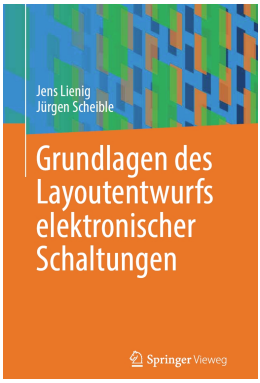
Jens Lienig, Jürgen Scheible
Grundlagen des Layoutentwurfs elektronischer Schaltungen
ISBN 978-3-031-15767-7, Springer Vieweg 2023



Kapitel 5
Layoutentwurf in Schritten: Von der Netzliste bis zum Layout-Postprozess

5.4 Verifikation

Nach dem Layoutentwurf ist das entstandene Layout möglichst vollständig zu verifizieren. Dabei werden die funktionale Korrektheit und die Herstellbarkeit des Entwurfs überprüft, also die korrekte Funktionalität des Entwurfs und das Vermeiden von Problemen während der Fertigung sichergestellt.

Der Entwurf eines elektronischen Systems ist ein aufwändiger und daher langwieriger und teurer Prozess. In diesem Prozess kann es zu Abweichungen kommen, die das Erreichen des Entwicklungszieles gefährden oder sogar unmöglich werden

lassen. Es gibt allgemeine Untersuchungen, die besagen, dass die Korrektur einer Abweichung mit jedem Entwurfsschritt, in dem sie unentdeckt bleibt, um eine Größenordnung aufwändiger und damit teurer wird. Man ist also bestrebt, Abweichungen möglichst frühzeitig zu erkennen, was eine Vielzahl von Verifikationsschritten erfordert.

Da das Endprodukt während des Entwurfsprozesses noch nicht vorliegt, und somit auch nicht getestet und bewertet („validiert“) werden kann, muss man verlässliche und aussagekräftige Kriterien ableiten, die eine solche Prüfung während des Entwurfsprozesses ermöglichen. Derartige Kriterien lassen sich aus den technologischen und funktionalen Randbedingungen ableiten. Diese Transformation zeigen wir in Abschn. 5.4.1. In den anschließenden Abschnitten gehen wir dann näher auf die einzelnen Verifikationsverfahren ein.

Wie in Abb. 5.31 dargestellt, umfasst ein vollständiger Verifikationsprozess beim Entwurf die folgenden Prüfungen: *formale Verifikation* (Abschn. 5.4.2), *funktionale Verifikation* (Abschn. 5.4.3), *Timing-Verifikation* (Abschn. 5.4.4) und *Layoutverifikation*. Die zuletzt genannte Layoutverifikation kann weiter unterteilt werden in die *geometrische Verifikation* (Abschn. 5.4.5) und die Verifikation basierend auf *Extraktion* und *Netzlistenvergleich* (LVS, Abschn. 5.4.6).

Da ein Layoutentwerfer auch die Verifikationsschritte vor der Layout-Generierung kennen sollte, behandeln wir zunächst diese dem Layoutentwurf vorgelagerten Verifikationen (formal, funktional und Timing), bevor wir mögliche Layoutverifikationen (DRC, ERC, Extraktion, LVS) im Detail vorstellen.

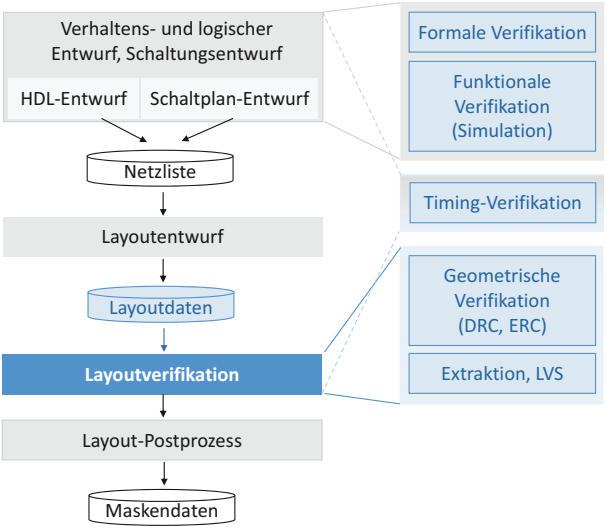


Abb. 5.31 Veranschaulichung der Einordnung der in den Abschn. 5.4.2 bis 5.4.6 behandelten Verifikationsschritte in den Entwurfsablauf

5.4.1 Grundlagen

Der Entwurf eines elektronischen Systems wird, wie wir bereits wissen, aufgrund der großen Komplexität dieser Aufgabe in mehrere Schritte aufgeteilt. Jeder Entwurfsschritt erzeugt ein neues Zwischenergebnis, mit dem man dem Entwurfsziel näherkommt. Diese Zwischenergebnisse sind es, die auf Abweichungen von den vorgegebenen funktionalen und technologischen Randbedingungen überprüft werden.

Die funktionalen Randbedingungen kommen aus dem jeweiligen Projekt. Zum Teil werden sie gleich zu Projektbeginn definiert (in der Spezifikation, z. B. der Rauschabstand eines Signals), zum Teil entstehen sie auch während des Entwurfsablaufs (z. B. ein maximal erlaubter Spannungsabfall auf einer Leitung). Die technologischen Randbedingungen werden vom Hersteller, also der Fab, geliefert; sie resultieren aus den Grenzwerten der eingesetzten Technologie.

Bereits in Kap. 2 (Abschn. 2.3.4) haben wir darauf hingewiesen, wie wichtig es ist, dass das Endergebnis eines IC-Entwurfs (d. h. das fertige Layout) korrekt ist, da ein gescheiterter Fertigungsversuch oder ein nichtfunktionaler Chip extreme Verluste mit sich bringt. Daher führt man die Überprüfungen auf Abweichungen mittels hochentwickelter rechnergestützter Verfahren durch. Damit das möglich ist, sind die zu prüfenden Randbedingungen allerdings vorher in ein für die Verifikationswerkzeuge lesbares und verarbeitbares Format zu übersetzen.

Diese „Übersetzung der Randbedingungen“ geschieht grundsätzlich durch eine zweifache Abbildung bzw. Konvertierung. Abb. 5.32 veranschaulicht das für alle auftretenden Randbedingungen. In der ersten Abbildung werden die aus der physikalischen Realität stammenden Randbedingungen zunächst als formale Regeln oder Vorgaben beschrieben, die in lesbarer Form, z. B. als Text, festgehalten werden. Diese formale Beschreibung der Randbedingungen stellt ein Metaformat dar (mittlere Säule in Abb. 5.32), aus dem sich dann in einer zweiten Abbildung die von den jeweiligen Verifikationsverfahren benötigten Datenformate erstellen lassen (rechte Säule in Abb. 5.32).

Nach dieser zweiten Abbildung stehen die Randbedingungen dann den Verifikationsverfahren im Entwurf (z. B. als Technologiedatei) zur Verfügung (rechte Säule in Abb. 5.32). Die Verifikationsverfahren (z. B. ein DRC) können hiermit das jeweilige Zwischenergebnis eines Entwurfsschrittes automatisch auf Einhaltung der Vorgabe überprüfen. Wird bei dieser Überprüfung eine Abweichung erkannt, so bezeichnen wir diese Verletzung als einen *Fehler* (*Error*).

Die Abb. 5.32 verdeutlicht auch, welche Rolle die unterschiedlichen Kategorien von Randbedingungen spielen. (Wir haben diese Kategorien in Kap. 4, Abschn. 4.5.2, eingeführt.) Die technologischen und funktionalen Randbedingungen sind es, die in die Layoutumgebung (rechte Spalte in Abb. 5.32) übersetzt und einer automatischen Überprüfung unterworfen werfen, da ihre Einhaltung über die Herstellbarkeit und die Funktionalität eines Chips oder einer Leiterplatte entscheidet. Die entwurfsmethodischen Randbedingungen haben die Aufgabe, diese Abbildung und damit eine automatische Verarbeitung überhaupt zu ermöglichen.

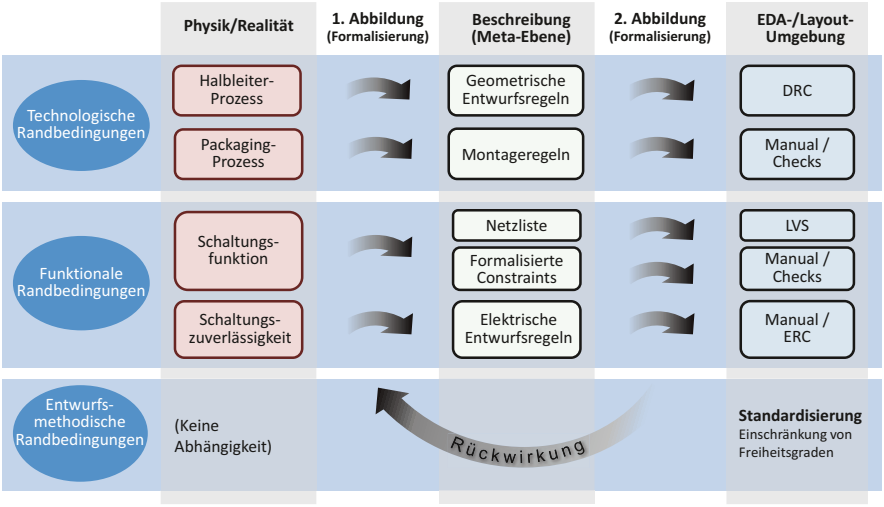


Abb. 5.32 Um technologische und funktionale Randbedingungen für Verifikationswerkzeuge zugänglich zu machen, sind zwei Abbildungen (Konvertierungen) erforderlich. Die erste Konvertierung führt zu formalen Regeln (mittlere Spalte), die dann in die Datenformate des jeweiligen Verifikationswerkzeugs umgewandelt werden (rechte Spalte). Entwurfsmethodische Randbedingungen stellen eine Rückkopplung von der zweiten zur ersten Abstraktion dar; sie begrenzen auch die Freiheitsgrade innerhalb des Layoutentwurfs

Hierzu ein Beispiel: Werkzeuge für den DRC benötigen die Beschreibung der Entwurfsregeln in einem ganz bestimmten werkzeugspezifischen Format. In diesem Format lassen sich komplexe Entwurfsregeln durch aus mehreren Kommandos bestehenden Routinen beschreiben (Beispiele finden sich in Kap. 3, Abschn. 3.4.3). Die Syntax und Semantik dieser Sprachen sind heute zwar sehr mächtig, bleibt prinzipiell aber immer beschränkt. Deshalb ist es notwendig, bereits die Regelbeschreibungen der Meta-Ebene so zu formulieren, dass diese zweite Abbildung in die EDA- bzw. Layoutumgebung ohne Informationsverlust möglich ist.

Entwurfsmethodische Randbedingungen sind damit die bei der Formulierung von Entwurfsregeln einzuhaltenden Restriktionen, damit diese Entwurfsregeln anschließend auch „programmierbar“ sind. Bezogen auf die Darstellung in Abb. 5.32 verkörpern entwurfsmethodische Randbedingungen eine Rückwirkung der zweiten Abbildung auf die erste Abbildung. Innerhalb des Layoutentwurfs (das Ergebnis der zweiten Abbildung in Abb. 5.32, d. h. die rechte Säule) fördern entwurfsmethodische Randbedingungen darüber hinaus die Standardisierung durch Einschränkung der Freiheitsgrade beim Entwurf.

Einem Entwickler sollte stets bewusst sein, dass beide Abbildungen eine Abstraktion darstellen, da die realen Randbedingungen (in Abb. 5.32 links dargestellt) bei diesen Zuordnungen formalisiert werden. Dies bedeutet, dass eine formalisierte Randbedingung, so wie sie beim Layoutentwurf erscheint, i. Allg. nicht deckungsgleich mit der realen, in der Technologie oder in der Schaltungstechnik auftretenden Anforderung sein muss. Wir wollen das an einem Beispiel aus dem Bereich der technologischen Randbedingungen verdeutlichen.

Aufgrund der hohen Komplexität moderner Technologien kommt es vor, dass eine technologische Randbedingung sich nicht exakt in eine Entwurfsregel abbilden lässt. In diesen Fällen bildet man die Entwurfsregel so ab, dass die reale Anforderung leicht übererfüllt wird. Dies führt dazu, dass Layout-Konstellationen im DRC als Fehler erkannt werden, obwohl sie nicht der realen technologischen Randbedingung widersprechen. In einem solchen Fall spricht man von einem *Dummyfehler*.

Dummyfehler können bei der Interpretation der DRC-Ergebnisse ignoriert werden, womit die entsprechende Layoutstruktur unverändert bleibt. Allerdings muss der Entwerfer sehr erfahren sein und die zugrundeliegende Technologie genau verstehen, um diese Fälle richtig zu behandeln.

Geometrische Entwurfsregeln im Layoutbereich (rechte Säule in Abb. 5.32) sind damit in der Regel konservativ formuliert, so dass die realen Anforderungen (d. h. die originalen technologischen Regeln) „sicher“ abgebildet sind. Technologische Randbedingungen werden im Ergebnis also tendenziell übererfüllt. Das bedeutet, dass man die Herstellbarkeit durch einen fehlerfreien DRC in der Regel zu 100 % sicherstellen kann. Im Gegensatz hierzu lassen sich funktionale Randbedingungen in heutigen Entwurfsumgebungen nur partiell abbilden. Ihre rechnergestützte Überprüfung ist also nicht vollständig möglich.

Dies ist eine wichtige Beobachtung, denn sie ist der Grund für weitere Überprüfungen, wie z. B. Simulationen, und auch ein Grund dafür, dass eine gültige *Verifikation* („Wurde die Schaltung richtig entworfen?“) trotzdem zu einer ungültigen *Validierung* („Wurde die richtige Schaltung entworfen?“) führen kann (Kap. 4, Abschn. 4.4, vgl. Abb. 4.18).

Bevor wir in den folgenden Unterkapiteln die verschiedenen Verifikationsmethoden im Layoutentwurf im Detail diskutieren, klassifizieren wir sie zunächst. Tab. 5.1 gibt einen Überblick über die verschiedenen Möglichkeiten zur Verifikation einer Schaltung. In dieser Tabelle ist auch das Testen erwähnt, bei dem ein Schaltungsentwurf im Hinblick auf die Anforderungen eines Kunden *validiert* wird (s. Abb. 4.18 in Kap. 4). Darauf gehen wir nachfolgend nicht weiter ein, da diese Validierung keinen direkten Bezug zum Layoutentwurf hat. Auch ignorieren wir hier die *Assertion-basierte Verifikation* (ABV), da sie auf höheren Abstraktionsebenen, wie z. B. der Spezifikation, angesiedelt ist.

Abb. 5.33 setzt die verschiedenen Verifikationsmethoden mit dem Y-Diagramm (Kap. 4, Abschn. 4.2.2) in Beziehung.

5.4.2 Formale Verifikation

Das Ziel der *formalen Verifikation* ist der Nachweis der Korrektheit einer Schaltungsimplementierung in Bezug auf ihre Spezifikation. Genauer gesagt untersucht sie die Korrektheit einer beabsichtigten Schaltung in Bezug auf eine bestimmte formale Spezifikation oder Eigenschaft mit Hilfe formaler mathematischer Methoden. Die bekanntesten formalen Verifikationsmethoden sind „Model Checking“ (Modellprüfung) – in kommerziellen Tools oft „Property Checking“ genannt – und „Equivalence Checking“ (Äquivalenzprüfung).

Tab. 5.1 Verschiedene Möglichkeiten zur Prüfung einer elektronischen Schaltung, die in den folgenden Unterabschnitten vorgestellt werden. Der Vollständigkeit halber schließen wir in dieser tabellarischen Auflistung auch das finale Testen mit ein, d. h. die Validierung eines Schaltungsentwurfs aus Sicht des Kunden

Prüfung	Was wird geprüft?	Wie wird geprüft?	Typ
Modellprüfung (Model check)	Logisches Merkmal (Vermutung ist wahr?)	Mathematische Modelle	Formale Verifikation
Äquivalenzprüfung	Logische Gleichwertigkeit von zwei Beschreibungen	Mathematische Modelle	Formale Verifikation
Simulation	Schaltungsverhalten versus Spezifikation	Virtuelles Experiment (Stimuli und Kennlinie)	Funktionale Verifikation
DRC (OPC, RET)	Layout versus technologische Randbedingungen (Herstellbarkeit)	Geometrische Entwurfsregeln	Geometrische Verifikation
LVS	Layout versus Schaltplan	Netzlistenextraktion aus dem Layout, regelbasiert	Geometrische Verifikation
PEX (plus Simulation)	Auswirkungen von Parasiten auf das Schaltungsverhalten	Parameterextraktion aus dem Layout, regelbasiert; gefolgt von Simulation	Geometrische und funktionale Verifikation
ERC	Layout vs. elektrische Prozessgrenzen (Zuverlässigkeit)	Extraktion der Konnektivität aus dem Layout, regelbasiert	Geometrische Verifikation
Testen	Eignung hinsichtlich Einsatzzweck	Reales Experiment, Kundenprüfung	Validierung

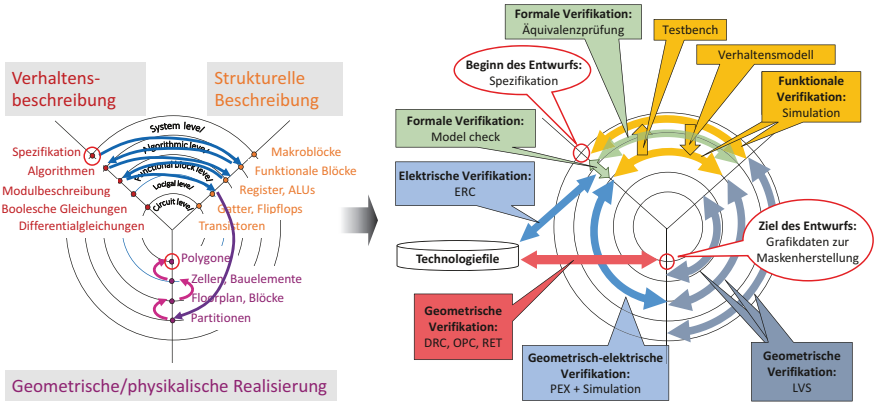


Abb. 5.33 Veranschaulichung der verschiedenen Verifikationsmethoden (vgl. Tab. 5.1) anhand des Y-Diagramms (rechts), in dem der Top-down-Entwurfsstil visualisiert wird (links)

Die Modellprüfung verifiziert eine bestimmte Eigenschaft eines Entwurfs oder einer Implementierung. Sie beweist (oder widerlegt), dass ein zu verifizierender Entwurf, der häufig in HDL-Code beschrieben wird, seine Spezifikationen erfüllt, d. h., dass er sich wie erwartet verhält. Sowohl das zu prüfende Entwurfsmodell als auch die Spezifikation werden in einer präzisen mathematischen Sprache formuliert. Im Wesentlichen muss eine gegebene Struktur bei dieser Prüfung eine bestimmte logische Formel erfüllen.

Die *Äquivalenzprüfung* hingegen vergleicht zwei Schaltungsbeschreibungen. Dabei wird erschöpfend geprüft, ob zwei Entwurfsdarstellungen, z. B. HDL-Code und eine abgeleitete Gatter-Netzliste, das gleiche funktionale Verhalten aufweisen. Es gibt verschiedene Ansätze zur Durchführung dieser Art von Beweis. Zum Beispiel können beide Schaltungsbeschreibungen durch eine normalisierte Notation, wie eine Netzlistensyntax, dargestellt werden, um den Vergleich zu vereinfachen. Die Äquivalenzprüfung ist die primäre Methode für die Syntheseüberprüfung.

Die formale Verifikation liefert entweder ein erfolgreiches Verifikationsergebnis oder zeigt, (i) dass die Schaltungsbeschreibung eine gewünschte Eigenschaft nicht erfüllt (Modellprüfung), oder (ii) dass zwei Schaltungsbeschreibungen nicht gleich sind (Äquivalenzprüfung).

Die formale Verifikation ist Teil der frühen Entwurfsschritte, wie z. B. der HDL-basierten Netzlistengenerierung, die wir in Abschn. 5.1 behandelt haben. Da sie vor dem eigentlichen Layoutentwurf zur Anwendung kommt, wird sie hier nicht weiter behandelt und stattdessen auf die Literatur verwiesen; z. B. enthält [10] eine leicht verständliche Einführung in das Thema.

5.4.3 Funktionale Verifikation: Simulation

Die funktionale Korrektheit einer Schaltung lässt sich durch eine Simulation nachweisen. Dabei wird anhand typischer Eingabemuster, sogenannter *Stimuli*, überprüft, ob die simulierten Ausgaben mit den beabsichtigten Ausgaben übereinstimmen. Alternativ kann man die Stimuli auch auf die Verhaltensbeschreibung des Entwurfs und auf die finale Gatterbeschreibung anwenden und dann ihre Reaktionen vergleichen und bewerten.

Etwaige Unterschiede in den Simulationsergebnissen können durch (i) Entwurfsfehler oder (ii) Simulationsfehler bzw. -ungenauigkeiten verursacht werden. In beiden Fällen sind weitere Untersuchungen erforderlich. Wenn jedoch die Simulationsergebnisse mit den Entwurfswerten übereinstimmen, wird das Vertrauen in die Korrektheit des Entwurfs gestärkt. Leider kann die Simulation nie garantieren, dass ein Entwurf in seiner Gesamtheit korrekt ist.

Die Simulation des Verhaltens eines Schaltkreises vor dem eigentlichen Aufbau kann die Entwurfseffizienz erheblich verbessern, indem sie Entwurfsfehler schon in einem frühen Stadium des Entwurfsflusses aufzeigt und Einblick in das Verhalten des Schaltkreises gewährt. Fast alle IC-Entwürfe stützen sich in hohem Maße auf die Simulation. Die bekanntesten Analogsimulatoren basieren auf dem Prinzip von

SPICE (Simulation Program with Integrated Circuit Emphasis) oder sind direkt davon abgeleitet; Digitalsimulatoren verwenden häufig Verilog- oder VHDL-Syntax (Abschn. 5.1).

Gängige Simulatoren enthalten sowohl analoge als auch ereignisgesteuerte digitale Simulationsfunktionen, sogenannte Mixed-Mode-Simulatoren. Das bedeutet, dass jede Simulation Komponenten enthalten kann, die analog oder ereignisgesteuert oder eine Kombination aus beidem sind. Mixed-Mode-Simulationen werden auf drei Ebenen durchgeführt: (i) mit einfachen digitalen Elementen, die Timing-Modelle und den integrierten digitalen Logiksimulator verwenden, (ii) mit Subcircuit-Modellen, die die tatsächliche Transistortopologie des ICs nutzen und (iii) mit Inline-Ausdrücken der booleschen Logik.

Simulationswerkzeuge haben in der Regel eine Schnittstelle zu einem Schaltplan-Editor, einer Simulations-Engine und einer Wellenformanzeige auf dem Bildschirm (Abb. 5.34). Diese Werkzeuge ermöglichen es dem Entwickler, eine simulierte Schaltung schnell zu ändern und zu sehen, wie sich die Änderungen auf die Ausgabe auswirken. Simulatoren enthalten in der Regel auch umfangreiche Modell- und Bauelementbibliotheken.

Bei der Prüfung einer Schaltung durch Simulation sollten wir immer bedenken, dass diese eine lange Ausführungszeit erfordert, insbesondere bei großen Designs. Auch fehlt in der Regel ein umfassender Satz von Stimuli, um den gesamten Entwurf zu validieren. Selbst bei schnellen Simulatoren und kleinen Schaltungen ist es unmöglich, alle denkbaren Eingabemuster und Schaltzustände zu berücksichtigen. (Beispielsweise würde die umfassende Simulation eines Multiplizierers für zwei 32-Bit-Binärzahlen 2^{64} Eingabemuster erfordern, was selbst bei einer Simulationsrate von 100 Millionen Multiplikationen pro Sekunde zu 5849 Jahren Ausführungszeit führen würde [11]). Dies zwingt den Entwickler oft dazu, sich auf eine begrenzte Anzahl zufällig erzeugter Stimuli zu verlassen, obwohl eine vollständige Abdeckung des Entwurfs wünschenswert wäre [5]. Folglich können einige Entwurfsfehler aufgrund „falscher Stimuli“ unentdeckt bleiben.[3]

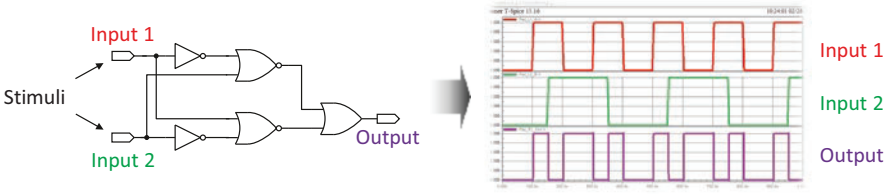


Abb. 5.34 Beispiel einer Schaltung mit XOR-Funktionalität, bestehend aus fünf Gattern, und ihrer (korrekten) Wellenformanzeige bei der Simulation

[3] Der sogenannte „Pentium-FDIV-Bug“ ist ein bekanntes Beispiel, bei dem ein gut simulierter Intel-Prozessor bei der Division einer Zahl falsche binäre Gleitkomma-Ergebnisse lieferte, was Intel einen Verlust von 475 Millionen Dollar einbrachte [12].

5.4.4 *Timing-Verifikation*

Der Begriff *Timing-Verifikation* wird üblicherweise für eine Prüfung verwendet, ob das Zeitverhalten (Timing) einer digitalen Schaltung noch gültig ist, nachdem das Layout vorliegt.

Kritische Pfade in einer Schaltung werden während der Logiksynthese berechnet, indem man alle Pfade auf Worst-Case-Verzögerungszeiten prüft, die durch wechselnde Signale verursacht werden. Der resultierende kritische Pfad definiert die schnellstmögliche Taktrate, bei der die Schaltung korrekte Ausgangssignale erzeugen kann. Auch das Layout der Schaltung muss die Zeitvorgaben einhalten. Im Wesentlichen muss die Schaltung zwei Timing-Prüfungen bestehen: die maximale Verzögerung, die sich auf die Setup-Bedingungen (längster Pfad) bezieht, und die minimale Verzögerung, die sich aus den Hold-Bedingungen (kürzester Pfad) ergibt (Abb. 5.35). Setup-Prüfungen charakterisieren die „Performance“ (Leistung) einer ausgelegten Schaltung, während eine nicht bestandene Hold-Prüfung auf eine fehlerhafte Schaltung hinweist.

Ein Ansatz für die Timing-Verifikation ist die *dynamische Timing-Analyse*. Dabei werden alle Leitungskapazitäten und -widerstände aus dem Layout extrahiert und die Schaltung unter Berücksichtigung dieser Werte simuliert. Diese Methode ist sehr zeitaufwändig, da viele Stimuli zu berücksichtigen sind. Die Beschränkung der dynamischen Timing-Analyse auf den kritischen Pfad (der während der Logiksynthese definiert wurde) ist nicht hilfreich, da dieser Pfad zu diesem späteren Zeitpunkt nicht mehr von Relevanz ist – schließlich kann die Layoutverdrahtung leicht einen anderen kritischen Pfad erzeugen, als während der Logiksynthese berechnet wurde.

Die statische Timing-Analyse (STA) wurde als effizientere Methode zur Timing-Verifizierung entwickelt. Sie basiert auf der aus dem Layout extrahierten Netzliste und berücksichtigt so auch Leitungskapazitäten und -widerstände. Die auf allen Pfaden berechneten Signalverzögerungen werden mit den vom Designer definierten Timing-Vorgaben verglichen. Die STA überträgt insbesondere die tatsächlichen Ankunftszeiten (AATs) und die erforderlichen Ankunftszeiten (RATs) auf die Pins für jedes Gatter oder jede Zelle. Die STA findet schnell Timing-Verletzungen

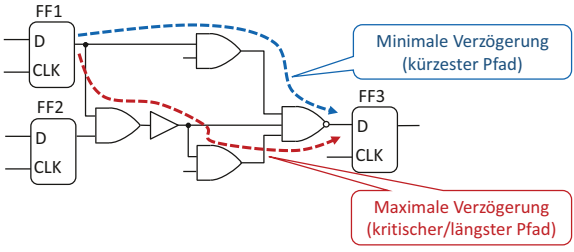


Abb. 5.35 Veranschaulichung der minimalen und maximalen Verzögerungen, die sich aus dem kürzesten und längsten Pfad ergeben

und diagnostiziert sie, indem sie kritische Pfade in der Schaltung aufspürt, die für diese Timing-Verletzungen verantwortlich sind [11]. In der Vergangenheit wurde die Logiksynthese dann unter Verwendung restriktiverer Timing-Vorgaben für diese kritischen Pfade wiederholt; heutzutage erzeugt die STA als Ausgabe eine optimierte Netzliste.

Die Verzögerungszeiten der logischen Gatter und der Leitungen sind Eingaben für die dynamische und statische Timing-Analyse. Während man die Gatterverzögerungen in den Timing-Modellen in der Bibliothek spezifiziert, sind die Signalverzögerungen der Leitungen mit Hilfe einer Vielzahl von Techniken zu berechnen. Unter diesen Techniken ist die momentbasierte Technik heute weit verbreitet, bei der die Impulsantworten des RLC-Netzwerks mit Hilfe von Zeit-Frequenz-Transformationsmethoden analysiert werden [5]. Bei einer alternativen momentbasierten Berechnung der Signalverzögerung der Leitung verwendet man das erste Moment der Impulsantwort; dies ist als *Elmore-Verzögerungsmodell* bekannt [13].

Jede zeitbezogene Schaltungssimulation *nach* der Layouterstellung benötigt layoutabhängige Timing-Informationen für den aktuell zu simulierenden Betriebszustand. Für diese Timing-Informationen wird das *Standard Delay Format* (*SDF*) verwendet. Die SDF-Datei enthält Interconnect delays (Leitungsverzögerung), Gate delays (Gatter-Verzögerungen) und Timing checks, die alle aus den Werkzeugen zum Layoutentwurf in ein abstrahiertes Format exportiert werden. Am wichtigsten sind dabei die Verzögerungen auf den Leitungen zwischen den Bauelementen (Interconnect delays), die sich aus der beim Layoutentwurf festgelegten konkreten Verdrahtung und damit deren bestimmbaren Längen ergeben.

Die Überprüfung des Timings erfordert auch eine Überprüfung auf resistive und kapazitive Kopplung. So tritt beispielsweise das sogenannte Übersprechen auf, wenn Signale in benachbarten Leitungen zwischen logischen Zuständen wechseln, und die kapazitive Kopplung zwischen ihnen einen Ladungstransfer verursacht [5]. Diese Kapazität hat auch einen erheblichen Einfluss auf die Signalverzögerung benachbarter Leitungen. Eine exakte Timing-Analyse ist daher wichtig, um die Signalverzögerungen auch nach der Layouterzeugung zu berechnen.

Verbale Ausdrücke wie *Timing-Closure* bezeichnen den Prozess der Erfüllung von Timing-Vorgaben durch Layout-Optimierungen und Netzlistenänderungen [4]. Zu diesen Layout-Optimierungen gehören die Timing-gesteuerte Platzierung und die Timing-gesteuerte Verdrahtung. Da sie für einen Layoutentwerfer von Bedeutung sind, wollen wir auf diese beiden Verfahren nachfolgend kurz eingehen.

Bei der Timing-gesteuerten Platzierung wird die Schaltungsverzögerung optimiert, um entweder alle Zeitvorgaben zu erfüllen oder die höchstmögliche Taktfrequenz zu erreichen. Sie nutzt die Ergebnisse von der STA, um kritische Netze zu identifizieren und versucht, die Signallaufzeiten durch diese Netze zu verbessern. Wie wir in Abschn. 5.3.2 ausführten, lässt sich die zeitgesteuerte Platzierung als netzbasiert oder pfadbasiert kategorisieren. Es gibt zwei Arten von netzbasierten Techniken – (i) Delay Budgeting weist dem Timing oder der Länge einzelner Netze Obergrenzen zu, und (ii) Netzgewichtung weist kritischen Netzen bei der Platzierung höhere Prioritäten zu [4]. Bei der pfadbasierten Platzierung wird versucht,

komplette zeitkritische Pfade zu verkürzen oder zu beschleunigen und nicht nur einzelne Netze. Obwohl sie genauer ist als die netzbasierte Platzierung, ist die pfadbasierte Platzierung nicht für große, moderne Entwürfe geeignet, da die Anzahl der Pfade in einigen Schaltungen, wie z. B. Multiplikatoren, exponentiell mit der Anzahl der Gatter wachsen kann [4].

Nach der detaillierten Platzierung, der Auslegung des Taktnetzwerks und seiner Optimierung zielt die *Timing-gesteuerte Verdrahtung* darauf ab, die verbleibenden Timing-Verletzungen zu korrigieren. Dabei wird versucht, eine oder beide der folgenden Größen zu minimieren: (i) die maximal auftretende Verzögerung der Netze und (ii) die Gesamtverdrahtungslänge [4]. Zu den Methoden der Timing-gesteuerten Verdrahtung gehören weiterhin das Erzeugen von Bäumen mit minimalen Kosten und minimalem Radius für kritische Netze sowie die Minimierung der Source-to-sink-Verzögerung kritischer Anschlüsse [4].

Wenn immer noch Zeitüberschreitungen auftreten, werden weitere Optimierungen, wie z. B. eine erneute Pufferung, vorgenommen.

5.4.5 Geometrische Verifikation: DRC, ERC

Unter dem Begriff *geometrische Verifikation* fasst man alle Prüfungen zusammen, die am fertigen (geometrischen) Layout oder während des Layoutentwurfs durchgeführt werden. Zu nennen sind hier vor allem der Design Rule Check (DRC) und der Electrical Rule Check (ERC).

Jeder Chiphersteller stellt geometrische Entwurfsregeln (Kap. 3, Abschn. 3.4) für seine Technologie dem Layoutentwerfer zur Verfügung (vgl. Abb. 4.19 in Kap. 4). Sie werden in einer *Technologiedatei* gespeichert, die Teil der Design-Suite für eine bestimmte Technologie ist (*Process design kit*, *PDK*). Letztlich dienen die Regeln der Erstellung von Fotomasken, damit diese ein fertigungsgerechtes Layout abbilden können. Genauer gesagt legen die Entwurfsregeln layoutspezifische Grenzwerte fest, um ausreichende Spielräume für die Variabilität des angewandten Halbleiterherstellungsprozesses zu gewährleisten.

Wie bereits erwähnt (Kap. 3, Abschn. 3.4.2 und Kap. 4, Abschn. 4.5.2), lassen sich geometrische Entwurfsregeln in Breiten-, Abstands-, Überhang-, Überlappungs- und Umschließungsregeln unterteilen (Abb. 5.36, links; vgl. Abb. 3.20 in Kap. 3). Eine weitere Kategorie von Regeln, die bei der geometrischen Verifikation überprüft werden können, sind die Antennenregeln (Abb. 5.36, rechts).

Der Design-Rule-Checker verwendet während des Verifikationsprozesses die oben erwähnte Technologiedatei (auch als *DRC-Deck* bezeichnet); die Layoutdaten werden normalerweise im GDSII/OASIS-Standardformat bereitgestellt. Der DRC hat sich von einfachen Messungen und booleschen Prüfungen zur Verifikation hochkomplexer Regeln entwickelt, welche bestehende Strukturen modifizieren, neue einfügen und das gesamte Design auf Prozessvorgaben, wie z. B. die Schichtdichte, prüfen. Moderne Design Rule Checker führen vollständige Prüfungen von geometrischen Entwurfsregeln durch (Kap. 3, Abschn. 3.4). Der DRC markiert Verstöße

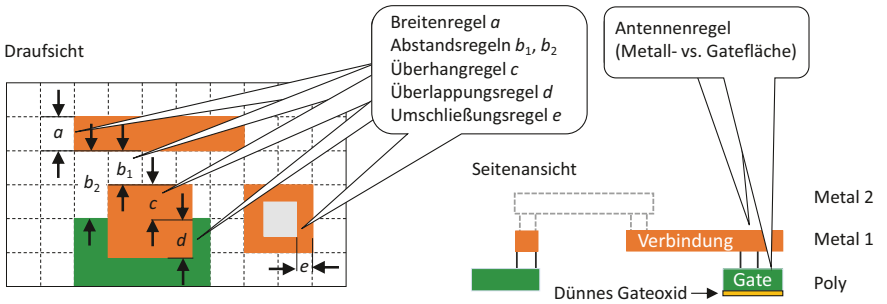


Abb. 5.36 Visualisierung der grundlegenden DRC-Prüfungen (Breiten-, Abstands-, Überhang-, Überlappungs- und Umschließungsregeln, vgl. Abb. 3.20 in Kap. 3) und der Antennenregel, die das zulässige Verhältnis von Poly- oder Metallfläche zur Gatefläche angibt (rechts)

entweder direkt im Layout (Abb. 5.37, links) oder das Programm listet die Entwurfsregelverletzungen auf.

In einigen speziellen Fällen verzichten Entwickler bewusst darauf, DRC-Verletzungen zu korrigieren. Dabei wird durch „vorsichtiges Tolerieren" derartiger Verletzungen versucht, z. B. die Leistung oder Packungsdichte auf Kosten der Ausbeute zu erhöhen. Je konservativer die Entwurfsregeln sind, desto wahrscheinlicher ist es, dass der Entwurf dennoch korrekt hergestellt wird; die Leistung und andere Ziele können jedoch unter derartigen „konservativen Entwurfsregeln" leiden.

Wie bereits erwähnt, lassen sich *Antennenregeln* in den DRC einbeziehen. Eine sogenannte *Antenne* ist eine Verbindung, d. h. ein Leiter aus Polysilizium oder Metall, der während des Herstellungsprozesses noch nicht beidseitig angeschlossen ist. Da die darüber liegenden Schichten noch nicht verarbeitet sind, ist

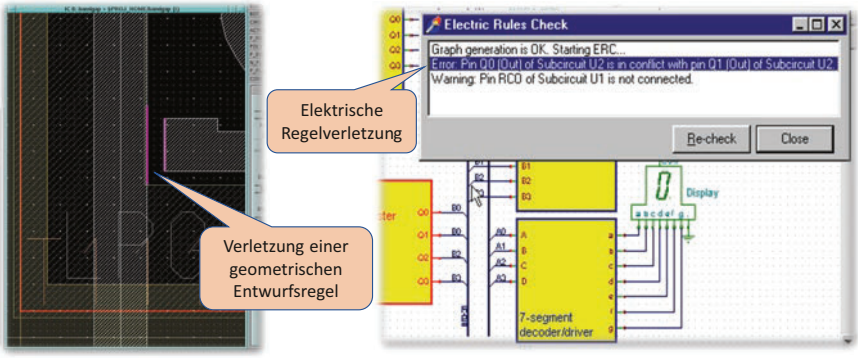


Abb. 5.37 Die Entwurfsregelprüfung (DRC) prüft die Einhaltung geometrischer Entwurfsregeln und zeigt hier eine Verletzung des Mindestabstands an (links). Im Gegensatz dazu prüft der Electrical Rule Check (ERC, rechts) auf Inkonsistenzen im elektrischen Netzwerk, die sich aus der Geometrie und der Konnektivität im Schaltplan oder Layout ableiten lassen. Vereinfacht gesagt, führt DRC eine Syntaxanalyse des Layouts und ERC eine Syntaxanalyse des Netzwerks durch

dieser Leiter vorübergehend nicht elektrisch mit dem Silizium verbunden oder geerdet (s. Abb. 5.36, rechts). Auf diesem temporären Verbindungsstück können sich während des Herstellungsprozesses Ladungen so weit ansammeln, dass Leckströme entstehen, die dann physische Schäden am einseitig angeschlossenen, dünnen Transistor-Gate-Oxid verursachen. Dieses zerstörerische Phänomen ist als *Antenneneffekt* bekannt. Die Hersteller geben normalerweise Antennenregeln vor, die oft als zulässiges Verhältnis von Polysilizium- und Metallfläche zur Gatefläche ausgedrückt werden. Dieses Verhältnis, vorgegeben für jede Metalllage, lässt sich dann vom DRC überprüfen. Stellenweise wird zusätzlich ein bestimmtes Verhältnis zwischen dem Umfang der Polysilizium- und Metallformen und der Gate-Fläche geprüft, da sich die Ladungen vorzugsweise an den Strukturkanten sammeln.

Da heutige Probleme bei der Herstellbarkeit eines Layouts oft über grundlegende geometrische Entwurfsregeln hinausgehen, enthalten moderne DRC-Werkzeuge weitere Prüfroutinen. Dazu gehören die schon in Kap. 3 behandelten booleschen Operationen und Größenfunktionen (Sizing) bei derartigen Regeln; auch Nachbarschaftsbeziehungen zwischen verschiedenen Lagen lassen sich in eine automatische Überprüfung einbeziehen. Auch diese Regeln werden direkt vom IC-Hersteller bereitgestellt.

Ein DRC kann extrem laufzeitintensiv sein, da die Prüfungen in der Regel für jeden Teilbereich der Schaltung durchzuführen sind, um die Anzahl der Fehler zu minimieren, die auf der obersten Ebene entdeckt werden. Moderne Designs können DRC-Laufzeiten von bis zu einer Woche aufweisen. Viele Entwurfsfirmen streben dennoch DRC-Zeiten von weniger als einen Tag an, denn nur so lassen sich vertretbare Zykluszeiten erreichen, da der DRC vor der Fertigstellung des Entwurfs in den meisten Fällen mehrmals auszuführen ist.

Die bisherigen Ausführen zeigen, dass der DRC sicherstellt, dass die Schaltung korrekt *herstellbar* ist. Es sollte auch klar sein, dass sich die korrekte *Funktionalität* auf diese Weise nicht überprüfen lässt; dies wird ausschließlich durch die Verhaltensprüfung der Schaltung sichergestellt (formale, funktionale und Timing-Verifikation), die wir bereits in den Abschn. 5.4.2, 5.4.3 und 5.4.4 behandelt haben.

Untersuchen wir nun den „Mittelweg“ zwischen einfacher Layout-Prüfung und komplexer Verhaltensprüfung, der die Domäne der *elektrischen Regelprüfer (ERC)* ist. Elektrische (Entwurfs-)Regeln gestalten eine Schaltung robuster, indem sie sie z. B. vor Schäden durch elektrostatische Entladungen schützen; sie verbessern auch ihre Zuverlässigkeit, indem sie die Alterung aufgrund von elektrischer Überlastung verringern. Diese Regeln sind in hohem Maße abhängig von (i) der angewandten Halbleitertechnologie, (ii) dem Schaltungstyp und (iii) der zukünftigen Verwendung der Schaltung als Komponente in einer Systemumgebung. Darüber hinaus werden die elektrischen Regeln häufig durch entwurfsspezifische Regeln und erfahrungsbasierte Regeln („Expertenwissen“) ergänzt.

Das Electrical-Rule-Checking ist also eine Methode, mit der sich die Robustheit und Zuverlässigkeit eines Entwurfs sowohl auf Schaltplan- als auch auf Layout-Ebene anhand verschiedener „elektrischer Entwurfsregeln“ überprüfen lässt. Dabei

wird die Korrektheit der Stromversorgungs- und Masseverbindungen überprüft und auf „schwebende“ Netze oder Pins sowie offene und kurzgeschlossene Netze kontrolliert. Indem man beispielsweise die Stromversorgungs-, Masse-, Eingangs- und Taktsignale durch den Schaltplan und/oder das Layout der Schaltung verfolgt, ist es möglich, auf falsche Ausgangstreiber, Inkonsistenzen in den Signalspezifikationen, nicht angeschlossene Schaltungselemente und vieles mehr zu prüfen. Die Ergebnisse werden entweder innerhalb des Schaltplan-/Layout-Editors visualisiert oder in einer Tabelle dargestellt (s. Abb. 5.37, rechts).

Elektrische Regeln spezifiziert man oft anhand topologischer Strukturen und nicht als einzelne Bauelement-/Pin-Prüfungen. Geometrische Regeln aus dem Layout werden ebenfalls mit diesen Topologien verknüpft, um eine ordnungsgemäße Funktionalität, Leistung und Ausbeute zu gewährleisten. Einige Regeln, wie z. B. spannungsabhängige Metallabstandsregeln, kombinieren sowohl geometrische als auch elektrische Prüfungen.

5.4.6 Extraktion und LVS

Die *Layout-versus-schematic-Prüfung*, oft abgekürzt als *LVS-Check* (auch: Layout versus Schaltplan), vergleicht die ursprüngliche Netzliste, die zur Erstellung des Layouts verwendet wurde, mit einer durch *Extraktion* aus dem erstellten Layout gewonnenen Netzliste. Damit wird nachgewiesen, dass das generierte Layout exakt mit der Originalnetzliste übereinstimmt. Genauer gesagt stellt die LVS-Prüfung sicher, dass Schaltungs- und Layoutentwurf übereinstimmen, indem sie (i) die elektrischen Verbindungen zwischen Bauelementinstanzen, (ii) die korrekten Bauelemente bzw. Zellen in der Netzliste und im Layout und (iii) funktionskritische Parameter der Bauelemente überprüft. Das LVS und der DRC sind oftmals die wichtigsten Verifikationswerkzeuge in einem IC-Entwurfsfluss.

Um beide Netzlisten vergleichen zu können, muss das LVS-Tool zunächst eine Netzliste aus den Layoutdaten *extrahieren*. Zu diesem Extraktionsschritt wird eine technologieabhängige Extraktionsdatei benötigt, die drei Definitionen enthält:

- Wie sind die Metalllagen miteinander verbunden, d. h. wie lässt sich mittels dieser Durchkontaktierungen ein *Netz* erkennen?
- Welche Kombination von Polygonen und Ebenen definiert ein *Bauelement*?
- Welche Eigenschaften der Polygone eines Bauelements bestimmen dessen elektrische *Parameter*?

Abb. 5.38 veranschaulicht den Inhalt einer solchen Extraktionsdatei. Der Inhalt einer Netzliste kann nur mit diesen drei Informationen (Verbindungen zwischen den

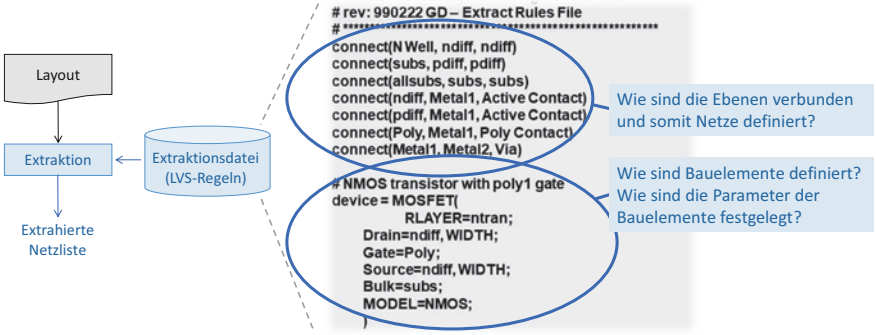


Abb. 5.38 Um die Netzliste aus den Layout-Polygonen zu extrahieren, wird eine Extraktionsdatei benötigt. Nur mit dieser kann das Extraktionswerkzeug „wissen", welche Polygonkonfiguration eine Durchkontaktierung (und damit ein Netz) oder ein Bauelement bildet und wie die Polygone eines Bauelements dessen elektrische Parameter bestimmen

Lagen, Bauelemente, Parameter der Bauelemente) aus einem gegebenen Layout abgeleitet werden, da das Layout ja nur aus Polygonen besteht.[4]

Der Extraktionsalgorithmus ist in der Lage, auf der Grundlage dieser Beschreibung eine Netzliste aus den Grafikdaten des Layouts zu erzeugen. Die Vorgehensweise ist wie folgt:

(1) Bestimmung der Bauelemente (Transistoren):
   (a) Bestimmen aller geometrischen Strukturen, welche die Bauelemente (Transistoren) repräsentieren.
   (b) Trennen der Bauelemente von den übrigen Layoutstrukturen.

(2) Bestimmen der elektrischen Knoten (Netze):
   Bestimmen aller geometrischen Strukturen, die elektrisch verbundene Einheiten bilden. Dies ist eine Maskenebene-übergreifende Operation.

(3) Generieren der Netzliste:
   (a) Bestimmen der Knoten (Netznamen), zu denen die mit den Bauelementen verbundenen geometrischen Strukturen gehören.
   (b) Zuweisen der Anschlusstypen (z. B. Gate und Source bei Transistoren).

Der Inhalt dieser Netzliste wird dann mit einer aus dem Schaltplan abgeleiteten Netzliste verglichen. Das gesamte LVS-Verfahren ist in Abb. 5.39 veranschaulicht.

[4] Es ist wichtig zu wissen, warum wir keine anderen Layout-Informationen, wie z. B. Bibliotheksinformationen, einbeziehen, schließlich würde dies die Aufgabe erheblich vereinfachen und die Netzlistenerkennung beschleunigen. Allerdings würde dann auch ein Fehler in der Bibliothek berücksichtigt werden – und die abschließende Netzlistenprüfung damit trotz des Fehlers identische Netzlisten ermitteln, da beide Listen von demselben bibliotheksbasierten Fehler betroffen wären. Dies würde das LVS unbrauchbar machen.

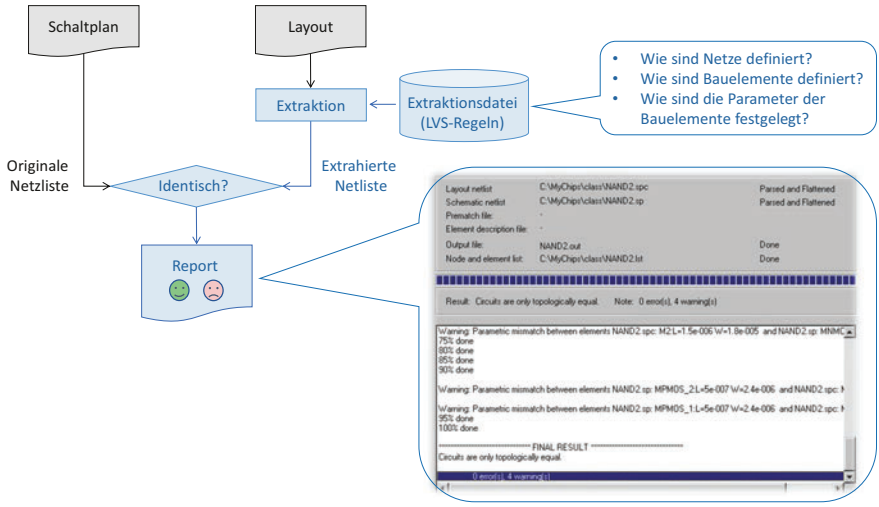


Abb. 5.39 Die Prüfung einer Schaltung mittels LVS basiert auf einer Netzlistenextraktion aus dem Layout. Diese extrahierte Netzliste wird mit der originalen Netzliste verglichen, die zur Erstellung des Layouts verwendet wurde

Der LVS vergleicht die Ausgabedaten (Layout) mit den Eingabedaten (Schaltplan) hinsichtlich der folgenden drei Schaltplanattribute:

- Netze: Sind alle elektrischen Verbindungen im Schaltplan – und nur diese – auch im Layout enthalten?
- Bauelemente: Sind alle Bauelemente aus dem Schaltplan – und nur diese – im Layout vorhanden?
- Parameter der Bauelemente: Haben alle Bauelemente im Layout die im Schaltplan angegebenen elektrischen Parameter?

Das Ergebnis des LVS ist eine Reportdatei (s. Abb. 5.39), welche die Anzahl und die Typen der Bauelemente sowie die Knoten in der ursprünglichen Netzliste (aus dem Schaltplan) und in der aus dem Layout extrahierten Netzliste enthält. In dieser Datei sind auch alle nicht übereinstimmenden Bauelemente in beiden Netzlisten aufgeführt. Es ist Aufgabe des Entwerfers, diese Probleme weiter zu untersuchen, da es sich bei diesen Vergleichsfehlern oder Warnungen sowohl um schwerwiegende Fehler als auch um nicht erkennbare Merkmale für das Extraktionswerkzeug handeln kann.

Eines der Probleme bei der LVS-Verifikation sind die wiederholten Iterationen der Entwurfsprüfung, die erforderlich sind, um die nicht übereinstimmenden Komponenten zwischen beiden Netzlisten zu finden und zu entfernen [5]. Da dies sehr zeitaufwändig sein kann, sollte beim LVS hierarchisch (anstelle eines flachen Vergleichs) vorgegangen werden. Oftmals lassen sich Speicherblöcke und andere IP-Elemente (Intellectual Property) hierarchisch (d. h. einmal intern, danach als Block) vergleichen, während Analogblöcke und Makrozellen eine flache Darstellung beibehalten, die dann auch komplett (jedes Mal) einzubeziehen ist [5].

Neben der Extraktion einer Netzliste aus einem Layout bieten Extraktionswerkzeuge auch die Möglichkeit der *parasitären Extraktion* (*PEX*). Hier werden die parasitären Effekte in den Leiterzügen berechnet. Die Wirkung parasitärer Effekte lässt sich mit (virtuellen) parasitären Schaltungselementen, kurz auch *Parasiten* genannt, darstellen, wobei Folgende hier in Frage kommen: (i) parasitäre Kondensatoren, (ii) parasitäre Widerstände, und (iii) parasitäre Induktivitäten.[5]

Um ein genaueres analoges Schaltungsmodell zu erstellen, ist die Extraktion von diesen Parasiten erforderlich. Auf der Grundlage von Bauelementemodellen und PEX-Ergebnissen können detaillierte Simulationen das tatsächliche Verhalten von digitalen und analogen Schaltungen nachbilden. Ein weiterer Faktor für das steigende Interesse an parasitären Effekten ist die Bedeutung der Verdrahtungskapazität in fortgeschrittenen Technologieknoten: Widerstände und Kapazitäten auf den Verbindungen haben bereits unterhalb des 0,5-µm-Technologieknotens einen erheblichen Einfluss auf die Schaltungsleistung. Parasitäre Zwischenverbindungen verursachen Signalverzögerungen, Signalrauschen und IR-Abfälle – alles wichtige Aspekte, die sich auf das Timing und die Leistung von Schaltungen auswirken, insbesondere bei analogen Schaltungen. Damit ist offensichtlich, dass Timing-Analyse, Leistungsanalyse, Schaltungssimulation und Signalintegritätsanalyse auf der Extraktion von Parasiten beruhen.

Die Methoden zur Extraktion von parasitären Schaltungselementen (Parasiten) lassen sich grob unterteilen in (i) Field-Solver, die physikalisch genaue Lösungen liefern, und (ii) Näherungslösungen mit Pattern-Matching-Techniken. Da Field-Solver sich nur auf kleine Problemfälle anwenden lassen, sind Pattern-Matching-Techniken der einzige praktikable Ansatz zur Extraktion von Parasiten für komplette moderne IC-Designs.

Das Extraktionswerkzeug kann man auch für Antennenprüfungen verwenden (Abschn. 5.4.5). Hier werden die Gate-Fläche und die Fläche des/der Leiter(s) extrahiert, ihr Verhältnis berechnet und mit einem Referenzwert verglichen.

Schließlich wird das Extraktionswerkzeug auch für bestimmte ERC-Funktionen benötigt (Abschn. 5.4.5). Ein Beispiel sind Pin-to-Pin-Prüfungen innerhalb des ERC, bei denen ein bestimmter Widerstandswert nicht zu überschreiten ist, um die ESD-Anforderungen einzuhalten.

[5] Zusätzliche parasitäre Kopplungseffekte enstehen durch das für alle Bauelemente gleiche Chipsubstrat. Diese Effekte werden jedoch nicht in allen Simulationswerkzeugen berücksichtigt.