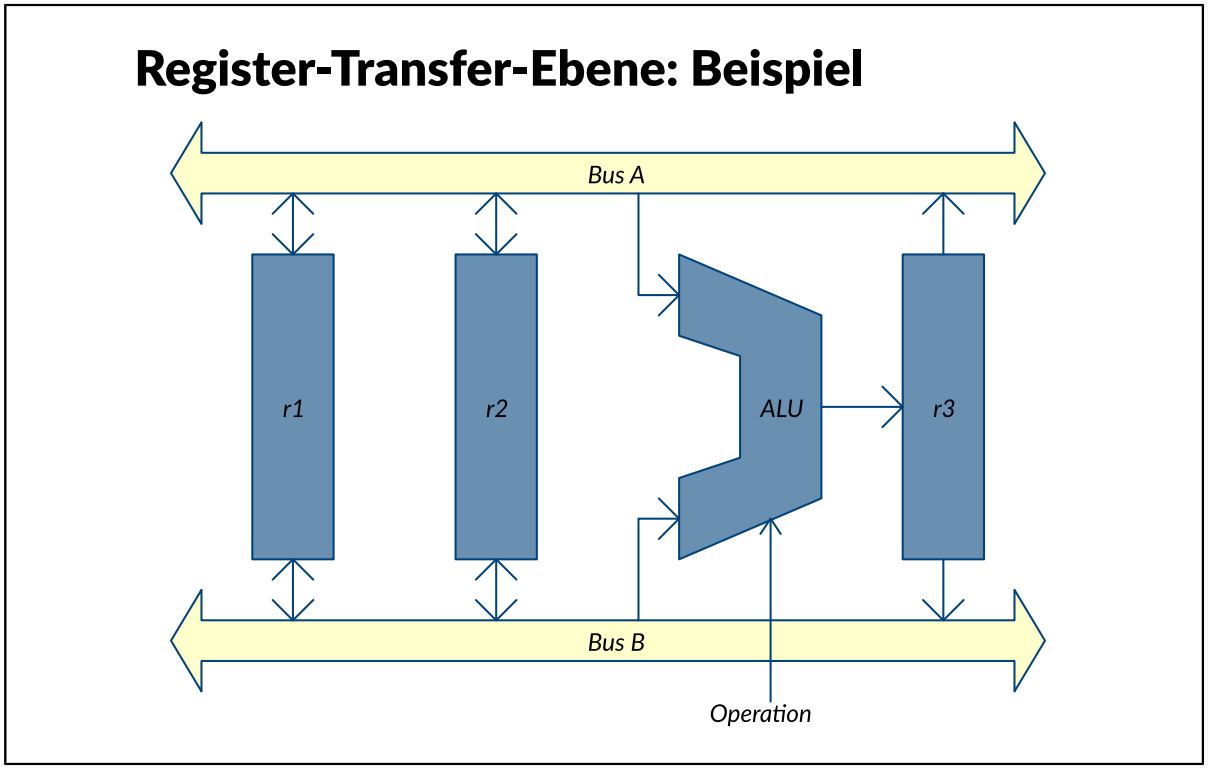
Entwurfsprozess: Register-Transfer-Ebene



Auf der algorithmischen Ebene wird das System meist in einer imperativen Weise betrachtet. Das heißt, dass die Sichtweise die des Steuerwerks ist (Kontrollfluss). Dieses entscheidet, wann nach welchen vorhergehenden Aktionen eine bestimmte Aktion durchgeführt werden darf. Die strikt sequentielle Anordnung in üblichen Programmiersprachen wird hier generalisiert, um auch Nebenläufigkeit zu erlauben. Auf der Registertransferebene (RT-Ebene; englisch Register-Transfer-Level, RTL) wird eine reaktive Sichtweise eingenommen. Das System wird nun aus Sicht der gesteuerten Objekte (Rechenwerke) betrachtet (Datenfluss). Jedes derartige Objekt beobachtet kontinuierlich eine objektspezifische Bedingung. Ist diese Bedingung wahr, führt das Objekt seine Aktion durch. Dabei modifiziert es üblicherweise die Bedingungen innerhalb des Bedingungsraumes, und kann die Ausführung anderer Objekte (einschließlich seiner selbst) ermöglichen.

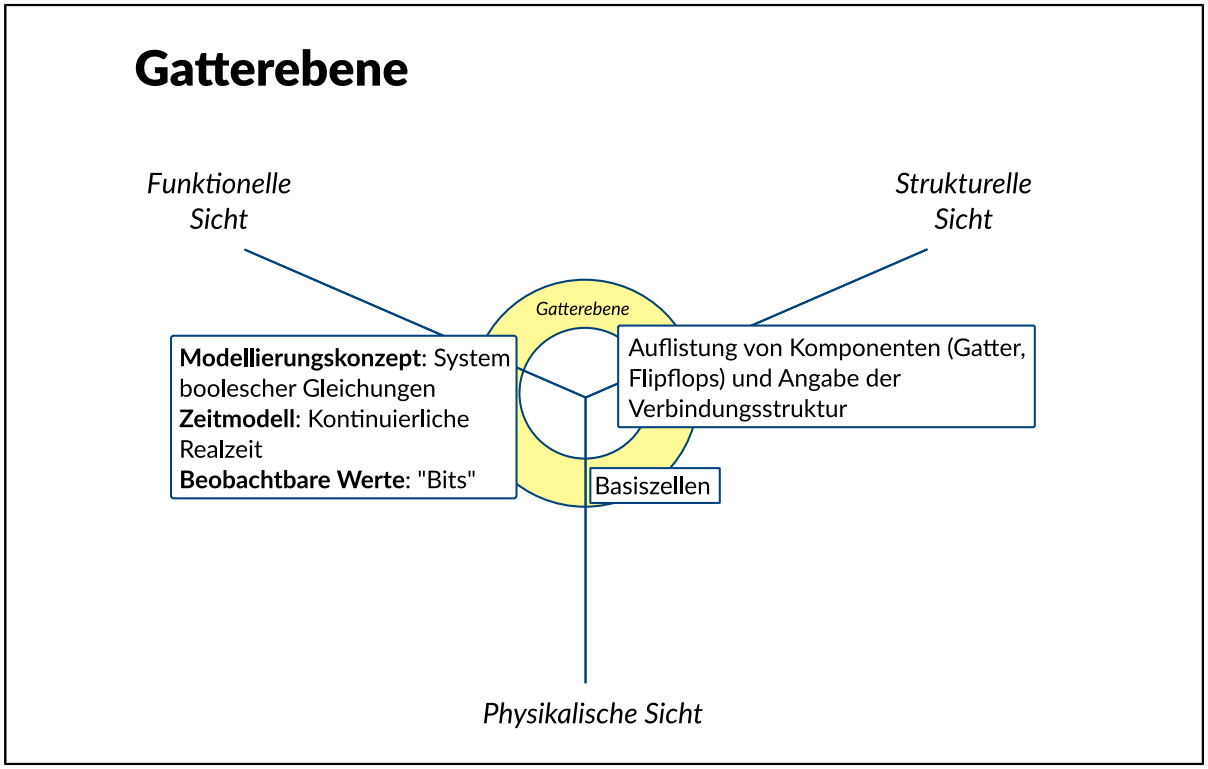
Auf dieser Ebene ist man üblicherweise an einem spezifischen synchronen Taktschema interessiert. Daher wird das Zeitschema auf dieser Ebene meist durch das Zählen von Taktzyklen gegeben. Es hängt von dem jeweiligen Implementierungskonzept für das Zeitschema ab, ob Taktpegel, steigende, fallende oder beide Flanken, oder gar eine Mischung dieser Techniken benutzt werden. Die beobachtbaren Werte sind nun Bitketten. In den meisten Fällen wird ihnen nicht mehr eine feste Interpretation (Typ) zugeordnet. Vielmehr werden sie von verschiedenen Objekten unterschiedlich interpretiert. Auf der RT-Ebene wird die endgültige Hardwarestruktur sichtbar. Das System wird daher als Verschaltung von Registertransfermodulen beschrieben. Typische derartige Module sind Register, ALU's, Multiplexer, Kodierer, Dekodierer, Schiebebausteine, etc. Aus physikalischer Sicht findet auf der RT-Ebene die endgültige Flächenplanung statt. Zunehmend häufig existieren für strukturelle Komponenten auch bereits physikalische Realisierungen (Makrozellen).

Entwurfsprozess: ... Beispiel



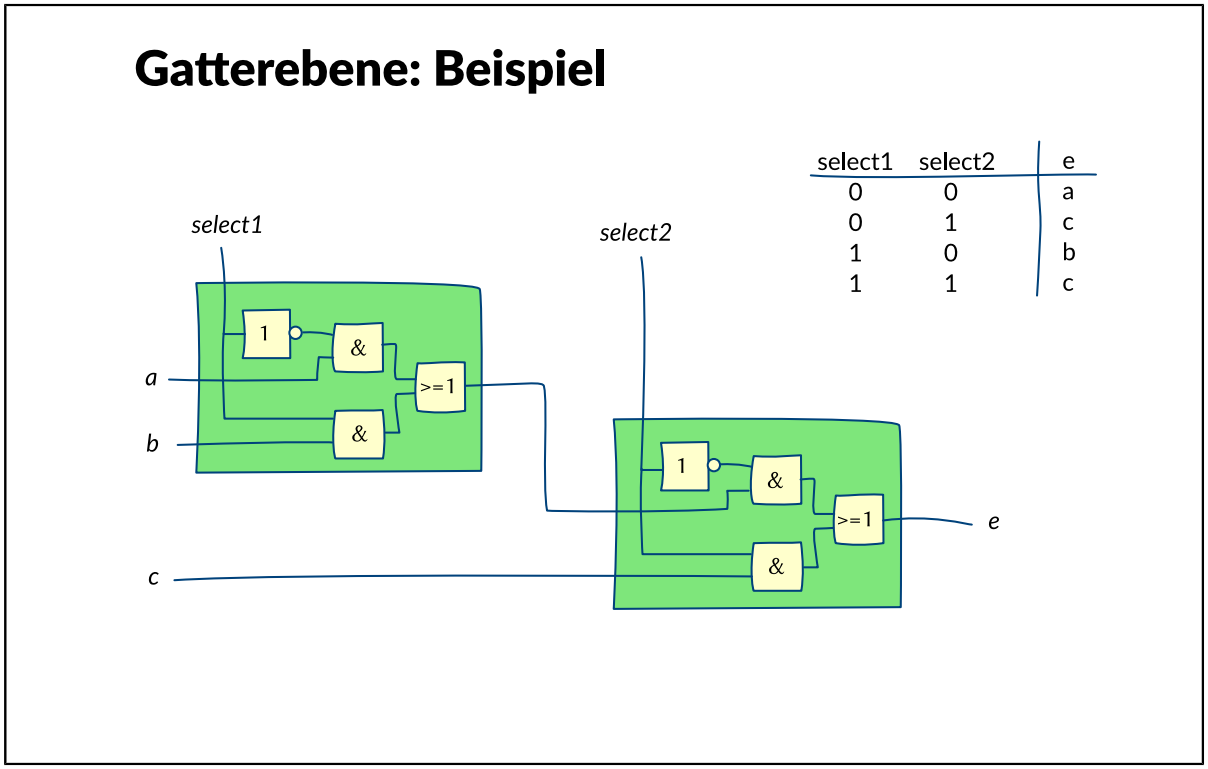
Dargestellt ist ein System bestehend aus zwei Bussen, zwei Registern, die je mit beiden Bussen bidirektional verbunden sind, und einer ALU, die von beiden Bussen gleichzeitig gelesen werden kann und ihr Ergebnis über eine dedizierte Verbindung in ein drittes Register schreibt. Dieses dritte Register kann auf beide Busse schreiben.

Entwurfsprozess: Gatterebene



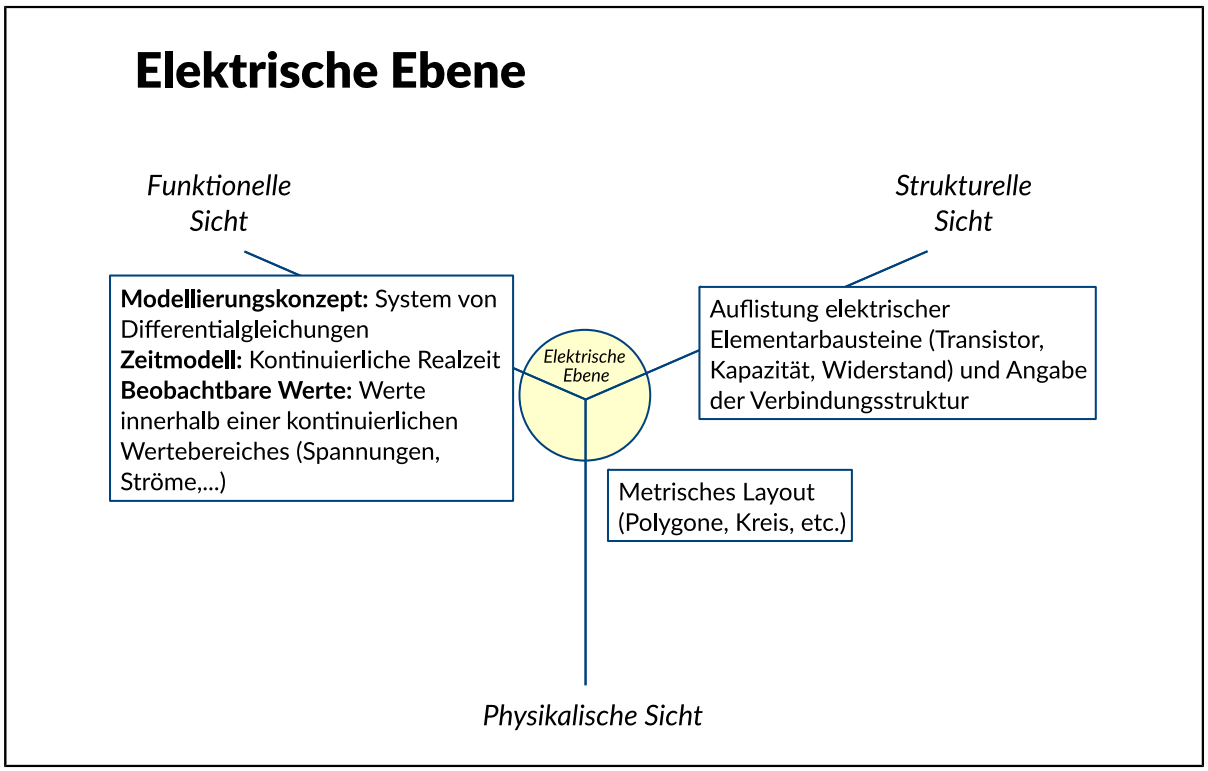
Beschreibungen auf der Gatterebene können als Erhöhung des Detailierungsgrads der Module auf der RT-Ebene angesehen werden. Allerdings geht die semantische Information über die Unterscheidung zwischen Daten- und Kontrollsignalen, wie sie auf der RT-Ebene noch vorhanden ist, vollständig verloren. Man hat nun lediglich ein Netz mit Gattern und Flipflops als Knoten und Einbit-Verbindungsleitungen als Kanten. Auf dieser Ebene ist man in vielen Fällen an präzisen Informationen über das Zeitverhalten interessiert. Daher ist das übliche Zeitmodell auf dieser Ebene das der kontinuierlichen Realzeit. Allerdings werden verschiedene approximative Verzögerungskonzepte benutzt, die von simplen Konzepten wie feste Nominalverzögerung bis hin zu Modellen reichen, die fast das analoge Verhalten der beteiligten Module widerspiegeln. Beobachtbare Werte sind "Bits" in einer mehrwertigen Logik. Die Knoten müssen nicht auf "Gatter" im engeren Sinn beschränkt sein, sondern sind Schaltungen, die durch einen booleschen Ausdruck bzw. durch ein Bündel boolescher Ausdrücke beschrieben werden können. Dadurch sind auch hierarchische Beschreibungen möglich. In der physikalischen Sicht stehen auf der Gatterebene häufig Bibliotheken mit so genannten Basiszellen zur Verfügung.

Entwurfsprozess: ... Beispiel



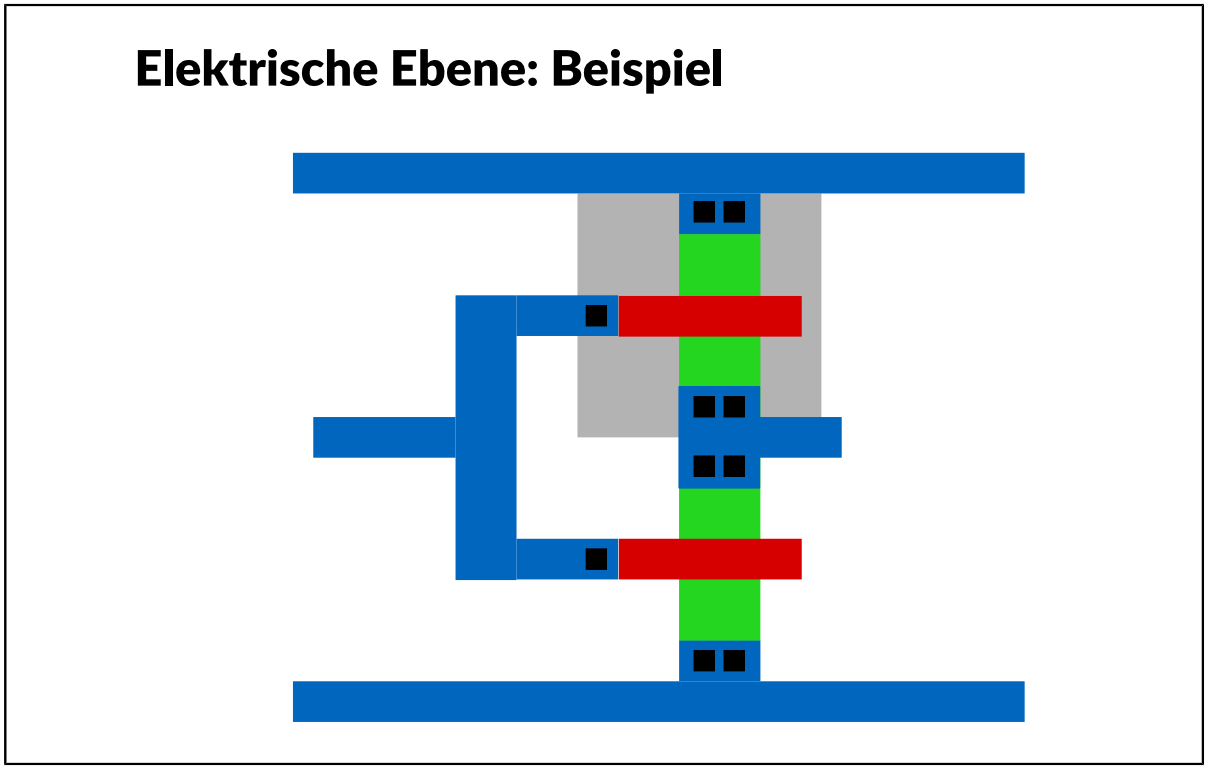
Die Folie zeigt einen 3-zu-1-Demultiplexer in einer möglichen Realisierung mit Standardgattern. Mit den beiden Eingangssignalen "select1" und "select2" kann aus den drei Eingängen der Schaltunge "a", "b" und "c" einer ausgewählt und auf den Ausgang "e" geschaltet werden. Gatterschaltungen können auch hierarchisch sein. Die beiden grün hinterlegten Gatteranordnungen könnten z.B. Instanzen eines 2-zu-1-Demultiplexers darstellen.

Entwurfsprozess: Elektrische Ebene



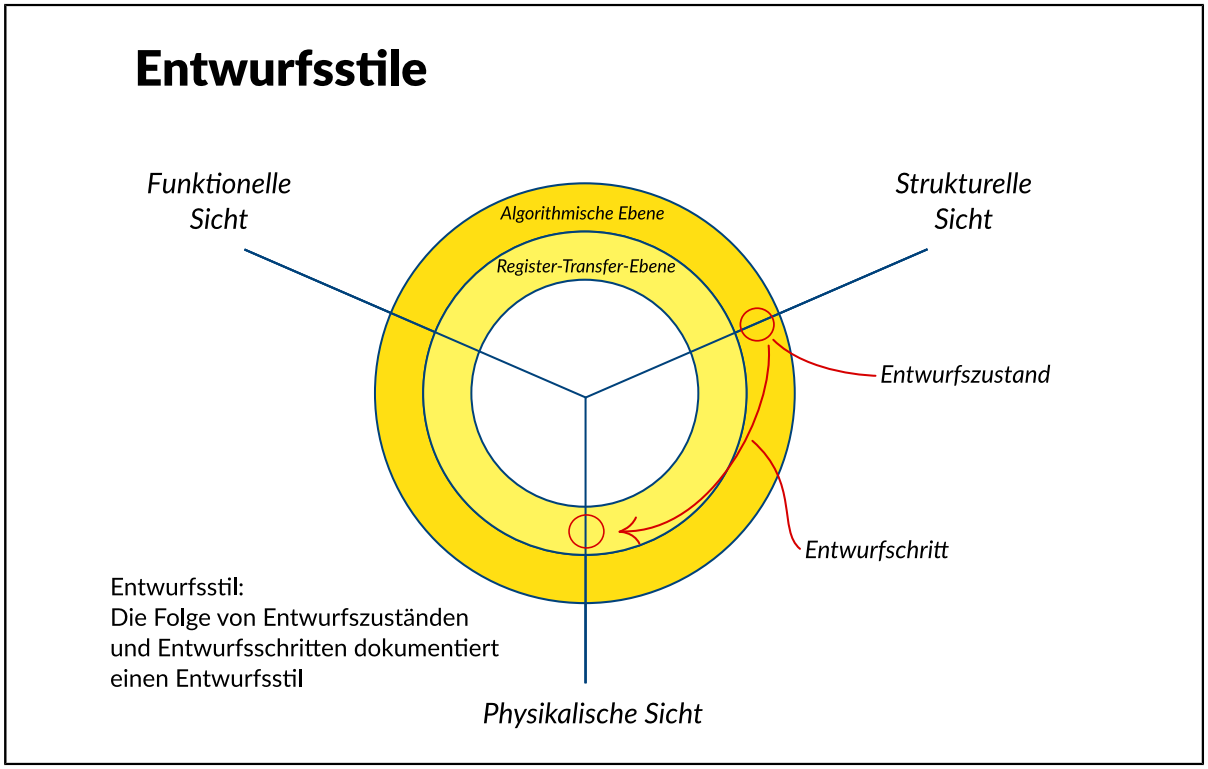
Auf dieser Ebene wird die digitale Interpretation der Schaltung aufgegeben und das analoge Verhalten betrachtet. Das Modellierungskonzept ist durch ein System von Differentialgleichungen über kontinuierlichen Wertebereichen und in kontinuierlichen Zeitbereichen gegeben. Die benutzten Elementarbausteine sind Widerstände, Kapazitäten, etc. Das metrische Layout unterscheidet sich vom symbolischen dadurch, dass jedes benutzte Objekt eine definierte Bemaßung hat.

Entwurfsprozess: ... Beispiel



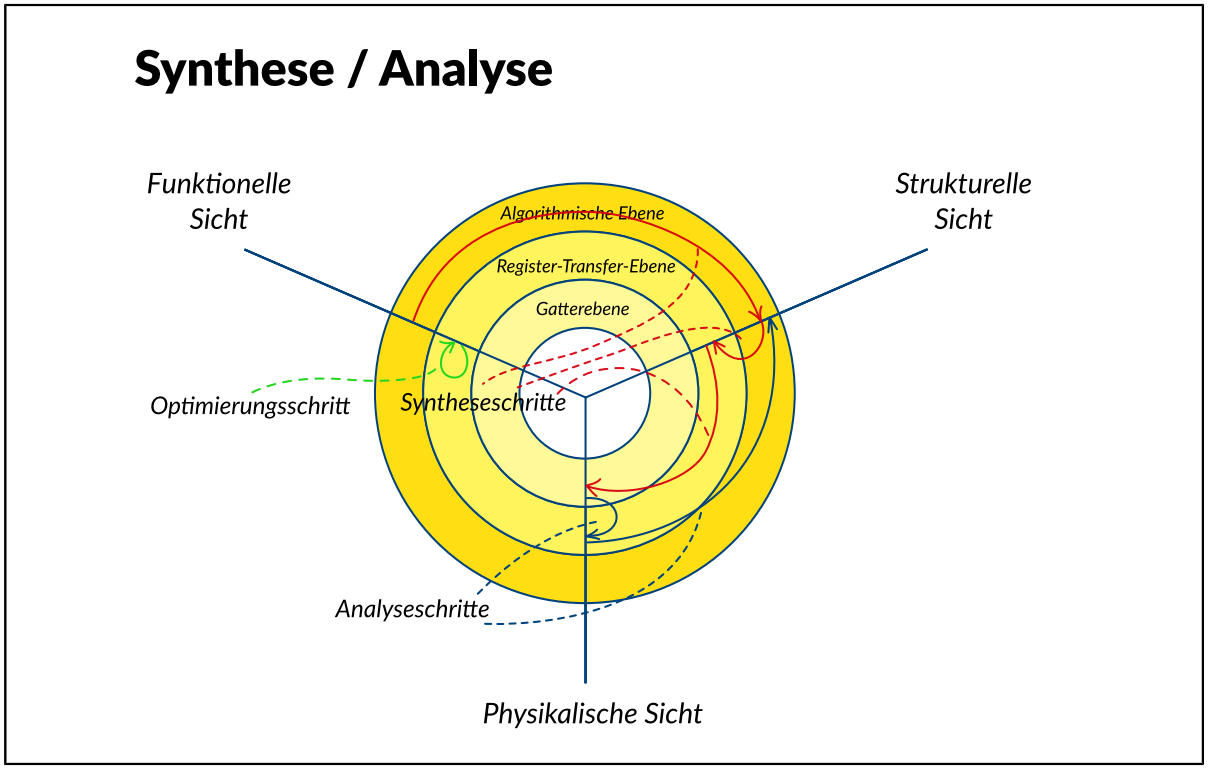
Das Beispiel zeigt die Layoutpolygone für einen CMOS-Inverter.

Entwurfsprozess: Entwurfsstile



Mit Hilfe des Y-Modells lassen sich beliebige Entwurfsstile darstellen. Jeder Entwurfszustand lässt sich durch einen Schnittpunkt zwischen einer der Achsen und einem der konzentrischen Kreise im Y-Modell repräsentieren. Ein Entwurfsschritt ist eine Abbildung zwischen zwei Entwurfszuständen. Ein Entwurfsstil ist durch eine Folge von Entwurfszuständen charakterisiert, die in der physikalischen Sicht auf der elektrischen Ebene endet. Der Ausgangspunkt des Entwurfsvorgangs, die Idee eines mikroelektronischen Systems, ist in diesem Modell gewöhnlich durch den Entwurfszustand "funktionelle Sicht - Systemebene" gekennzeichnet.

Entwurfsprozess: Synthese/Analyse



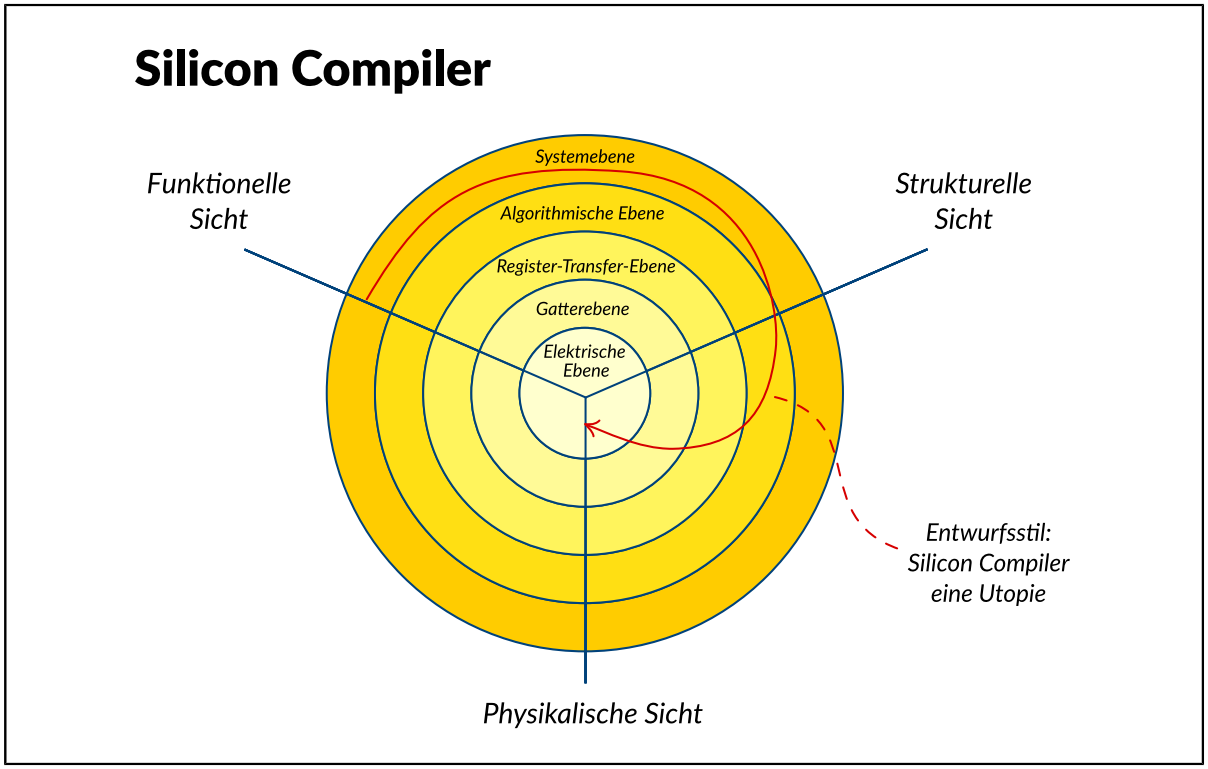
Betrachtet man einen Entwurfsschritt in Richtung auf das Entwurfsziel (ein metrisches Layout) hin, so wird die damit verbundene Abbildung als Syntheseschritt bezeichnet, betrachtet man die umgekehrte Richtung, so spricht man von einem Analyseschritt.

Ein Syntheseschritt führt das Entwurfsobjekt in einen Zustand, der der Realisierung näher liegt. Ein solcher Schritt ist gewöhnlich dadurch gekennzeichnet, dass der Abstraktionsgrad sinkt bzw. der Detailliertheitsgrad der Beschreibung steigt. Auf diese Weise wird in die Entwurfsbeschreibung neue Information eingebracht, die vorher nicht explizit vorhanden war. In diesem Sinne stellt ein Syntheseschritt einen kreativen Prozess dar. Die Disziplin der Entwurfsautomatisierung beschäftigt sich damit, Algorithmen zu entwickeln, die den kreativen Prozess der Synthese automatisieren.

Ein Analyseschritt vollzieht eine Abstraktion bzw. Extraktion. Aus einer gegebenen detaillierten Entwurfsbeschreibung werden abstrakte Informationen durch Zusammenfassen und Generalisieren von Details gewonnen. Solche Schritte dienen gewöhnlich zur Validierung eines Syntheseschritts. Dies ist insbesondere dann erforderlich, wenn ein Syntheseschritt manuell oder durch ein selbst nicht formal verifiziertes Verfahren durchgeführt wurde, so dass die "konstruktionsbedingte Korrektheit" (correctness by construction) nicht garantiert werden kann. Bei einem Analyseschritt wird immer ein Ebenenwechsel durchgeführt.

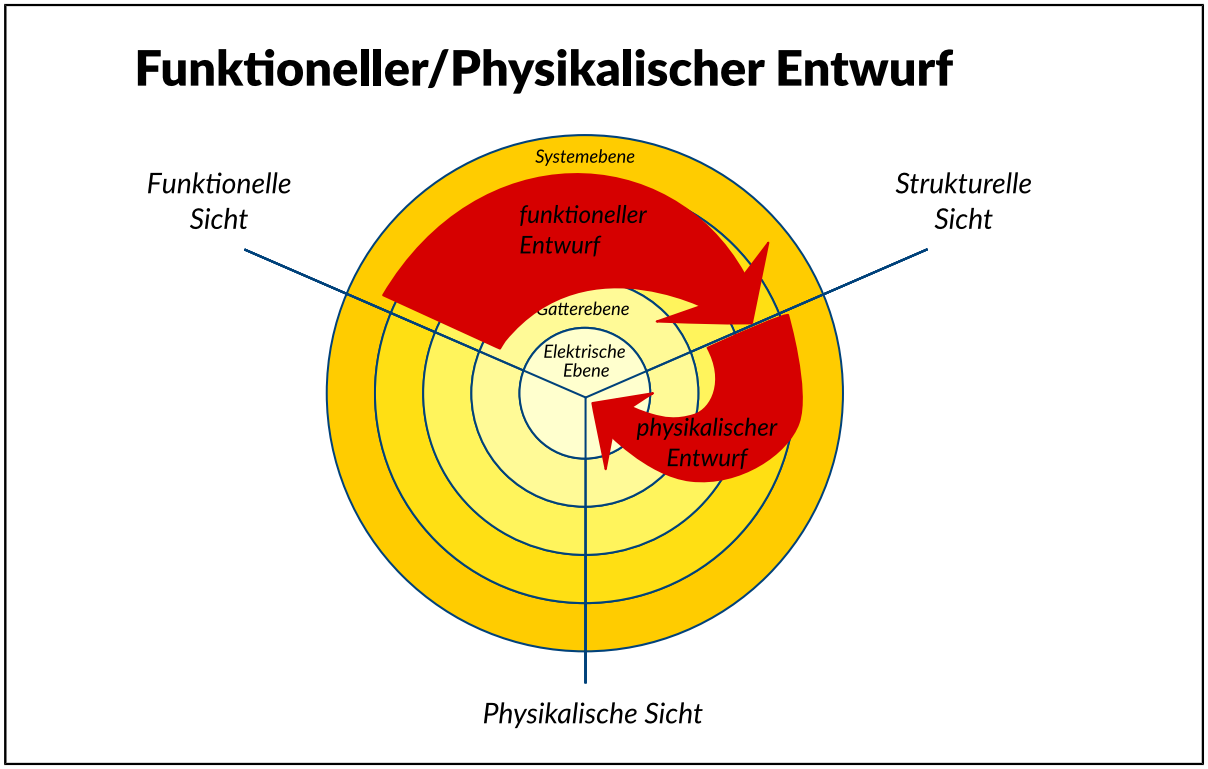
Ein Optimierungsschritt beeinhaltet keinen Ebenen- oder Sichtenwechsel. Der Entwurf wird in Hinblick auf ein (oder mehrere) Kriterien hin optimiert, um den in der Spezifikation angegebenen Anforderungen zu genügen.

Entwurfsprozess: Silicon Compiler



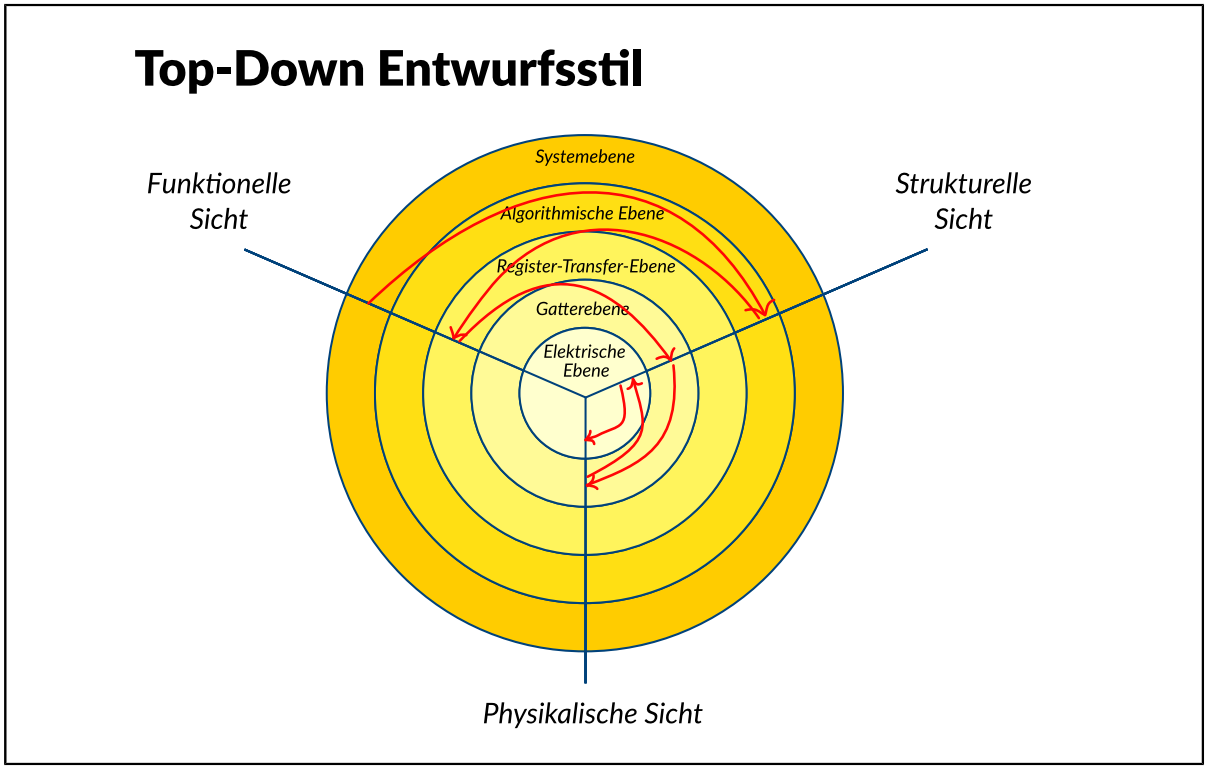
Die ideale Verwirklichung eines zeitoptimierten Entwurfsstils ist der vollautomatische Entwurf durch einen so genannten Silicon Compiler. Dieser erzeugt aus einer funktionellen Beschreibung (Spezifikation) in einer hohen Entwurfsebene automatisch ein metrisches Layout auf elektrischer Ebene. Obwohl der Begriff in der Literatur viel verwendet wurde und noch wird, erscheint seine Realisierung unter praktischen Randbedingungen nach wie vor utopisch.

Entwurfsprozess: Funktioneller/Physikalischer Entwurf



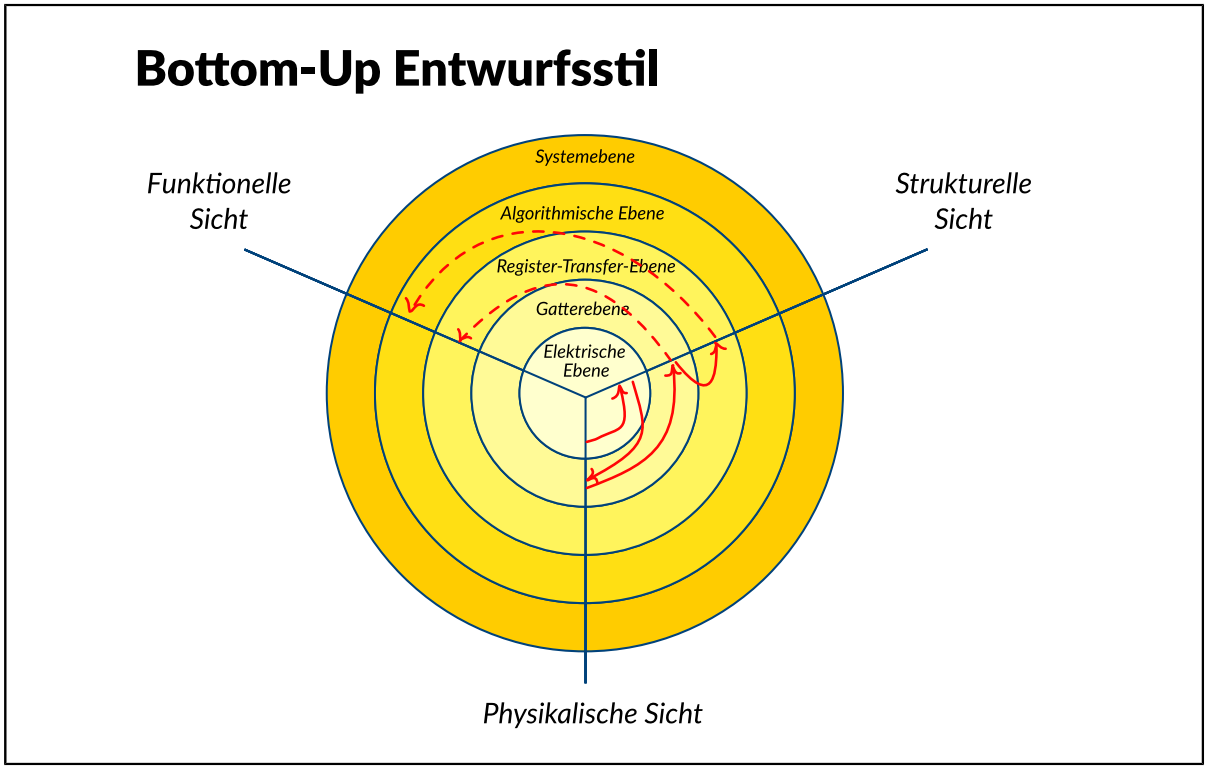
Stattdessen besteht heute ein Entwurf aus einer Vielzahl von Einzelschritten, mit denen der Weg von der Spezifikation bis zum metrischen Layout zurückgelegt wird. Wie bereits erwähnt ist dabei eine grobe Aufteilung zwischen funktionellen und physikalischen Entwurf sinnvoll. Dies kann ebenfalls in Y-Diagramm veranschaulicht werden.

Entwurfsprozess: Top Down Entwurfsstil



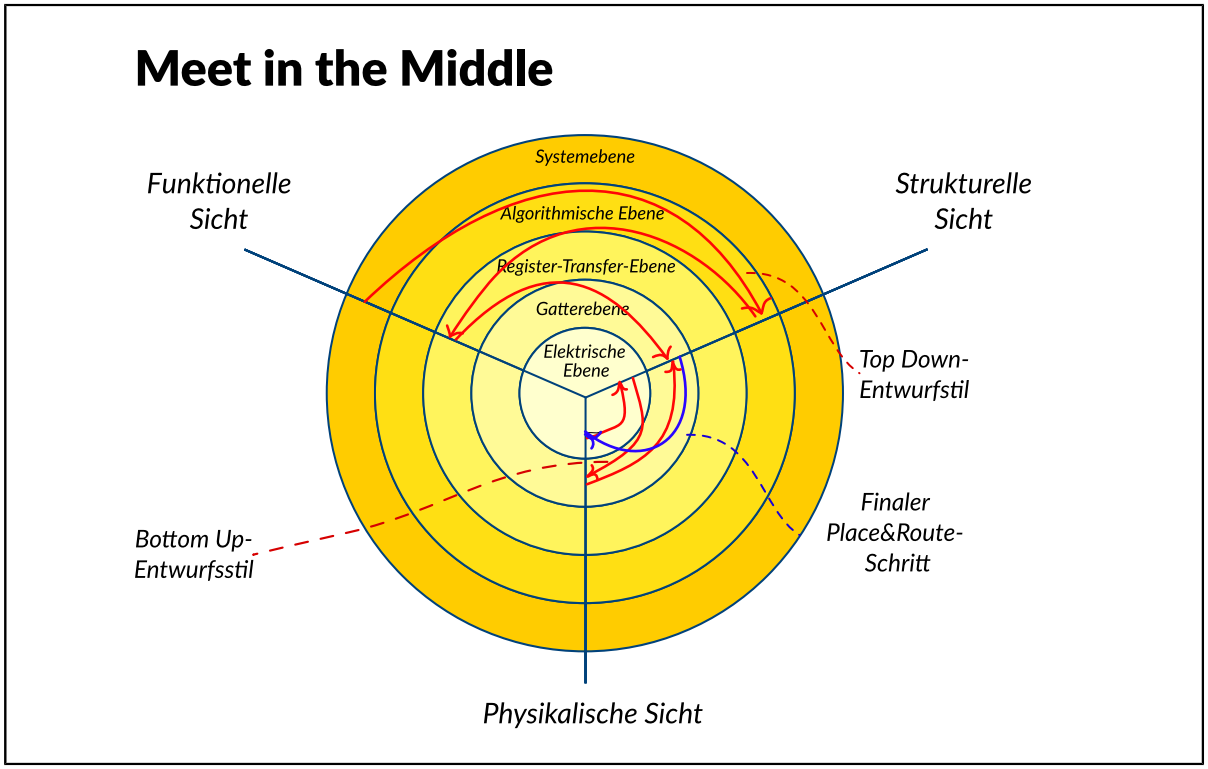
Wir haben bisher vorausgesetzt, dass der Entwurfsprozess auf hohen Entwurfsebenen beginnt und durch schrittweise Verfeinerung niedrigere Entwurfsebenen erreicht. Eine solche Vorgehensweise wird ganz allgemein als Top-Down-Entwurfsstil bezeichnet. Dieser Stil erscheint zur Beherrschung der Komplexität sinnvoll, begrenzt jedoch die Ausnutzung prinzipiell vorhandener Freiheitsgrade, da diese auf den mittleren Ebenen häufig nicht genutzt werden können, da Varianten nicht durchgepspielt werden können. Trotz allem wird der Top-Down Entwurfstil heute z.B. beim Analog-Entwurf sehr häufig noch eingesetzt.

Entwurfsprozess: Bottom Up Entwurfsstil



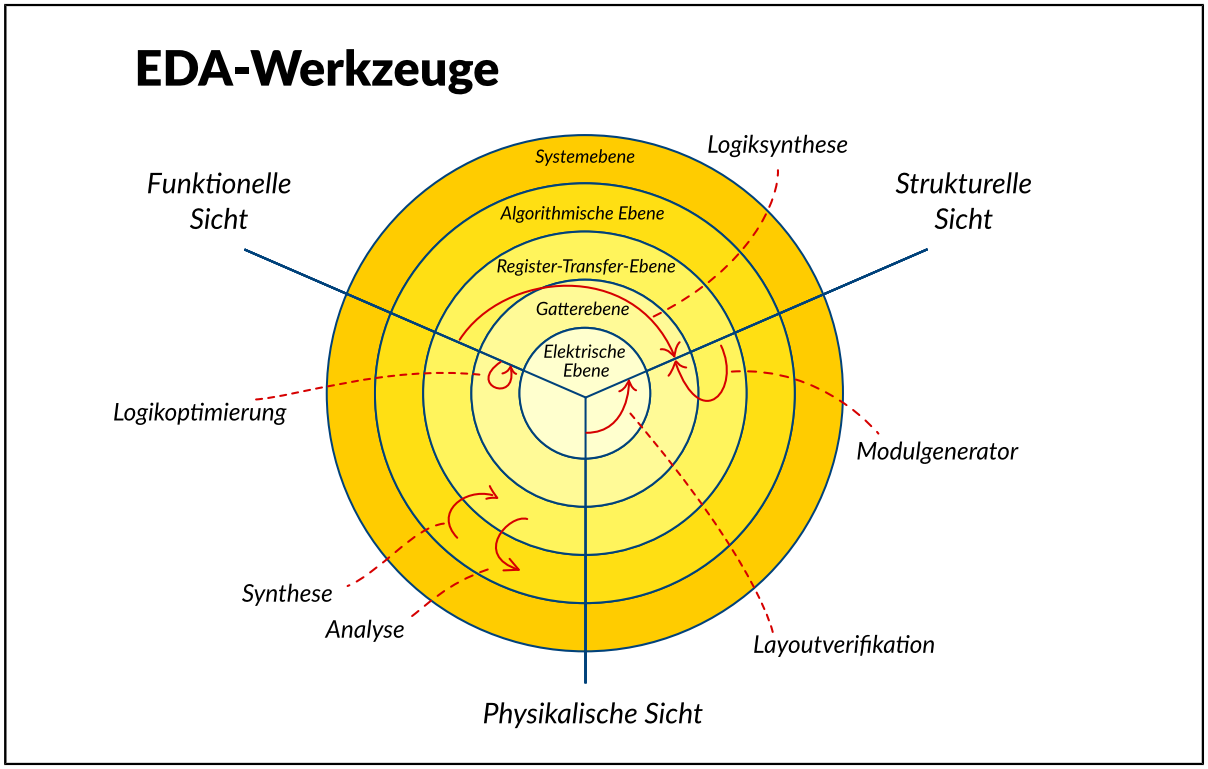
Historisch gesehen hat sich zunächst ein Bottom-Up-Entwurfsstil durchgesetzt, der, ausgehend von geometrischen Entwurfsregeln und elektrischen Transistoreigenschaften, aus Elementen niedriger Ebenen jeweils die Elemente der nächsthöheren Ebene zusammensetzt. Allerdings lässt sich dieses "aus kleineren Bausteinen größere zusammenbauen" nur bis zur Gatter-, maximal Register-Transfer-Ebene durchhalten. Danach ist die Zahl der möglichen Schaltungen einfach zu groß.

Entwurfsprozess: Meet in the Middle



Auf dem heutigen Stand der Technik ist eine "Meet-in-the-Middle"-Entwurfsstil üblich: Einerseits werden bei "Bottom-Up" die entsprechenden Gatter bis zu dieser Ebene entwickelt, andererseits wird der Weg "Top-Down" vom Funktionskonzept bis zur Gatter-Ebene eingeschlagen. Als Mitte lassen sich auch andere Ebenen heranziehen, beispielsweise die RT-Ebene. Nach dem beide Wege durchgeführt sind und eine Gatter(/RT-)netzliste vorliegt, muss jedoch immmer noch ein finaler Place and Route-Schritt durchgeführt werden.

Entwurfsprozess: EDA-Werkzeuge



Bevor wir uns den EDA-Werkzeugen im Einzelnen zuwenden, sollen hier anhand des Y-Diagramms noch einige allgemeine Definitionen vorgenommen werden. Dazu müssen wir uns die Tätigkeiten bei einem einzelnen Entwurfsschritt, also beim Übergang von einem Punkt des Diagramms zu nächsten, genauer ansehen.

Erzeugung (Synthese)

Aus den Dokumenten, die im Verlauf des Entwicklungsprozesses (z.B. Spezifikation, Netzliste, etc.) generiert wurden, muss das jeweils nächste erzeugt werden. Dies kann manuell, automatisch oder rechnergestützt geschehen. Ein Beispiel für automatisches Erzeugen sind Platzierungs- und Verdrahtungsprogramme, die aus Gatterschaltungen in Struktursicht Anordnungen von Zellen in physikalischer Sicht erzeugen.

Rechnergestützt wäre z.B. ein Platzierungs- und Verdrahtungsprogramm, das interaktive Eingriffe bzw. Vorplatzierung erlaubt, oder ein Programm, das nach Analyse einer algorithmischen Beschreibung gewichtete Vorschläge für eine Architektur macht, unter denen der Entwickler zu wählen hat.

Im Allgemeinen werden wir erzeugende EDA-Werkzeuge, bei denen sich gleichzeitig die Entwurfsebene und die Sicht (oder auch nur die Sicht) ändern als Synthesewerkzeuge und solche, bei denen sich nur die Entwurfsebene ändert, die Sicht jedoch gleich bleibt, als Generatoren bezeichnen. Ein Werkzeug, das aus booleschen Gleichungen eine Gatterschaltung erzeugt, ist demnach ein Logiksynthesewerkzeug, ein anderes Werkzeug, das aus einer komplexen Struktur, z.B. einem Multiplizierer eine Realisierung auf Gatterebene ableitet, ist ein (Modul-)Generator. Leider erfolgt die Benutzung dieser Begriffe nicht überall einheitlich. Insbesondere findet man bei Werkzeugen zur Optimierung von Gatterschaltungen oder bei der Generierung von Gatterschaltungen aus struktureller RTL(Register Transfer Level)-Sicht häufig - wenn auch fälschlicherweise - den Begriff "Logiksynthese".

Prüfen (Analyse)

Nur wenn der Entwurfsschritt automatisch gemacht wurde und das benutzte Hilfsmittel die Korrektheit des Ergebnisses garantiert, kann auf eine Überprüfung des Ergebnisses verzichtet werden. In allen anderen Fällen ist der Vorgang fehleranfällig, so dass eine Überprüfung zwingend notwendig ist.

Solche Prüfprogramme sind z.B. Design-Rule-Checker, die geometrische Regeln für Maskenlayouts überprüfen, Syntax-Checker für Funktionsbeschreibungen in höheren Beschreibungssprachen und vor allem Simulatoren für die verschiedenen Abstraktionsebenen. Andere Beispiele sind Netzlisten-Extraktoren zusammen mit Netzlisten-Vergleichern, die den Schritt von der Netzliste zum Maskenlayout (Layoutverifikation) überprüfen. Aber auch Programme, die Ergebnisse von Simulationen auf verschiedenen Abstraktionsebenen (intelligent) vergleichen, gehören dazu, ebenso wie die formale Verifikation. Häufig wird zwischen Validierung und Verifikation unterschieden. Unter Validierung versteht man das Einholen hinreichender Belege dafür, dass die Entwurfsziele erreicht wurden. Verifikation ist dagegen der Vergleich mit einem validierten (oder verifizierten) Standard, um die Einhaltung von Entwurfsregeln jeglicher Art zu beweisen.

Optimieren

Genügt der erreichte Punkt im Y-Diagramm nicht allen Anforderungen, wird man durch Optimierung versuchen, diesen Mangel zu beseitigen. Beispiele sind Kompaktierer, die die benötigte Fläche im Maskenlayout verringern sollen, Logik-Optimierer, welche beispielsweise die Anzahl der benötigten Gatter verringern sollen, aber auch Architektur-Modifikationen, welche die Performance erhöhen sollen.