Konstruktiv hierarchischer Ansatz zur Platzierung und Verdrahtung analoger integrierter Schaltungen

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieurin / Doktoringenieur

(Dr.-Ing.)

von Dipl.-Math. Björn Lipka

geb. am 02.07.1978 in Hannover

genehmigt durch die Fakultät für Elektrotechnik und Informationstechnik

der Otto-von-Guericke-Universität Magdeburg

Gutachter: Prof. Dr.-Ing. Wolfgang Mathis

Prof. Dr.-Ing. Heinrich Klar

Prof. Dr.-Ing. Abbas Omar

Promotionskolloquium am 13.02.2012

Vorwort

Die vorliegende Arbeit entstand während meiner Zeit als wissenschaftlicher Assistent am Lehrstuhl „Integrierte Schaltungen” der Fakultät Elektrotechnik und Informationstechnik der Otto-von-Guericke Universität Magdeburg.

An dieser Stelle danke ich allen Kollegen an der Universität Magdeburg, die mir in Gesprächen wertvolle Hinweise zur Entwicklung der Programme gegeben haben.

Weiterhin danke ich Herrn Prof. Wolfgang Mathis vom Institut für Theoretische Elektrotechnik der Leibniz Universität Hannover, Herrn Prof. Heinrich Klar vom Institut für Mikroelektronik der Technischen Universität Berlin und Herrn Prof. Abbas Omar vom Institut für Elektronik, Signalverarbeitung und Kommunikationstechnik der Otto-von-Guericke Universität Magdeburg für das Interesse, das sie mit der Übernahme des Referats dieser Arbeit entgegen gebracht haben.

Mein ganz besonderer Dank gilt Herrn Prof. Ulrich Kleine, der mich an der Otto-von-Guericke Universität Magdeburg über die vergangenen Jahre intensiv betreut hat und durch wertvolle Diskussionen und Hinweise einen wesentlichen Anteil am Gelingen dieser Arbeit beigetragen hat.

Natürlich möchte ich auch meiner Familie danken, die mich in all den Jahren meiner Ausbildung unterstützt und somit erst die Grundlage für diese Arbeit geschaffen hat.

Kurzfassung

Das Design leistungsfähiger, integrierter Analogschaltungen ist, bedingt durch die große Anzahl variabler Schaltungsparameter, äußerst komplex. Obwohl der Markt im Vergleich zu den digitalen Schaltungen verhältnismäßig klein ist, steigt der Bedarf an analogen Schaltungskomponenten stetig an. Die Layoutsynthese digitaler Schaltungen ist, infolge des Einsatzes automatischer Entwicklungswerkzeuge, mit einem vertretbaren Aufwand zu bewältigen. Demgegenüber werden die Layouts analoger Schaltungen durch die Schaltungsdesigner überwiegend von Hand gezeichnet. Daher ist eine umfassende Unterstützung bei dem Entwurf analoger Schaltungen notwendig. Das Leistungsvermögen und die Einschränkungen eines Werkzeugs haben einen entscheidenden Einfluss auf die Güte und die Kosten der produzierten Schaltungen, sowie auf die notwendigen Ressourcen für den Entwurf. Es ist das automatische Layoutwerkzeug ALADIN entwickelt worden, welches Schaltungsentwicklern die Möglichkeit bietet, spezielles Wissen und Erfahrungen in den Syntheseprozess mit einzubinden, um auf diese Weise qualitativ hochwertige Layouts zu erzeugen.

Im Rahmen dieser Arbeit wird ein sukzessives Platzierungs- und Verdrahtungsverfahren vorgestellt. Aufbauend auf den zu platzierenden Modulen und einer gegebenen Netzlistenbeschreibung wird ein Verbindungsgraph erzeugt. Ein hierarchischer Kompaktierer, der stets zwei Objekte zu einem neuen vereinigt, bildet den Kern des Platzierungsablaufs. Basierend auf dem Verbindungsgraphen wird ein binärer Platzierungsbaum erstellt, der die Reihenfolge und den Zeitpunkt der Kompaktierungen beschreibt. Mit Hilfe eines Entscheidungsbaums werden die geeignete Kompaktierungsrichtung, Orientierung und Topologie der Module jeweils ausgewählt. Nach einem erfolgreichen Kompaktierungsschritt wird automatisch die Verdrahtung der Module durchgeführt und das zugehörige Layout erzeugt. Dazu wird aus der orthogonalen Polygonhülle der Module ein Wegenetz-Graph berechnet. Neben der Auswertung des Graphen unter Verwendung eines Routensuchverfahrens, wird der Einsatz einfacher Leitungselemente zur Realisierung der Verbindung untersucht. Das resultierende Layout des jeweiligen Pfades wird einer Entwurfsregelprüfung mit einer automatischen Korrektur der Regelverletzungen unterzogen.

Zur Dokumentation der Leistungsfähigkeit der präsentierten Verfahren sind verschiedene analoge Schaltungen entworfen worden. Für verschiedene Operationsverstärker und Komparatoren werden neben dem Aufbau und den Layouts, die Ergebnisse der Postlayout-Simulationen vorgestellt. Einige der Schaltungen sind in einer 0.25µm CMOS Technologie der Firma IHP GmbH gefertigt worden, u.a. ein neuartiger zweistufiger Kaskoden-Operationsverstärker. Die resultierenden Messergebnisse werden ebenfalls präsentiert.

Inhaltsverzeichnis

1 Einleitung

Bedeutende Fortschritte auf dem Gebiet der Halbleiterfertigungsverfahren haben dazu geführt, dass integrierten Schaltungen in allen Bereichen der technischen Innovation eine zentrale Bedeutung zukommt. Der Bedarf an immer komplexeren Bausteinen mit entsprechend gesteigerter Leistungsfähigkeit und Integrationsdichte ist stetig gestiegen. Um den Entwurf derartig umfangreicher Schaltungen, insbesondere die digitalen Schaltungen, zu bewältigen, sind die Schaltungsdesigner stark abhängig von leistungsfähigen CAD-Werkzeugen. Diese Programme führen die Synthese eines Schaltungsentwurfs von einer Beschreibung bis hin zur Umsetzung in ein Layout durch. Das Leistungsvermögen und die Einschränkungen eines Werkzeugs haben dabei einen entscheidenden Einfluss auf die Güte und die Kosten der zu produzierenden Schaltungen, sowie auf die notwendigen Ressourcen für den Entwurf. Damit ist die Thematik der Entwurfswerkzeuge integrierter Schaltungen ein bedeutender und stetig wachsender Forschungsbereich.

Der Bedarf an anwendungsspezifischen digitalen Schaltungen ist stark angestiegen. Digitale ICs sind sowohl im Schaltbild als auch im Layout verhältnismäßig einfach zu bewältigen. Komplexe Systeme müssen schnell entworfen werden, um dem Wettbewerbsdruck standhalten zu können. Dazu sind in den letzten Jahren eine große Anzahl an Synthese-Werkzeugen auf dem Gebiet der digitalen, integrierten Schaltungen entworfen worden, deren Verbreitungsgrad und Akzeptanz hinreichend groß ist. Der Markt für analoge Schaltungen ist demgegenüber vergleichsweise klein. In diversen komplexeren Schaltungen werden analoge Komponenten vermieden, wenn diese durch eine entsprechende digitale Signalverarbeitung ersetzt werden kann, die zudem durch eine Software implementiert und konfiguriert werden kann. Dennoch ist die Nachfrage nach analogen Schaltungskomponenten stetig gewachsen. In allen Einsatzbereichen der Elektronik, Kommunikationstechnik, Steuer- und Regeltechnik, Sensorik oder Konsumelektronik erfordert die Schnittstelle zur Umwelt in den meisten Fällen analoge Signale.

Analoge Schaltungen werden häufig in Handarbeit hergestellt. Sie weisen eine erhebliche Anzahl kritischer Parameter auf, die von der Beschaffenheit des Layouts besonders betroffen sind. Im Gegensatz zu digitalen Schaltungen, wo bei einer Vielzahl an Bauelementen von einer fixen Abmessung ausgegangen werden kann, sind analoge Komponenten einem sehr starken Größenunterschied unterworfen. Dadurch wird der Entwurf der Schaltung und des Layouts erschwert. Obwohl die analogen Komponenten zumeist nur einen kleinen Teil der Chipfläche bei Mixed-Signal Schaltungen benötigen, entfällt der größere Anteil an Zeit- und Entwurfskosten auf diese Schaltungen. Sie sind häufiger verantwortlich für Designfehler und teure Redesigns. Die Ursache liegt im größeren Freiheitsgrad analoger Schaltungen im Vergleich zu den digitalen Schaltungen. Infolge einiger besonderer und notwendiger Nebenbedingungen, wie die große Variation der Transistordimensionen, Empfindlichkeit gegenüber parasitären Kapazitäten oder Symmetriebedingungen, ist der Entwurf analoger integrierter Schaltungen deutlich anfordernder. Auf der einen Seite ist die Anzahl der verschiedenen Elementgruppen, wie Stromspiegel oder Differenzstufen, vergleichsweise gering, andererseits variieren diese Zellen erheblich im Hinblick auf ihre Dimensionen und Lage der Anschlüsse. Die geometrischen Parameter von MOS-Transistoren (Weite, Länge und Faltungsfaktor) können, je nach Anwendung, in einem großen Bereich verändert werden.

Im Zuge der Verkleinerung der minimalen Strukturgrößen auf einem Chip, sowie die wachsenden Anforderungen hinsichtlich eines geringen Leistungsverbrauchs und einer hohen Geschwindigkeit, verschlechtern sich die elektrischen Parameter eines MOS Transistors in Bezug auf analoge Schaltungen zunehmend. Damit nimmt das Designfenster, innerhalb dessen die Dimensionierungen der Elemente gewählte werden können, stetig ab. Eine Optimierung wird im Allgemeinen ohne das Layout durchgeführt. Für die parasitären Elemente wird lediglich eine grobe Abschätzung verwendet. Ebenso wird eine Anwendung verschiedener Topologien nur selten geprüft. Eine verlässliche Optimierung erfordert jedoch Kenntnisse über die elektrischen Parameter, die aus dem Layout hervorgehen, wie beispielsweise parasitäre Kapazitäten, RC-Zeitkonstanten der Gates, Stromdichte oder auch das Temperaturverhalten und die resultierenden Einflüsse.

Aus diesen Gründen ist eine umfassendere Unterstützung beim Design analoger Schaltungen notwendig. Es sind eine Reihe von Forschungsarbeiten auf dem Gebiet der Werkzeuge für den Schaltungsentwurf und die Layouterstellung analoger Schaltungen durchgeführt worden [Cohn94, Rijm89]. Aus den Arbeiten gingen diverse Realisierungen hervor, die bei den Schaltungsdesignern jedoch nur wenig Billigung erfahren haben. Eine Ursache ist darin zu finden, dass es sehr schwierig ist, schnelle und flexible Werkzeuge für jede Stufe der Layoutsynthese zu entwerfen, die stets hochqualitative Layouts hervorbringen. Ein weiterer Grund ist der, dass die Schaltungsdesigner häufig eine geeignete Kontrolle über den Syntheseprozess vermissen. Die Werkzeuge umgehen mehr oder weniger den Designer [Rjim89].

Für gewöhnlich geht die Konstruktion von analogen Layouts eher auf jeden einzelnen Transistor als auf Elementgruppen oder Teilschaltungen ein [Cohn94, Garr88, Garr91, Lamp95, Lamp96a, Lamp96b], wodurch die Komplexität, insbesondere für umfangreiche Schaltungen, ansteigt. Die Flexibilität durch Einbeziehung des Schaltungsdesigners ist teilweise ebenfalls eingeschränkt. Das ALADIN Designpaket (Automatic Layout Design Aid for Analog Integrated Circuits) ermöglicht es Schaltungsdesigner, ihr Wissen in den Syntheseprozess mit einzubringen [Wolf96, Wolf98a, Wolf98b, Wolf99a, Wolf99b, Wolf01, Zhan00, Zhan01, Zhan06a, Zhan06b].

Bedingt durch die Anforderungen analoger Schaltungen, ist es vorteilhafter, einzelne Transistoren zu Blöcken, den sogenannten Makrozellen bzw. Modulen, zusammenzufassen. Damit jeweils ein akzeptables Layout für eine große Anzahl an parametrisierbaren Modulen verfügbar ist, sind entsprechende Modulgeneratoren bereitzustellen. Um die Modulgeneratorbibliothek überschaubar zu halten, aber dennoch optimale Layouttopologien für einen weiten Parameterbereich bereithalten zu können, ist in ALADIN eine Modulgeneratorumgebung implementiert worden. Darin können mit Hilfe einer einfachen, natürlichen Beschreibungssprache technologie- und anwendungsunabhängige Generatoren entwickelt werden. Das konstruktive Modulgeneratorkonzept basiert auf einer relativen Platzierung von Objekten ohne Berücksichtigung der Entwurfsregeln in der Modulbeschreibung. Infolge der automatischen Berücksichtigung dieser Regeln durch die Umgebung, ist das resultierende Layout stets entwurfsregelkonform.

Nach der Erstellung der Topologien bleiben die Modul-Layouts derart anzuordnen, dass der Flächenbedarf und die Leitungslängen, unter Einhaltung weiterer Nebenbedingungen, minimal werden. Es ist jedoch zu beachten, dass sich bei der Verwendung nicht optimaler Module selbst mit aufwändigen Platzierungs- und Verdrahtungsverfahren nur suboptimale Ergebnisse erzielen lassen.

Für die vorliegende und die vorhergehenden Arbeiten stand das Designframework II von Cadence zur Verfügung. Aus diesem Grund ist eine Schnittstelle zu dieser Entwicklungsumgebung entworfen worden.

Um die Verlässlichkeit zu gewährleisten, erfolgen die Platzierung und Verdrahtung nahezu simultan. Die Zellen werden sukzessive miteinander in Verbindung gesetzt und positioniert. Damit ist eine gewisse Vorhersagbarkeit gegeben.

Nach diesem einleitenden Kapitel folgt im nächsten Kapitel eine Beschreibung passiver integrierter Elemente. Es wird der Einfluss des Substrats auf die Bauelemente kurz dargestellt und die Möglichkeiten zur Minderung der Substrateinkopplungen beschrieben. Unter Verwendung der MOGLAN Bibliothek werden Modulgeneratoren entworfen, mit denen das Layout von symmetrischen spirale Induktivitäten, Präzisionswiderständen, binären Kapazitätsfeldern und gepaarte Kapazitäten erstellt werden kann. Um die Induktivitäten im Layout simulieren zu können, wird für diese eine entsprechende Modellierung präsentiert. In Kapitel 3 erfolgt eine Beschreibung der im Rahmen dieser Arbeit entstandenen Operationsverstärker und Komparatoren. Für das Layout sind die Zellen mit Hilfe der Modulgeneratorumgebung erstellt worden. Für einen Teil der Operationsverstärker sind Prototypen in einer 0,25µm CMOS Technologie produziert worden, so dass neben den Simulationsergebnissen die entsprechenden Messergebnisse präsentiert werden. Die nähere Erläuterung bekannter Lösungsansätze und -verfahren zur Platzierung und Verdrahtung integrierter analoger Schaltungen sind Bestandteile von Kapitel 4.

In Kapitel 5 wird das in dieser Arbeit entwickelte Verfahren zur Platzierung analoger, integrierter Schaltungen vorgestellt. Zunächst erfolgt eine Beschreibung des Layoutmodells, welches für die Platzierung genutzt wird. Anschließend werden die Schritte erläutert, die notwendig sind, um einen Verbindungsgraphen aus der zu bearbeitenden Schaltung zu erhalten. Aufgrund der Tatsache, dass die Kompaktierung von Modulen ein zentraler Bestandteil der Platzierung in dieser Arbeit ist, wird ein kurzer Überblick zu den verschiedenen Kompaktierungsverfahren gegeben. Basierend auf dem Verbindungsgraphen wird ein binärer Platzierungsbaum erstellt. Dieser Platzierungsbaum legt die Reihenfolge fest, wann die jeweiligen Module kompaktiert und verdrahtet werden sollen. Die Auswertung des Baumes erfolgt in einem Post-Order-Durchlauf. Dabei werden, unter Benutzung von Entscheidungs- und Schnittbäumen, die Lage und Orientierung der Module zueinander ermittelt und anschließend die Platzierung und Verdrahtung der Module durchgeführt.

Die Vorgehensweise zur Verdrahtung analoger integrierter Schaltungen wird in Kapitel 6 dargelegt. Basierend auf der orthogonalen Polygonhülle der beteiligten Zellgeometrien wird ein Wegenetz-Graph, der die möglichen Pfade repräsentiert, erstellt. Neben der Anwendung eines Routensuchverfahrens in dem resultierenden Graphen, um zwei Terminals miteinander zu verbinden, wird der Einsatz einfacher Leitungselemente zur Realisierung der Verdrahtung geprüft. Nach der Layouterstellung der ermittelten Routen wird eine Entwurfsregelprüfung mit einer automatischen Korrektur der möglichen Regelverletzungen durchgeführt. Die Leistungsfähigkeit der entworfenen Werkzeuge wird anhand von ausgewählten Beispielen in Kapitel 7 dokumentiert. Abschließend wird in Kapitel 8 eine Zusammenfassung dieser Arbeit ausgeführt. Dabei werden die wichtigsten Ergebnisse in kompakter Form wiedergegeben. Zusätzlich erfolgt ein Ausblick auf zukünftige Arbeiten.

2 Passive integrierte Elemente

Typische integrierte Schaltungen bestehen aus komplexen, miteinander verbundenen und interagierenden Blöcken, die sich aus digitalen, abgetasteten analogen (z.B. SC-Schaltungen bzw. D/A- oder A/D-Umsetzern) und zeitkontinuierlichen analogen Schaltungen zusammensetzen können. Um die beim Design der Schaltung anvisierte Leistungsfähigkeit auch beim gefertigten Schaltkreis zu erzielen, ist es notwendig, parasitäre Effekte und herstellungsbedingte Einflüsse während der Entwurfsphase mit zu berücksichtigen. Neben Transistoren und Dioden werden für analoge Schaltungskomponenten häufig zusätzliche passive Bauelemente benötigt. So finden Induktivitäten in der Hochfrequenztechnik häufig Anwendung in Filtern oder Bias-Netzwerken von Verstärkern, z.B. zur DC-Entkopplung. Integrierte Kapazitäten kommen in SC-Schaltungen, Verstärkern oder Hochfrequenz-ICs zum Einsatz und integrierte Widerstände werden beispielsweise in D/A- bzw. A/D-Umsetzern eingesetzt. Es gibt eine Vielzahl an Einsatzmöglichkeiten und auch eine große Anzahl Realisierungsvarianten für die passiven Bauelemente. Die entscheidenden Kriterien für die Auswahl von verschiedenen Realisierungsmöglichkeiten sind der Nominalwert des Bauelements, die Genauigkeitsanforderungen und parasitäre Effekte, die das entsprechende Bauteil aufweisen kann. Eine hinreichend genaue Einstellung der Verhältnisse der Nominalwerte ist bei den integrierten Schaltungen in der Regel möglich. Hingegen sind größere Toleranzen bezüglich der Absolutgenauigkeit nicht zu vermeiden.

In diesem Kapitel werden zu Beginn der Einfluss des Substrats auf die Bauelemente näher erläutert und Möglichkeiten zur Reduktion von Substrateinkopplungen aufgezeigt. Anschließend wird der Aufbau integrierter Induktivitäten und die wichtigsten auftretenden physikalischen Effekte beschrieben. Es werden eine Modellierung der Induktivität und ein Modulgenerator zur Erstellung symmetrischer spiralförmiger Induktivitäten präsentiert. Im nachfolgenden Abschnitt soll die Genauigkeit von integrierten Widerständen und die Einflussnahme durch die Temperatur diskutiert werden. Außerdem werden Modulgeneratoren zur automatischen Erstellung des Layouts von integrierten Präzisionswiderständen beschrieben. Abschließend erfolgt eine Untersuchung der Genauigkeit von integrierten Kapazitäten und der Präsentation von Generatoren zur Layoutkonstruktion von binären Kapazitätsfeldern und gepaarten Kapazitäten.

2.1 Einfluss des Substrats

Im Gegensatz zu diskret aufgebauten Schaltungen können bei integrierten Schaltungen einzelne Bauelemente nicht für sich allein betrachtet oder modelliert werden. Alle Elemente befinden sich gemeinsam auf demselben Substrat. Dieses Substrat ist kein idealer Isolator, so dass Kopplungen zwischen den Elementen und Schaltungsteilen auftreten. Unter bestimmten Voraussetzungen können einige dieser Kopplungen beim Design vernachlässigt werden, da ihr Einfluss auf die Schaltung gering ist. So fällt nur der Einfluss des Substrats bis zur Kontaktierung ins Gewicht, wenn beispielsweise das Substrat um ein Element oder Schaltungsteil sehr niederohmig mit Masse verbunden wird. In Abhängigkeit von der Empfindlichkeit der Schaltung wird die Verkopplung mit anderen Elementen daraufhin derart klein, dass sie nicht mehr berücksichtigt werden muss.

Um die Art der Verkopplungen genauer beschreiben zu können, soll zunächst auf die Eigenschaften des Substrats näher eingegangen werden.

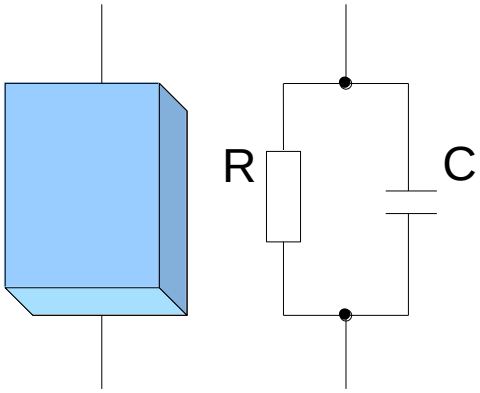


Abbildung 2-1: Ersatzschaltbild eines Substratquaders

Betrachtet man einen Quader innerhalb des Substrats (siehe Abbildung 2-1) und nimmt einen Stromfluss vom oberen zum unteren Rand an, so lässt sich die Impedanz durch das in Abbildung 2-1 dargestellte Ersatzschaltbild beschreiben. Ein genaues Modell besteht also aus einer Vielzahl infinitesimal kleinen Parallelschaltungen von ohmschen Widerständen und Kapazitäten entlang der Richtung des Stroms, für den die Kopplung beschrieben werden soll. Für die Werte von R und C gilt dabei:

$$R = \rho \frac{l}{A}, \tag{2-1}$$

$$C = \varepsilon_0 \varepsilon_r \frac{A}{l}, \tag{2-2}$$

$$RC = \rho \varepsilon_0 \varepsilon_r. \tag{2-3}$$

Dabei ist ρ der spezifische Widerstand, l die Länge des Quaders und A dessen Fläche quer zur Stromrichtung. Die Konstanten ε_0 und ε_r beschreiben die Dielektrizität des Materials.

Eine einfache Beschreibung ist die Verkopplung durch lediglich eine einzige parallele RC- Schaltung zwischen zwei Punkten in der Schaltung. In den überwiegenden Fällen erweist sich diese Vorgehensweise als hinreichend. Für niedrige Frequenzen kann eine Modellierung der Verkopplung durch das Substrat auch einfach nur durch einen ohmschen Widerstand erfolgen. So ist diese Betrachtung bei der Modellierung des Einflusses von Wirbelströmen, die durch integrierte Induktivitäten hervorgerufen werden, durchaus ausreichend [Pete03].

Alle Kopplungen über das Substrat sind abhängig von der Art der Kontaktierung. Eine Abschätzung der Stärke der Kopplung kann daher erst nach der Fertigstellung des Layouts erfolgen. Aus diesem Grund wird sie in keinem der Modelle für einzelne Bauelemente berücksichtigt [Hein04]. Es ist die Aufgabe des Designers diese Einflüsse so gering wie möglich zu halten, abzuschätzen und in die Nachsimulation nach der Layouterstellung mit einfließen zu lassen.

2.1.1 Möglichkeiten zur Reduzierung von Substrateinkopplungen

Substrateinkopplungen sind insbesondere bei Mixed-Signal Schaltungen zu beobachten, da sich hier die rauschempfindlichen analogen Schaltungskomponenten und die digitalen Schaltungen, die das Rauschen verursachen, auf demselben Substrat befinden. Mit Hilfe eines sorgfältigen Layouts ist jedoch möglich, den Effekt der Substrateinkopplung zu minimieren. Im Folgenden werden die Möglichkeiten der räumlichen Trennung von rauschempfindlichen und rauschenden Komponenten und der Einsatz von Guardringen zur Reduzierung der Substrateinkopplungen näher erläutert.

2.1.1.1 Räumliche Trennung

Ein Lösungsansatz zur Reduzierung der Substrateinkopplungen ist gegeben durch eine räumliche Trennung der rauschempfindlichen Schaltungsbereiche von den stark rauschenden Komponenten. Die Effektivität dieser Vorgehensweise ist jedoch stark abhängig von der Beschaffenheit des Substrats, das zur Herstellung der Schaltung verwendet wird. So wird bei der Fabrikation von integrierten Schaltungen zwischen hochohmigen und niederohmigen Substrat unterschieden [Su93]. Das niederohmige Substrat besteht aus einer schwach dotierten Epitaxie-Schicht oberhalb von der stark dotierten Hauptschicht. Der überwiegende Teil der lateralen Ströme fließt in diesem Fall durch die stark dotierte Schicht, welche sich unter elektrischen Gesichtspunkten damit wie einziger Knoten verhält. Eine räumliche Trennung zwischen Rauschquellen und rauschempfindlichen Strukturen führt in diesem Fall zu keiner signifikanten Reduzierung der Einkopplung [Su93]. Im Gegensatz dazu setzt sich das hochohmige Substrat lediglich aus einer gleichmäßig schwach dotierten Schicht zusammen. Infolge eines fehlenden niederohmigen Bereichs, ist der Stromfluss innerhalb des Substrats weitestgehend gleichmäßig. Aus diesem Grund kann bei Verwendung des hochohmigen Substrats die Isolierung zwischen digitalen und analogen Schaltungsbereichen durch eine Vergrößerung der räumlichen Distanz verbessert werden [Su93].

2.1.1.2 Guardringe

Ein Guardring bezeichnet einen Substraktkontakt, der ringförmig um eine zu schützende Struktur gelegt wird. Ein derartiger Ring wird aus p+-Diffusion bzw. n+-Diffusion gebildet und auf VSS- bzw. VDD-Potential gelegt. Das Ziel eines Guardrings besteht darin, die Ausbreitung von Störungen aus dem Substrat in die Epitaxie-Schicht zu verhindern [Puwa95]. Der Guardring ist dabei so nah wie möglich an die zu schützenden Elemente zu platzieren.

2.2 Induktivitäten

Induktivitäten sind Schlüsselelemente hochfrequenter Schaltungen. Sie werden benötigt, um verlustfreie Anpassungen zu realisieren oder kommen als Lasten bei Schaltungen zum Einsatz, die bei geringen Versorgungsspannungen arbeiten oder schmalbandig sein müssen.

Problematisch ist jedoch die umfassende Modellierung. Insbesondere das Substrat, auf dem sie aufgebracht sind, kann in CMOS nicht als idealer Isolator angenommen werden. Die Folge sind sowohl Verluste durch kapazitive Kopplungen, als auch die Erzeugung von Wirbelströmen im Substrat (Eddy Currents). Derzeit werden von keinem Hersteller parametrisierbare Modelle von Induktivitäten zur Verfügung gestellt. In manchen Fällen wurden lediglich einige Spulen implementiert und anschließend vermessen, wodurch die Freiheit beim Schaltungsdesign stark eingeschränkt ist [Hein04].

Nachfolgend wird der Einfluss des Substrats sowie die Grundlagen zum Aufbau von integrierten Induktivitäten aufgezeigt und erklärt. Anschließend wird der im Rahmen dieser Arbeit angefertigte Generator für integrierte Induktivitäten sowie die zur Simulation verwendeten Modelle näher erläutert.

2.2.1 Aufbau integrierter Induktivitäten

Die Ausführung einer Induktivität ist auf verschiedene Arten möglich und kann viereckig, oktagonal oder sogar hexagonal sein. Die größte Induktivität pro Fläche wird jedoch mit einer runden Realisierung erzielt. Aus fertigungstechnischen Gründen sind solche Spulen jedoch in vielen Prozessen nicht zugelassen.

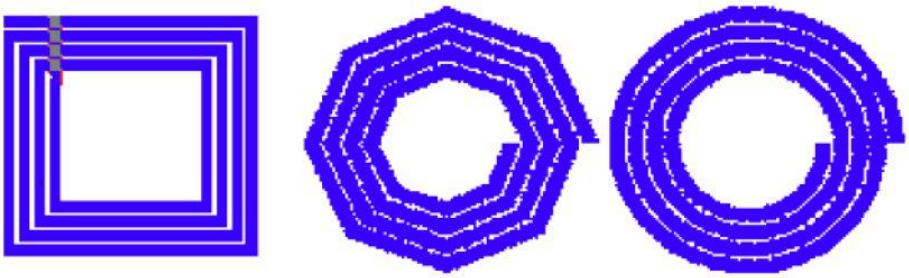


Abbildung 2-2: Beispiele integrierter Induktivitäten

Das Layout einer Induktivität kann auch von der Art der Anwendung bestimmt sein. So ist beispielsweise für differentielle Schaltungen oft eine symmetrische Spule wünschenswert. In Abbildung 2-3 ist das automatisch generierte Layout einer symmetrischen spiralen Induktivität dargestellt.

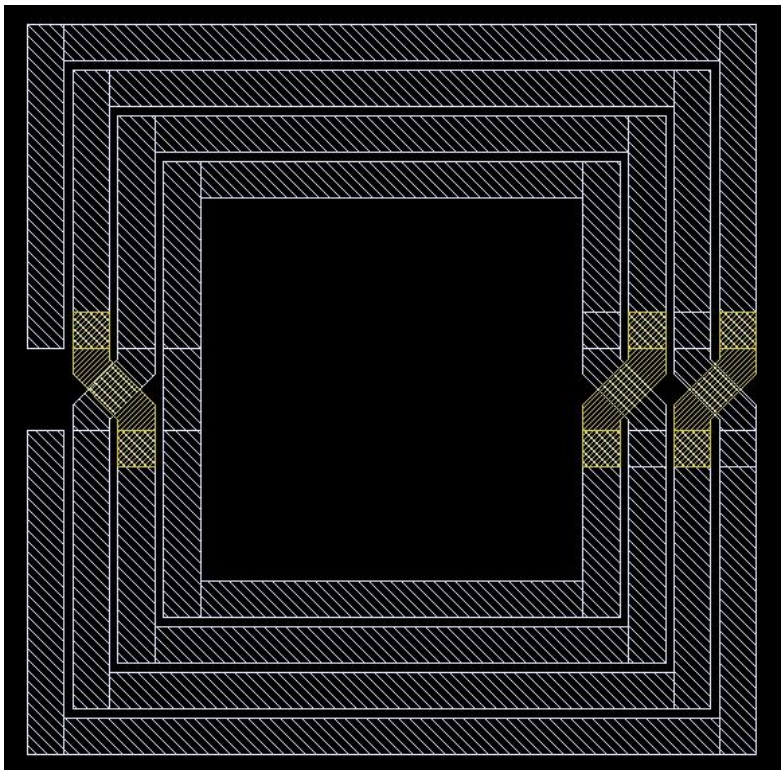


Abbildung 2-3: Layout einer symmetrischen Induktivität

Um eine möglichst hohe Güte der Spule zu erreichen, werden in der Regel alle zur Verfügung stehenden Metall-Lagen genutzt. In einigen Fällen kann es von Vorteil sein, die unteren Metall-Lagen nicht zu verwenden. Auf diese Weise lässt sich der Abstand der Spule zum Substrat vergrößern. Daraus resultiert eine Abnahme der parasitären Kapazität.

2.2.2 Physikalische Effekte

Im Folgenden werden die wichtigsten physikalischen Effekte beschrieben, die bei einer integrierten Spule auftreten und zu berücksichtigen sind.

2.2.2.1 Eigenresonanz

Infolge der parasitären Kapazität zwischen dem Metall der Spule und dem Substrat verhält sich die Induktivität wie ein RC-Schwingkreis. Die Frequenz, bei der dieser Schwingkreis in Resonanz gerät,

wird Eigenresonanzfrequenz f_{res} der Spule genannt. Anhand der nachfolgenden Gleichung kann sie ermittelt werden

$$f_{res} = \frac{1}{2\pi\sqrt{LC}}. \tag{2-4}$$

Hierbei ist L die Induktivität der Spule und C die parasitäre Kapazität zum Substrat. Oberhalb dieser Frequenz verhält sich die Spule wie eine Kapazität. Um eine starke Abnahme der Güte zu vermeiden, sollte die Frequenz, bei der die Spule zum Einsatz kommt, möglichst weit unterhalb der Resonanzfrequenz liegen [Pete03].

Um eine hohe Eigenresonanzfrequenz zu erzielen, muss die Kapazität, bei einem festen Wert L der Induktivität, möglichst gering sein. Dies kann durch einen großen Abstand zum Substrat oder eine kleine Fläche der Spule erreicht werden. Es ist zu beachten, dass eine kleine Fläche zunehmende ohmsche Verluste im Leiter zur Folge hat.

2.2.2.2 Skin Effekt

Der Skin Effekt, auch Stromverdrängung genannt, gewinnt bei Induktivitäten an Einfluss, wenn diese im Gigahertzbereich betrieben werden. Der Effekt zeichnet sich dadurch aus, dass infolge des höherfrequenten Wechselstroms die Stromdichte im Inneren des Leiters niedriger ist als an der Oberfläche. Die Eindringtiefe d ist abhängig von der magnetischen Permeabilität μ, dem spezifischen Widerstand des Materials ρ und der Frequenz f:

$$\delta = \sqrt{\frac{\rho}{\pi\mu f}}. \tag{2-5}$$

Der Effekt ist durch den Schaltungsentwickler nur sehr beschränkt beeinflussbar. In verschiedenen Publikationen wird gezeigt, dass bei planaren Induktivitäten der Einfluss des Skin Effekts mit der Leiterbahnbreite ansteigt. Nach [Long97] liegt die optimale Leiterbahnbreite bzgl. des ohmschen Widerstands bei 15μm. Eine weitere Vergrößerung der Leiterbahnbreite führt nicht zu einer Verbesserung der Güte.

2.2.2.3 Wirbelströme

In den Windungen einer Spule und im Substrat unter einer Spule können Ströme auftreten, die durch das Magnetfeld einer Induktivität verursacht werden. Diese Wirbelströme (eddy current) erzeugen ihrerseits ein Magnetfeld, welches bei hohen Frequenzen und großen Querschnitten den Strom aus der Mitte des Leiters verdrängt (Skin Effekt). Der Einfluss dieses Effekts auf die Güte der Spule ist größer als der des eigentlichen Skin Effekts.

Betrachtet man zunächst nur die induzierten Ströme im Substrat, so ist deren Einfluss auf die Induktivität vernachlässigbar klein, wenn es sich um ein Substrat mit niedriger Dotierung handelt. Bei niederohmigen Substraten nimmt der Einfluss dieses Effekts hingegen zu. Das durch die induzierten Substratströme entstandene Magnetfeld durchdringt den Leiter der Spule und ruft wiederum Ströme hervor, die dem ursprünglichen Strom entgegen gerichtet sind. Zur Ermittlung genauer Werte der Wirbelströme und deren Einfluss auf die Spule sind Simulationen mit Finiten Elementen durchzuführen, da eine direkte Berechnung nur sehr ungenau möglich ist. Anschaulich kann man sich die Ströme im Substrat und in der Induktivität auch wie in einem Übertrager vorstellen. Der ohmsche Widerstand im Substrat überträgt sich auf die Primärspule und verschlechtert auf diese Weise deren Güte. In der Mitte der Induktivität ist das Magnetfeld infolge der Überlagerung aller Anteile der Windungen am stärksten. Dies hat zur Folge, dass auch die Wirbelströme hier am größten sind und

damit sowohl auf den Wert der Induktivität, als auch auf den Wert des ohmschen Widerstands einer Spule den meisten Einfluss ausüben. Aus diesem Grund ist das Zentrum bei Induktivitäten mit einer hohen Güte nicht zu verwenden.

Eine Möglichkeit Wirbelströme im Substrat und die damit verbundenen ohmschen Verluste in der Spule zu vermeiden, besteht in der Verwendung einer Substratabschirmung. Es wird hierbei eine leitfähige Schicht aus Metall oder Polysilizium unter den Wicklungen der Spule aufgebracht. Auf diese Weise wird ein Eindringen des Magnetfeldes in das Substrat verhindert. Um Wirbelströme innerhalb des Schildes zu vermeiden, ist es notwendig, dieses rechtwinklig zu den Leitern der Spule mit Unterbrechungen (Schlitze) zu versehen. Ein Nachteil bei der Verwendung von Substratabschirmungen ist die Zunahme des Kapazitätswertes unter der Spule, infolge des geringeren Abstands des unteren Leiters zur Abschirmung als zum Substrat. Die Eigenresonanzfrequenz der Spule wird auf diese Weise ebenfalls herabgesetzt. Aus diesem Grund ist bei Anwendungen, die beispielsweise eine sehr geringe parasitäre Kapazität der Spule benötigen, auf eine derartige Abschirmung zu verzichten.

2.2.3 Modellierung

Um eine integrierte spiralförmige Induktivität in einer CMOS Technologie modellieren zu können, wird die Spule in rechteckige Segmente unterteilt. Zur Vereinfachung der Implementierung werden lediglich quadratische spiralförmige Spulen betrachtet. Eine Erweiterung auf andere Geometrien, wie beispielsweise oktagonale Spiralen, ist mit einem erhöhten Rechenaufwand möglich. Das Ersatzschaltbild, welches für jedes Segment angewandt wird, ist der Abbildung 2-4 zu entnehmen [Ashb94]. Neben der Reihenschaltung, bestehend aus der Induktivität L_S und dem parasitären Leitungswiderstand R_S, werden ebenfalls die Streukapazitäten zwischen den Leitungen (C_S) und die Wechselwirkungen mit dem Substrat (C_{ox}) berücksichtigt. Das ohmsche und das weitere kapazitive Verhalten des Substrats werden durch die Parallelschaltung von C_{si} und R_{si} beschrieben. Die Induktivität L_S beinhaltet neben der Eigeninduktivität des jeweiligen Segments, die wechselseitige Induktivität zwischen den entsprechenden parallel verlaufenden Segmenten. Der Reihenwiderstand R_S vergrößert sich mit steigender Frequenz infolge des Skin Effekts. Die folgende Gleichung ermöglicht eine Berechnung des Widerstands für die einzelnen Segmente der spiralförmigen Spule unter Berücksichtigung des genannten Effekts [Yue99]:

$$R_S = \frac{\rho L}{W\delta\left(1-e^{-t_W/\delta}\right)}, \tag{2-6}$$

mit

$$\delta = \sqrt{\frac{\rho}{\pi\mu f}}. \tag{2-7}$$

wobei ρ den spezifischen Widerstand des Materials, welches für Leitungen der spiralen Induktivität verwendet wird, beschreibt. Die Dicke der Leitung wird mit Hilfe von t_W spezifiziert, L ist die Länge des jeweiligen Segments, W ist die Breite der Leitung, μ die Permeabilität des Leitungsmaterials und f die Frequenz, für die der Serienwiderstand berechnet werden soll.

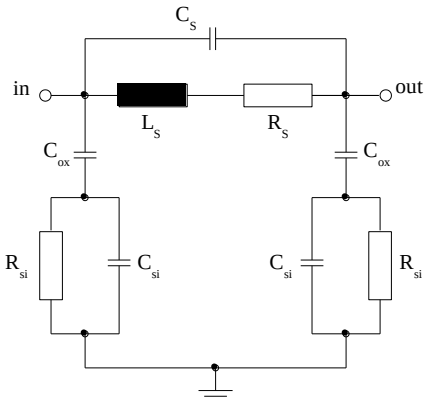


Abbildung 2-4: Ersatzschaltbild der Spule

Die Kapazität Cox baut sich zwischen der untersten Maskenebene der spiralen Spule und dem Substrat auf. Der Wert für die Kapazität berechnet sich wie folgt:

$$C_{ox} = \frac{1}{2} \cdot L \cdot W \cdot \frac{\varepsilon_0 \cdot \varepsilon_{ox}}{t_{ox}}, \tag{2-8}$$

dabei ist ε_0 die Dielektrizitätskonstante des Vakuums, ε_{ox} die relative Dielektrizitätskonstante des Oxids zwischen der untersten Ebene der Spule und dem Substrat. Die Länge des jeweiligen Segments wird durch L angegeben, W beschreibt die Leitungsbreite und t_{ox} definiert den Abstand zwischen der Maskenebene und der Epitaxie-Schicht. Falls in der einzusetzenden Technologie keine Epitaxie-Schicht existiert, ist durch t_{ox} der Abstand zum Substrat gegeben.

Die hochfrequenten kapazitiven Effekte, die bei der integrierten spiralförmigen Spule auftreten können, werden durch die Kapazität C_{si} modelliert. Eine Einflussnahme durch die Leitfähigkeit des Substrats wird durch den Widerstand R_{si} berücksichtigt. Die Abmessungen einer spiralen Induktivität sind in der Regel deutlich größer als die Oxiddicke und damit eher vergleichbar mit der Dicke des Substrats. Aus diesem Grund verhalten sich die Werte der Substratkapazität und des Substratwiderstands näherungsweise proportional zur Fläche, die durch die Spule eingenommen wird. Damit können C_{si} und R_{si} mittels folgender Gleichungen approximiert werden:

$$C_{si} = \frac{1}{2} \cdot L \cdot W \cdot C_{substrat}, \tag{2-9}$$

und

$$R_{si} = \frac{2 \cdot R_{substrat}}{L \cdot W}, \tag{2-10}$$

wobei L die Länge des jeweiligen Segments der Spule darstellt und W die Leitungsbreite ist. Der Kapazitätswert des Substrats pro Flächeneinheit ist durch $C_{substrat}$ gegeben, während $R_{substrat}$ den Widerstand des Substrats pro Flächeneinheit beschreibt.

Die Berechnung der Werte der Induktivität selber ist deutlich aufwendiger. Wie eingangs bereits erwähnt wurde, ist neben der Eigeninduktivität die wechselseitige Induktivität zu parallel verlaufenden Segmenten der Spule für jeden Leitungsabschnitt zu berechnen. Nach [Yue00] kann die Eigeninduktivität eines rechteckförmigen Leitungssegments durch die nachfolgende Gleichung bestimmt werden:

$$M_0 = 2 \cdot L \cdot \left(\log\frac{2 \cdot L}{W + t_W} + \frac{W + t_W}{3 \cdot L} + \frac{1}{2}\right), \tag{2-11}$$

wobei L die Länge des Segments, W die Leitungsbreite und t die Dicke der Leitung definieren. Zur Berechnung der wechselseitigen Induktivität von parallel verlaufenden Leitungen ist die Abbildung 2-5 zu betrachten.

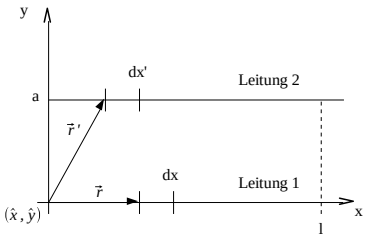


Abbildung 2-5: Wechselseitige Teilinduktivität von zwei parallel verlaufenden Leitungen

Hier bezeichnet a den Abstand zwischen den beiden Leitungen und l die Länge hinsichtlich der die beiden Segmente parallel verlaufen. Es gilt ferner:

$$\vec{r} = x \cdot \hat{x}, \tag{2-12}$$

$$\vec{r}' = x' \cdot \hat{x} + a \cdot \hat{y}, \tag{2-13}$$

$$|\vec{r}' - \vec{r}| = \sqrt{(x' - x)^2 + a^2}. \tag{2-14}$$

Damit ergibt sich für die wechselseitige Teilinduktivität:

$$M = \int_0^l \int_0^l \frac{dx \cdot dx\prime}{\sqrt{(x'-x)^2 + a^2}} = 2 \cdot l \cdot \left[\log\left(\frac{l}{a} + \sqrt{1 + \left(\frac{l}{a}\right)^2}\right) - \sqrt{1 + \left(\frac{l}{a}\right)^2} + \frac{a}{l}\right]. \tag{2-15}$$

Um aus der resultierenden Teilinduktivität die tatsächliche wechselseitige Induktivität zwischen den parallel verlaufenden Leitungen zu ermitteln, ist die Distanz a durch den mittleren geometrischen Abstand GMD zu ersetzen [Hoer65, Gree74]. Ein hinreichend genauer Ausdruck zur Berechnung dieses Wertes ist durch die folgende Gleichung gegeben:

$$\log GMD = \log d - \frac{w^2}{12d^2} - \frac{w^4}{60d^4} - \frac{w^6}{168d^6} - \frac{w^8}{360d^8} - \frac{w^{10}}{660d^{10}}. \tag{2-16}$$

wobei d die Distanz zwischen den Leitungen und w die Leitungsbreite angeben.

Basierend auf Grovers Formeln [Grov62] entwickelte Greenhouse einen Algorithmus zur Berechnung von ebenen rechtwinkeligen Spiralen [Gree74]. Diese Methode zeigt, dass die Gesamtinduktivität einer spiralförmigen Spule dadurch berechnet werden kann, dass die Eigeninduktivität jedes Leitungssegments sowie die wechselseitigen Induktivitäten zwischen den entsprechenden Segmenten aufsummiert wird. Die wechselseitige Induktivität hängt dabei vom Winkel der Segmente zueinander, deren Länge und dem Abstand ab. Orthogonal verlaufende Segmente weisen dabei keine Kopplung auf, da zwischen diesen kein magnetischer Induktionsfluss besteht.

Für die Auswertung der Gesamtinduktivität einer quadratischen Spirale mit N Windungen sind 4N Terme für die Eigeninduktivität, 2N(N-1) positive und $2N^2$ negative Terme für die wechselseitige Induktivität zu berechnen. Alle geraden Segmente einer spiralförmigen Spule seien durchnummeriert

von 1 bis N. Die Gesamtanzahl der Segmente ist damit durch N gegeben. Die Nummerierung erfolgt von außen nach innen. Damit lässt die Induktivität der Spule nach Algorithmus 2-1 berechnen.

Algorithmus 2-1: Berechnung der Induktivität einer Spule

```
M0:=Summe der Eigeninduktivitäten der Segmente si mit si∈S und 1≤i≤N;
M-:=0;
für i:=1 bis N-2
   für j:=1 bis n
      k:=i+4*j-2;
      wenn k≤N dann
         Mik:=wechselseitige Induktivität zwischen Segment si und sk;
         M-:=M-+Mik;
      wenn_ende
   für_ende
für_ende
M-:=M-*2;
M+:=0;
für i:=1 bis N-2
   für j:=1 bis n
      k:=i+4*j;
      wenn k≤N dann
         Mik:=wechselseitige Induktivität zwischen Segment si und sk;
         M+:=M++Mik;
      wenn_ende
   für_ende
für_ende
M+:=M+*2
L:=M0+(M+-M-):
```

2.2.4 Modulgenerator zur Erstellung von symmetrischen Induktivitäten

In diesem Abschnitt wird ein Generator zur automatischen Erstellung des Layouts einer integrierten symmetrischen spiralförmigen Induktivität beschrieben. Die MOGLAN Bibliothek dient als Grundlage für den Generator. Dadurch ist ferner eine Unabhängigkeit von der zu verwendenden Technologie gewährleistet. Als Parameter besitzt dieses Modul die Anzahl der Windungen der Spule, die Breite der Leitung und die Innenweite der Spule. Innerhalb dieses Gebiets werden keine Windungen erstellt. Weiterhin kann die Maskenebene vom Benutzer ausgewählt werden, die für die Erstellung der Leitungen verwendet wird. Um die Spule auch bei einer Postlayout-Simulation mit zu berücksichtigen, ist es möglich eine Spectre-Beschreibung durch den Generator erstellen zu lassen. Dazu werden das im Abschnitt “Modellierung” beschriebene Ersatzschaltbild und die entsprechenden Gleichungen umgesetzt. Für die erfolgreiche Modellierung der Simulationsdaten sind durch den Benutzer Angaben zum Flächenwiderstand des Leitungsmaterials, zu der Dicke der Epitaxie-Schicht, zu dem Abstand zwischen der Epitaxie-Schicht und der Leitungsebene sowie zur Dielektrizitätskonstante des Oxids zu machen. Falls die zu benutzende Technologie über keine Epitaxie-Schicht verfügt, sind die entsprechenden Werte für das Substrat zu wählen.

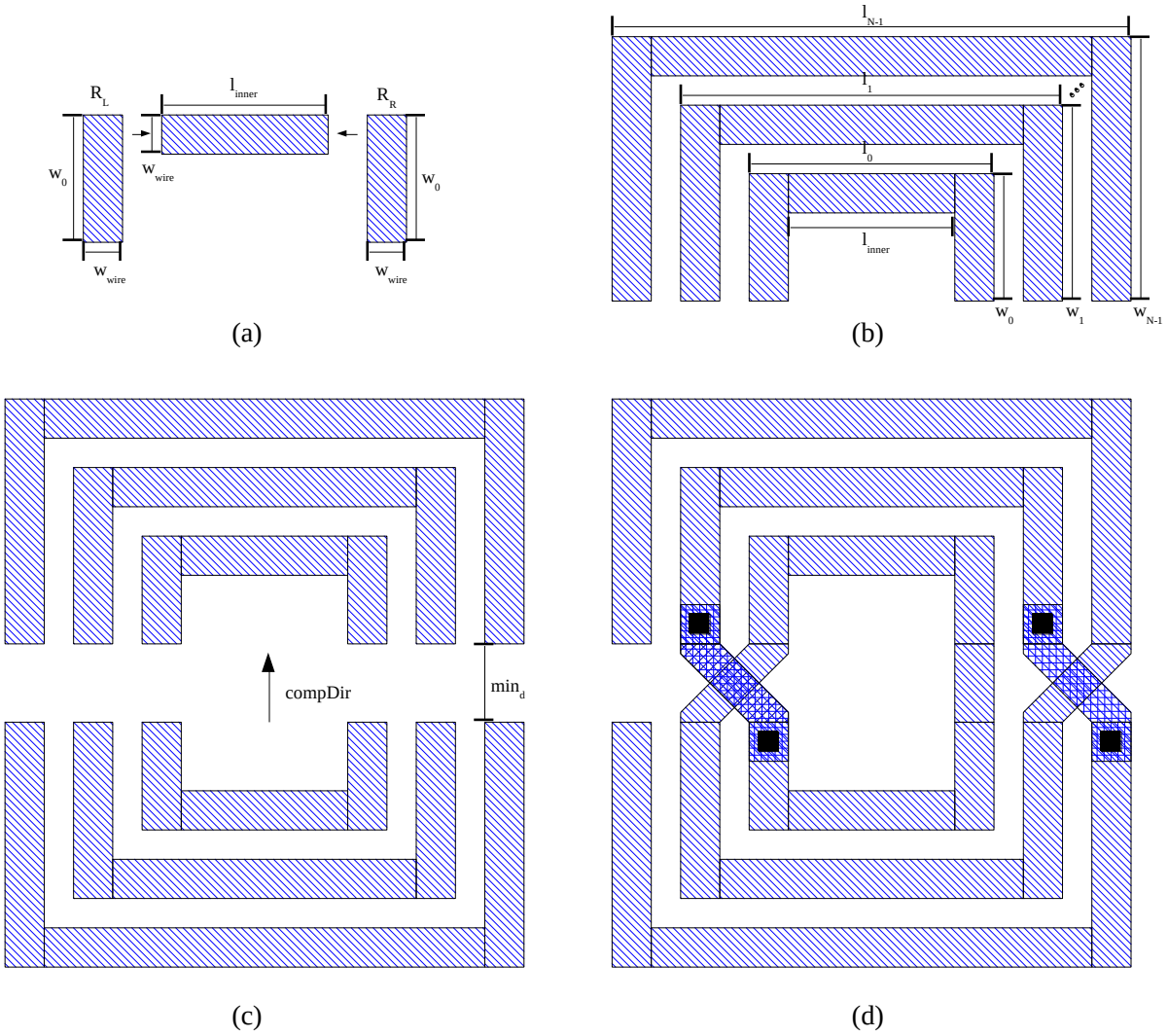


Abbildung 2-6: Abfolge zur Erstellung einer symmetrischen spiralen Induktivität

Der Ablauf der Konstruktion des Layouts einer symmetrischen spiralförmigen Induktivität in einer CMOS Technologie ist in Abbildung 2-6 dargestellt. Der erste Schritt beschreibt die Erstellung von Teilwindungen, bestehend aus je drei Rechtecken (siehe Abbildung 2-6 (a)), die aneinander kompaktiert werden. Der Vorgang wird N-mal durchgeführt, wobei N die Anzahl der Windungen der Spule definiert. Die Größen w_{wire} und l_{inner} werden durch den Benutzer vorgegeben und spezifizieren die Leitungsbreite bzw. die Innenweite der spiralen Induktivität. Die Abmessung w_0 der Rechtecke R_L und R_R errechnet sich wie folgt:

$$w_0 = w_{wire} + \frac{1}{2} \cdot \left(l_{inner} - (min_d + 2 \cdot w_{wire})\right), \tag{2-17}$$

wobei

$$min_d = minDist(Layer_{ind}) + 2 \cdot w_{wire}, \tag{2-18}$$

den Abstand zwischen den beiden Mengen der Teilwindungen darstellt (siehe Abbildung 2-6 (c)). Der Ausdruck $minDist(Layer_{ind})$ beschreibt die durch die Technologie vorgegebene Mindestdistanz zwischen Strukturen, die mit Hilfe der Maske $Layer_{ind}$ gefertigt werden. Iterativ werden nun die Teilwindungen der Spule erstellt (siehe Abbildung 2-6 (c)). Die jeweils neuen Abmessungen können rekursiv berechnet werden. Es ist

$$w_i = w_{i-1} + w_{wire} + minDist(Layer_{ind}), \tag{2-19}$$

$$l_i = l_{i-1} + 2 \cdot (minDist(Layer_{ind}) + w_{wire}), \tag{2-20}$$

wobei w_0 durch Gl. (2-17) gegeben ist und für l_0 gilt:

$$l_0 = l_{inner} + 2 \cdot w_{wire}. \tag{2-21}$$

Auf diese Weise entstehen zwei Strukturen mit N Teilwindungen, die im nächsten Schritt aneinander kompaktiert werden (siehe Abbildung 2-6 (c)), wobei ein Mindestabstand von min_d (siehe Gl. (2-18)) eingehalten wird. Abschließend erfolgt eine kreuzweise Verbindung der entsprechenden Leitungen (siehe Abbildung 2-6 (d)). Das resultierende Layout einer integrierten Induktivität nach erfolgreicher Ausführung des Generators ist der Abbildung 2-3 zu entnehmen.

Es ist das Layout für verschiedene Induktivitäten in einer 025µm CMOS Technologie der Firma IHP GmbH angefertigt worden. Die nachfolgende Tabelle 2-1 zeigt die Ergebnisse des Generators zur Berechnung der Gesamtinduktivität der jeweiligen Spulen. Um einen Vergleich zu den tatsächlichen Werten zu haben, sind die Meßergebnisse der Spulen ebenfalls in die Tabelle mit aufgenommen worden. Die Messwerte sind den Angaben der Firma IHP GmbH entnommen worden.

Tabelle 2-1: Vergleich zwischen Simulationswerten und Messwerten für verschiedene Induktivitäten

			Induktivität [nH]	
Anzahl Windungen	**Innenweite [µm]**	**Leitungsbreite [µm]**	**Laut Hersteller**	**Generator**
7	159	7.98	11.7	11.69
8	109.6	6.99	14.7	13.63
8	176.4	7.98	23.8	21.04

2.3 Widerstände

Grundlegend stehen in den Technologien vier verschiedene Arten von Widerständen zur Verfügung:

- Diffusionswiderstand
- Polysiliziumwiderstand
- Wannenwiderstand
- Filmwiderstand

Zur Erzeugung von Diffusionswiderständen werden spezielle Strukturen geformt, wie sie auch für die Erstellung von Transistor-Diffusionsgebieten genutzt werden. Es entsteht ein sogenannter Volumenwiderstand, der sich aus der Länge, Weite und Tiefe der Dotierschicht ergibt. Polysiliziumwiderstände nutzen den ohmschen Widerstand der Polyschicht aus. Sie können mittels Maskenformen auch mäanderförmig strukturiert werden. Infolge einer derartigen Strukturierung sind sehr große Widerstandswerte realisierbar. Beim vergrabenen Widerstand bildet eine relativ leicht dotierte Wanne im Substrat den Widerstandskörper [Gray93]. Die Herstellung von Dünnfilm-Widerständen setzt zusätzliche Prozess-Schritte voraus. Diese Schritte bedingen eine Sondertechnologie, so dass sie üblicherweise nicht für CMOS-Prozesse zur Verfügung steht. Bei dieser Art von Widerständen werden dünne Schichten von Tantal oder Nickel-Chrom-Verbindungen auf dem Wafer aufgebracht [Gray93].

2.3.1 Genauigkeit integrierter Widerstände

In diesem Abschnitt werden die Absolutgenauigkeit sowie die Paarungsgenauigkeit von integrierten Widerständen näher untersucht. Zu diesem Zweck ist im Folgenden ein rechteckförmiger Widerstand

zu betrachten. Weiterhin sei der Widerstand der Kontaktierungen als vernachlässigbar anzusehen. Der Widerstandswert berechnet sich wie folgt:

$$R = \frac{\bar{\rho}}{\bar{x_j}} \cdot \frac{L}{W} = R_{\square} \cdot \frac{L}{W}. \tag{2-22}$$

wobei $\bar{\rho}$ den mittleren spezifischen Widerstand, L und W die Abmessungen des Widerstands und x_j die durchschnittliche Dicke der zu betrachtenden Struktur beschreiben. Der Flächen-Widerstand (Square-Widerstand) $R_{\square}$ ist lediglich von den Prozessparametern abhängig, wenn die Weite W und die Länge L identisch sind. Generell werden Widerstände als Vielfaches des Squarewiderstands aufgefasst (siehe Abbildung 2-7).

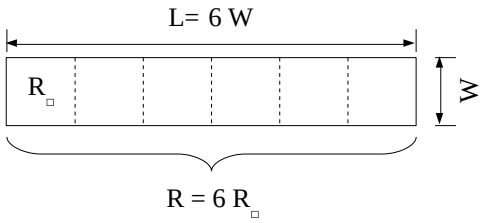


Abbildung 2-7: Berechnung des Widerstandswertes mit Hilfe des Flächenwiderstands

Häufig werden bei der praktischen Umsetzung integrierter Widerstände Strukturen mit einer Länge verwendet, die deutlich größer als die Weite ist, da in der Regel Widerstände benötigt werden, deren Wert über dem spezifischen Widerstandswert liegt. Eine Beeinflussung der Genauigkeit durch die Länge im Vergleich zur Weite ist in diesen Fällen vernachlässigbar. Insgesamt ist die Präzision der geometrischen Parameter abhängig von der Genauigkeit der Photolithographie, dem Ätzverfahren und der lateralen Ausdiffusion.

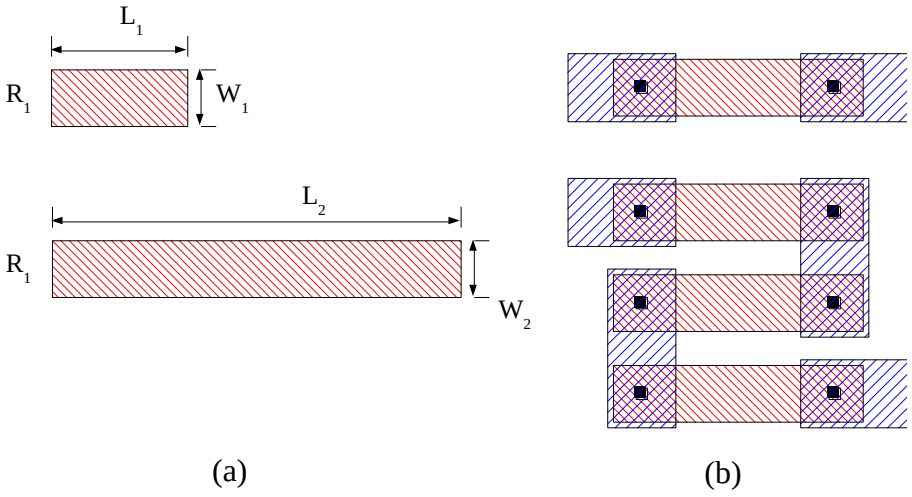


Abbildung 2-8: Beispiel zur Untersuchung der Paarungsgenauigkeit integrierter Widerstände

Ferner wird die Absolutgenauigkeit von Widerständen beeinträchtigt durch die Parameter der Prozessführung. So nehmen die voneinander unabhängigen Größen wie der Diffusionskonzentration, der Implationsdosis, der Diffusionstemperatur und der Diffusionszeit Einfluss auf die Genauigkeit. Die genannten Werte spiegeln sich in dem spezifischen Widerstandswert und der Dicke des Widerstands wider und können durch den Designer nicht direkt beeinflusst werden. Die daraus resultierenden Toleranzen können in einem Bereich von ca. 5% bis hin zu 20% auftreten. Der Absolutwert eines

Widerstands ist aus diesem Grund mit einer relativ hohen Ungenauigkeit behaftet. Hinsichtlich der Frage wie genau ein bestimmtes Widerstandsverhältnis hergestellt werden kann, ist im weiteren Verlauf die Abbildung 2-8 zu betrachten.

Es sind zwei Widerstände mit einem Verhältnis von (Abbildung 2-8 (a)) und deren mögliche Realisierungsform (Abbildung 2-8 (b)) dargestellt. Es gilt:

$$\frac{R_1}{R_2} = \frac{\overline{\rho}_1 \cdot \bar{x}_{j2}}{\overline{\rho}_2 \cdot \bar{x}_{j1}} \cdot \frac{L_1 \cdot W_2}{L_2 \cdot W_1} \approx \frac{L_1 \cdot W_2}{L_2 \cdot W_1}, \tag{2-23}$$

wobei

$$\frac{\overline{\rho}_1 \cdot \bar{x}_{j2}}{\overline{\rho}_2 \cdot \bar{x}_{j1}} \tag{2-24}$$

vom Prozess abhängig ist und

$$\frac{L_1 \cdot W_2}{L_2 \cdot W_1} \tag{2-25}$$

durch das Layout beeinflusst wird. Falls die Widerstände mit denselben Prozess-Schritten zeitgleich gefertigt werden und eine enge räumliche Nähe zueinander aufweisen, so heben sich die aus der Prozessfolge resultierenden Toleranzen weitestgehend auf. Die Paarungsgenauigkeit ist daher vom Layout abhängig. Die Wahl geeigneter Layoutstrukturen liefert eine Toleranzgrenze von besser als 1‰.

2.3.2 Einfluss der Temperatur

Die Ausrichtung von hochpräzisen Bauelementen im Hinblick auf wärmeproduzierende Schaltungsbereiche ist bei den integrierten Schaltungen ebenfalls von Bedeutung. So sind Transistoren und Widerstände temperaturabhängige Komponenten. Um eine Beeinflussung der Genauigkeit durch einen Temperaturgradienten zu minimieren, sollten die betroffenen Elemente symmetrisch zu den Wärmequellen ausgerichtet werden.

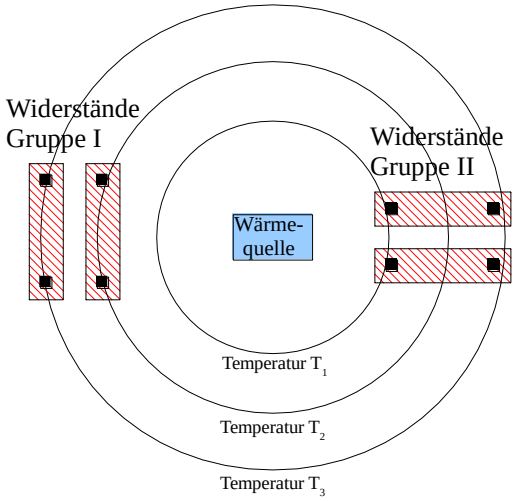


Abbildung 2-9: Ausrichtung von Widerständen zu einer Wärmequelle

Die in Abbildung 2-9 gezeigten Widerstände der Gruppe I werden eine Diskrepanz aufweisen, da beide Widerstände unterschiedlichen Temperaturbereichen ausgesetzt sind. Demgegenüber sind die Widerstände der Gruppe II symmetrisch bzgl. der Temperaturzonen ausgerichtet, so dass

Paarungsfehler bedingt durch Temperatureinflüsse in diesem Fall nahezu ausgeschlossen werden können.

2.3.3 Modulgenerator zur Erstellung von Präzisionswiderständen

In diesem Abschnitt wird die Funktionsweise des Modulgenerators zur Erstellung von Präzisionswiderständen [Zhan09] anhand von ausgewählten Beispielen beschrieben. Für die Generierung sind die Eingabe der Länge und der Weite eines einzelnen Widerstandssteifens, die Anzahl der gewünschten Zeilen und das Verhältnis der Widerstände innerhalb der Präzisionswiderstandsstruktur erforderlich. Die Gesamtanzahl an Widerstandsstreifen ermittelt der Generator automatisch mittels der Verhältnisangabe der Widerstände. Nach erfolgreicher Ausführung ergibt sich beispielsweise das in Abbildung 2-10 gezeigte Layout eines Präzisionswiderstands.

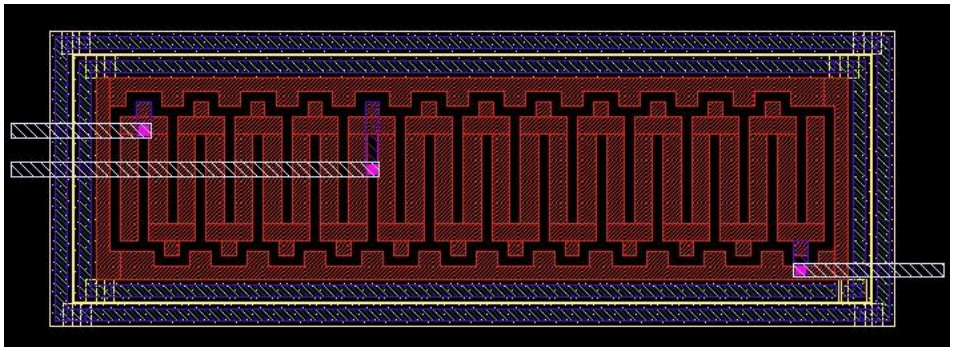


Abbildung 2-10: Layout eines Präzisionswiderstands

Die Anschlüsse der jeweiligen Widerstände werden zu einer Seite des Moduls herausgeführt. Die Verbindungen innerhalb der Widerstandsstruktur werden bei mehrzeiligen Präzisionswiderständen selbsttätig durchgeführt (siehe Abbildung 2-11).

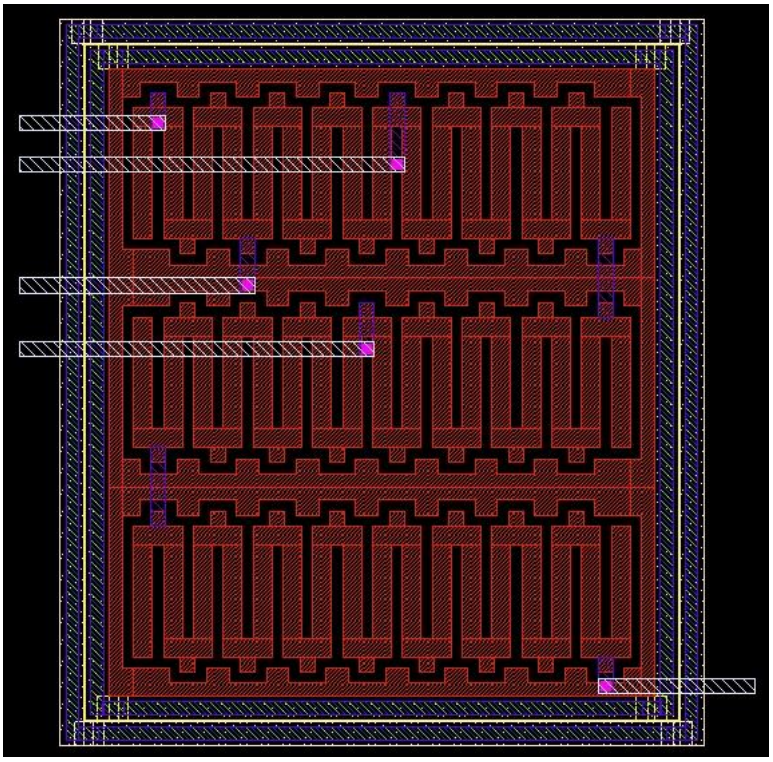


Abbildung 2-11: Layout eines mehrzeiligen Präzisionswiderstands

Im Folgenden wird die Generierung einer Zeile eines Präzisionswiderstands näher erläutert. Die Generierung für drei Widerstandsstreifen und den entsprechenden Dummy-Strukturen ist in Abbildung 2-12 dargestellt.

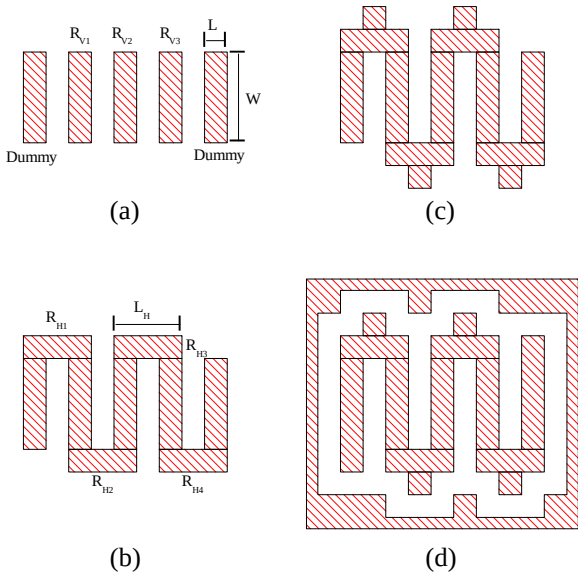


Abbildung 2-12: Abfolge zur Generierung eines Präzisionswiderstands

Der erste Schritt beinhaltet eine Kompaktierung der Widerstandsstreifen in horizontaler Richtung (Abbildung 2-12 (a)). Die jeweiligen Streifen weisen die vom Benutzer vorgegebene horizontale Länge L und die vertikale Weite W auf. Der minimale Abstand zwischen den Streifen ist durch die zu verwendende Technologie vorgegeben und wird durch den Generator automatisch berücksichtigt. Der erste und der letzte Widerstandsstreifen stellen die Dummy-Elemente dar, während die verbleibenden Elemente die aktiven Widerstände darstellen. Im anschließenden Schritt werden durch eine vertikale Kompaktierung von horizontal verlaufenden Rechtecken die einzelnen Widerstandsstreifen miteinander verbunden (Abbildung 2-12 (b)). Die Größe der herankompaktierten Rechtecke berechnet sich in horizontaler Richtung wie folgt:

$$L_H = 2 \cdot L + \Delta_{min}. \tag{2-26}$$

wobei Δ_{min} den Mindestabstand zwischen den vertikalen Streifen bezeichnet. Die Weite der horizontalen Streifen beträgt W. Auf jedem der horizontalen Verbindungselemente wird ein kleines Rechteck zentriert herankompaktiert (Abbildung 2-12 (c)), um auf diese Weise eine Anschlussmöglichkeit für spätere Metall-Leitungen zu ermöglichen. Im letzten Schritt wird die Umgebung des Präzisionswiderstands, bestehend aus entsprechenden Dummy-Elementen, erzeugt (Abbildung 2-12 (d)). Um eine hinreichend große Paarungsgenauigkeit der einzelnen Widerstände zueinander zu erzielen, ist auf einen gleichmäßigen Abstand zwischen dem Widerstand und der Dummy-Struktur zu achten.

Falls die Anzahl an Zeilen für den Präzisionswiderstand größer als eins ist, wird für jede Zeile eine Grundstruktur nach der beschriebenen Vorgehensweise erstellt. Es wird dabei jede generierte Grundstruktur in vertikaler Richtung an die vorhergehende Widerstandszeile herankompaktiert, wobei diejenigen Strukturen mit einer geraden Zeilennummer zuvor an der x-Achse gespiegelt werden. Anschließend erfolgt eine Verbindung mit Hilfe von Metall-Leitungen von denjenigen Widerständen, die sich über mehr als eine Zeile ausdehnen. Zur Entkopplung zum Substrat, werden

um die erstellte Widerstandsstruktur abschließend zwei Guardringe platziert. Es werden weiterhin die notwendigen Anschlüsse der Widerstände innerhalb der Struktur mit Hilfe von Metall-Leitungen an den Rand des Moduls herausgeführt. Das resultierende Ergebnis für einen einzeiligen bzw. einen mehrzeiligen Präzisionswiderstand ist in Abbildung 2-10 bzw. in Abbildung 2-11 dargestellt.

2.4 Kapazitäten

In Abhängigkeit von der jeweiligen Prozesstechnologie lassen sich grundsätzlich verschiedene Realisierungen für integrierte Kondensatoren beschreiben. Derartige Kondensatorstrukturen sind für analoge und Mixed-Signal Schaltungen oft unerlässlich, auch wenn nur verhältnismäßig kleine Kapazitätswerte realisiert werden können. Ein einfacher Kondensator bildet sich durch die Isolation (SiO2) zwischen einer Polysilizium- und einer Metallebene. Stehen in einer Technologie mehrere Polyebenen zur Verfügung, so kann auch mit Hilfe einer Isolationsschicht zwischen zwei Polyebenen ein Kondensator erstellt werden [Alle87]. Eine dritte Möglichkeit ist Verwendung einer Epitaxieschicht als Gegenelektrode zu einer Poly- oder Metallbahn. Ebenfalls ist eine Struktur realisierbar, die zwischen den Metallisierungsebenen angeordnet ist. Abschließend ist es ebenfalls sinnvoll alle Leiterbahnen miteinander zu verknüpfen, so dass bei gleicher Fläche nahezu eine Verdopplung des Kapazitätswertes erzielt werden kann.

2.4.1 Genauigkeit integrierter Kapazitäten

Alle Prozessparameter bei Herstellung integrierter Schaltungen unterliegen einer lokalen Variation. Diese Streuung ist zufällig über die gesamte Chipfläche verteilt und kann nicht vorherbestimmt werden. So variiert bei der Fabrikation einer integrierten Kapazität die Oxiddicke und infolge von Über- bzw. Unterätzen kommt es zu einer zufallsbedingten Änderung an den Kanten. Aus diesen beiden Faktoren ergibt sich eine Abweichung des Kapazitätswertes von dem nominellen Wert der im Layout gezeichneten Geometrien. Eine wirksame Methode diesen lokalen Prozessfehlern entgegenzuwirken besteht nach [McCr81] darin, die betroffenen Elemente so groß zu wählen, dass die genannten Effekte lediglich eine untergeordnete Rolle spielen.

Die weiteren Möglichkeiten den Einfluss der negativen Effekte zu minimieren sollen im Folgenden näher untersucht werden. Dazu sind die in Abbildung 2-13 gegebenen beiden quadratischen Kapazitäten zu betrachten, deren Kapazitätsverhältnis zueinander K_d betrage.

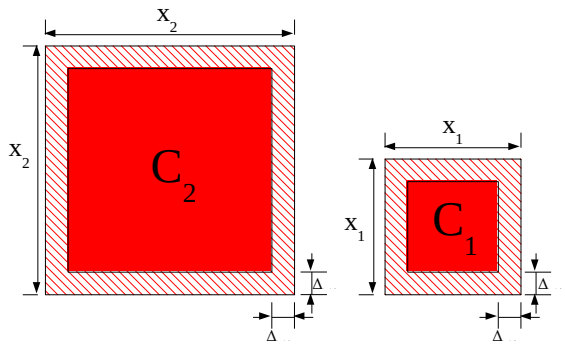


Abbildung 2-13: Beispiel zur Untersuchung der Paarungsgenauigkeit zweier quadratischer Kapazitäten

Durch die Wahl geeigneter Werte für x_1 und x_2 wird das zu erzielende Kapazitätsverhältnis im Layout eingestellt. Es gilt damit:

$$K_d = \frac{x_2^2}{x_1^2}. \tag{2-27}$$

Falls es bei den Kapazitäten zu einer Über- oder Unterätzung in der Größenordnung Δx kommt, errechnet sich das reale Verhältnis wie folgt:

$$K_r = \frac{(x_2-\Delta x)^2}{(x_1-\Delta x)^2}. \tag{2-28}$$

Der relative Fehler lässt sich dadurch berechnen nach:

$$err_K = \frac{K_d-K_r}{K_d} \approx \frac{2\Delta x}{x_2} \cdot \left(\sqrt{K_d} - 2\right). \tag{2-29}$$

Angenommen die Kapazität C_1 weist die Dimensionen 30µm × 30µm auf und die Abweichungen an den Kanten beträgt $\Delta x \approx 0.1\mu m$. Für ein Kapazitätsverhältnis der Kapazitäten C_1 und C_2 von 1:4 ergibt sich damit nach Gl. (2-29) ein Fehler von err_K=0,66%. Bei einer Anwendung der beschriebenen Kapazitäten in einem A/D- bzw. D/A- Umsetzer könnte der Fehler zu einer Abweichung von ±0,5 Bit führen. Damit ist es offensichtlich, dass die Kapazitäten in der dargestellten Form nicht für den Entwurf hochgenauer Schaltungen geeignet sind. Eine Reduzierung des Fehlers ist durch die Verwendung von Einheitskapazitäten gegeben (siehe Abbildung 2-14 (a)). In diesem Fall werden die Flächen der Kapazitäten um den gleichen Faktor reduziert, das Verhältnis zwischen den jeweiligen Kapazitäten bleibt jedoch unverändert.

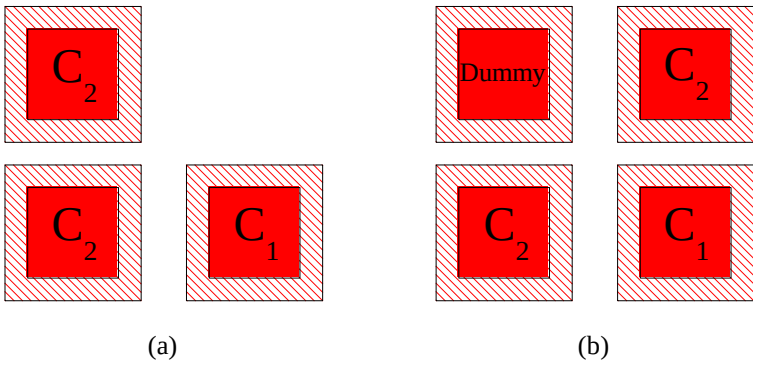


Abbildung 2-14: Verwendung von Einheitskapazitäten zur Einstellung eines Verhältnisses von 1:2

Es ist zu berücksichtigen, dass die Oxiddicke an unterschiedlichen Stellen des Wafers bzw. des Chips variiert. Im Hinblick auf die Paarungsgenauigkeit würde die Vernachlässigung dieses Gradienten bei der Verwendung von Einheitskapazitäten zu einer signifikanten Abweichung führen. Eine Verbesserung der Paarungsgenauigkeit kann in diesem Fall durch die Anwendung des Prinzips des gemeinsamen Schwerpunkts, d.h. eine konzentrische Anordnung der jeweiligen Einheitskapazitäten, erzielt werden (siehe Abbildung 2-14 (b)). Dabei ist auf eine dichte Platzierung der benachbarten Bauelemente zueinander zu achten.

Die Benutzung von Einheitskapazitäten ist nur dann möglich, wenn die Werte der Kapazitäten durcheinander teilbar sind. Andernfalls sind rechteckförmige Kapazitäten zu verwenden. In diesem Fall ist jedoch eine Abhängigkeit der Paarungsgenauigkeit gegenüber lithographischen Streuungen und dem Ätzprozess gegeben. Die Einflussnahme der Fehler kann mit Hilfe einer geeigneten geometrischen Dimensionierung reduziert werden.

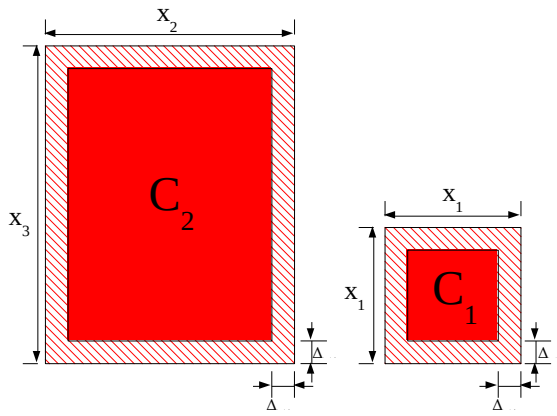


Abbildung 2-15: Beispiel zur Untersuchung der Paarungsgenauigkeit rechteckiger Kapazitäten

Das Verhältnis der in Abbildung 2-15 gezeigten Kapazitäten beträgt:

$$K_d = \frac{x_2 \cdot x_3}{x_1^2}. \tag{2-30}$$

Infolge von Abweichungen durch das Ätzen ergibt sich das nachfolgende reale Verhältnis:

$$K_r = \frac{(x_2 - \Delta x)\cdot(x_3 - \Delta x)}{(x_1 - \Delta x)^2}. \tag{2-31}$$

Für den relativen Fehler gilt damit:

$$err_K \approx \frac{2\Delta x}{x_2^2} \cdot K_d \cdot \left(2x_1 - \frac{x_1^2(x_2+x_3)}{x_2 x_3}\right). \tag{2-32}$$

Es werden nun Werte für x_2 und x_3 gesucht, so dass der relative Fehler err_K bzgl. Δx minimiert wird. Dazu ist die erste Ableitung von err_K in Abhängigkeit von Δx zu bestimmen und gleich Null zu setzen. Es ist:

$$0 = \frac{\delta err_K}{\delta \Delta x} = \frac{2x_2 x_3}{x_2^2 x_1^2} \cdot \left(2x_1 - \frac{x_1^2(x_2+x_3)}{x_2 x_3}\right) = \frac{2x_1}{x_2+x_3} - \frac{x_1^2}{x_2 x_3}. \tag{2-33}$$

Der Term $2x_1/(x_2 + x_3)$ beschreibt das Umfangsverhältnis der beiden Kapazitäten, während der Term $x_1^2/(x_2 x_3)$ das Flächenverhältnis repräsentiert. Damit Gl. (2-33) erfüllt ist, muss das Umfangsverhältnis dem Flächenverhältnis entsprechen. Das Auflösen der Gl. (2-33) zur Bestimmung der optimalen Dimensionierung liefert:

$$x_2^2 - 2K_d x_2 + K_d = 0 \tag{2-34}$$

und damit gilt

$$x_2 = K_d \pm \sqrt{K_d(K_d - 1)}. \tag{2-35}$$

Eine Einschränkung ist aus Gl. (2-35) ersichtlich. Um eine verwertbare Lösung zu erhalten, ist für K_d ein Verhältnis größer als Eins zu wählen. Das bedeutet, dass die kleinste Kapazität als quadratische Einheitskapazität geformt werden sollte.

2.4.2 Modulgenerator zur Erstellung eines binären Kapazitätsfeldes

Zur Erstellung eines binären Kapazitätsfeldes wird im ersten Schritt des Modulgenerators eine quadratische Einheitskapazität generiert. Die Abmessungen und die Masken, die zur Konstruktion der Kapazitäten verwendet werden, sind durch den Benutzer frei wählbar. Bei den Abmessungen wird von Seiten des Generators auf eine automatische Einhaltung der Design-Regeln geachtet. Auf diese Weise können beispielsweise Poly-Poly, Poly-Diffusion oder Metall-Metall Kapazitäten erzeugt werden. Der Aufbau ist dennoch gewissen Einschränkungen unterlegen, da sich nicht alle Realisierungsformen für die Kapazitäten im Hinblick auf die zu verwendenden Masken in jeder Technologie umsetzen lassen.

Aufgrund der Tatsache, dass das Ätzen bei der Herstellung nicht zu vernachlässigende Problem mit sich bringen kann, insbesondere in Bezug auf die Ecken der Kapazitäten, werden diese bei der oberen Maske mit 45° Kanten versehen. Um eine Verbindung mit später Verdrahtung zu ermöglichen, wird in der Mitte der Einheitskapazität ein Kontakt erstellt. Zur Gewährleistung einer einheitlichen Umgebung für jede Einheitskapazität, werden um das Feld der Kapazitäten Dummy-Strukturen aufgebaut. Die Verbindung der unteren Elektrode der Kapazitäten erfolgt in dem Material der angegebenen Maske, während die oberen Elektroden unter Verwendung von Metall-Leitungen zusammengeschlossen werden.

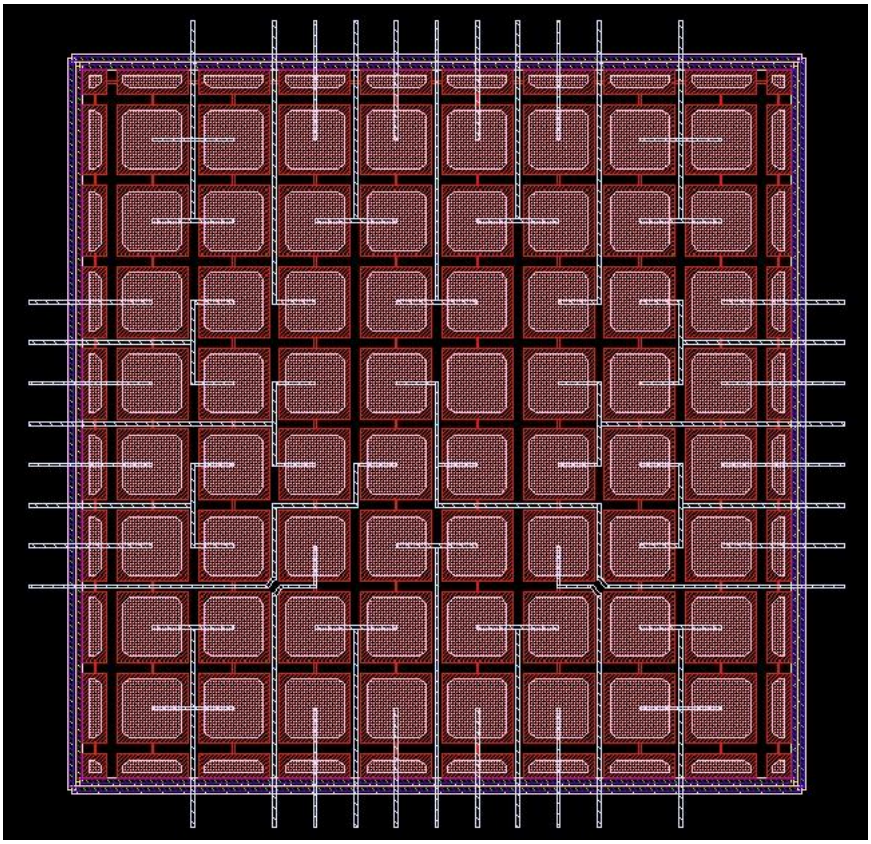


Abbildung 2-16: Automatisch generiertes Layout eines 6-Bit Kapazitätsfeldes

Insgesamt kann ein binäres Kapazitätsfeld für 2N Kapazitätsanteile ($1 \leq N \leq 6$) generiert werden. Die 1-Bit Kapazität befindet sich im Zentrum des Moduls. Die nachfolgenden Kapazitätsanteile werden aufsteigend hinsichtlich des Bit-Wertes konzentrisch um die 1-Bit Kapazität angeordnet. Auf diese Weise wird die mögliche Einflussnahme durch Schwankungen der Oxid-dicke und einer resultierenden Paarungsungenauigkeit mit berücksichtigt. Um ferner den Abstand zwischen den

Kapazitäten minimal zu halten, ist der Verlauf von maximal einer Leitung zwischen den Einheitskapazitäten erlaubt. Weiterhin ist es nicht gestattet, Leitungen über die Kapazitäten zu verlegen und es darf zu keiner Kreuzung von Verbindungen verschiedener Kapazitätsanteile innerhalb des Moduls kommen. Mit diesen Maßnahmen soll der Einfluss von Streukapazitäten auf die Paarungsgenauigkeit minimiert werden. Unter diesen Vorgaben können lediglich die 1- und die 2-Bit Kapazitäten innerhalb des binären Kapazitätsfeldes vollständig verdrahtet werden. Eine kreuzungsfreie Verbindung der Kapazitäten für 3- bis 6-Bit ist innerhalb des Feldes nicht möglich, so dass die Leitungen der Teilfelder nach außen geführt werden. Der Abbildung 2-16 ist das fertige Layout eines 6-Bit Kapazitätsfeldes zu entnehmen. Zur Reduzierung von Störeinflüssen aus dem Substrat sind zwei Guardringe um das binäre Kapazitätsfeld automatisch erstellt worden.

2.4.3 Modulgenerator zur Erstellung von gepaarten Kapazitäten

Der im Nachfolgenden beschriebene Modulgenerator ermöglicht die Konstruktion gepaarter Kapazitäten, die nicht mit einem binären Kapazitätsfeld, bestehend aus einer angemessenen Anzahl an Einheitskapazitäten, erstellt werden können. Als Eingabeparameter stehen die Anzahl der Kapazitäten und eine Liste der korrespondierenden Kapazitätswerte zur Verfügung. Aus dieser durch den Nutzer vorgegebenen Liste der Kapazitätswerte $C_1 \ldots C_n$ wird vor der Generierung des Moduls das Minimum ermittelt. Dieser im Folgenden als C_{min} bezeichnete Wert dient zum einen zum Aufbau der quadratischen Einheitskapazität, die mit Hilfe dieses Kapazitätswertes erstellt wird, zum anderen wird damit das Verhältnis zu den verbleibenden Kapazitätswerten berechnet. Ohne Beschränkung der Allgemeinheit sei im Folgenden $C_i = C_{min}$, dann gilt:

$$\begin{aligned} \eta_1 &= \frac{C_1}{C_{min}} \\ &\vdots \\ \eta_{i-1} &= \frac{C_{i-1}}{C_{min}} \\ \eta_i &= 1 \\ \eta_{i+1} &= \frac{C_{i+1}}{C_{min}} \\ &\vdots \\ \eta_n &= \frac{C_n}{C_{min}} \end{aligned} \tag{2-36}$$

Die Abmessungen der quadratischen Einheitskapazität berechnen sich unter Verwendung des Kapazitätswertes C_{sheet}, der den Kapazitätsbelag pro Flächeneinheit in Abhängigkeit von der einzusetzenden Technologie beschreibt, wie folgt:

$$L_{min} = W_{min} = \sqrt{\frac{C_{min}}{C_{sheet}}}. \tag{2-37}$$

Die Länge und die Breite der verbleibenden Kapazitäten werden durch die nachstehenden Funktionen berechnet:

$$\begin{aligned}
L_1 &= L_{min} \cdot \left(\eta_1 - \sqrt{\eta_1^2 - \eta_1}\right) \\
W_1 &= W_{min} \cdot \left(\eta_1 + \sqrt{\eta_1^2 - \eta_1}\right) \\
&\vdots \\
L_{i-1} &= L_{min} \cdot \left(\eta_{i-1} - \sqrt{\eta_{i-1}^2 - \eta_{i-1}}\right) \\
W_{i-1} &= L_{min} \cdot \left(\eta_{i-1} - \sqrt{\eta_{i-1}^2 - \eta_{i-1}}\right) \\
&\vdots \\
L_n &= L_{min} \cdot \left(\eta_n - \sqrt{\eta_n^2 - \eta_n}\right) \\
W_n &= L_{min} \cdot \left(\eta_n - \sqrt{\eta_n^2 - \eta_n}\right)
\end{aligned} \quad . \tag{2-38}$$

Zur Gewährleistung einer einheitlichen Umgebung für jede Kapazität werden entsprechende Dummy-Strukturen erstellt. In der Abbildung 2-17 ist das fertige Layout für drei gepaarte Kapazitäten dargestellt. Zur Reduzierung der Folgen durch das Ätzen werden die Ecken der oberen Maskenebene der Kapazitäten mit 45° Abschrägungen versehen. Ferner erhält jede Kapazität, die nicht als Dummy-Struktur dient, einen mittig platzierten Kontakt zum Anschluss von Metall-Leitungen. Die Verdrahtungen der jeweiligen Kapazitäten werden kreuzungsfrei zum Rand des Moduls herausgeführt. Anschließend erfolgt die Konstruktion zweier Guardringe um das Kapazitätsfeld.

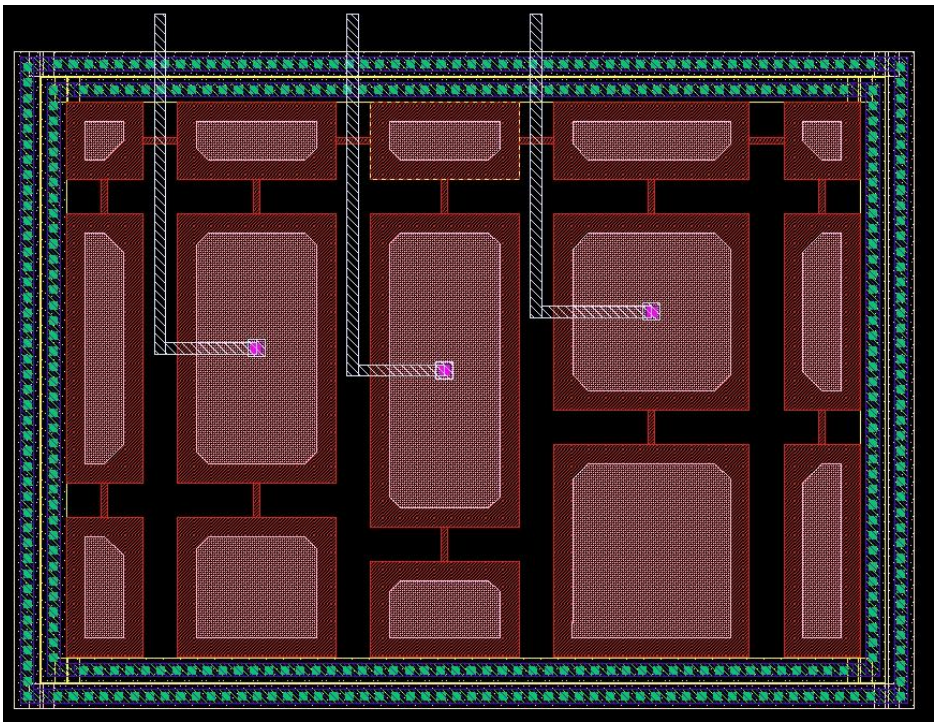


Abbildung 2-17: Automatisch erstelltes Layout von 3 gepaarten Kapazitäten

3 Operationsverstärker und Komparatoren

Ein Operationsverstärker besteht aus einer differentiellen Eingangsstufe, eventuell einer Zwischenstufe und einer Ausgangsstufe. Die Stufen sind häufig mittels einer Kompensation verbunden. Ebenso ist ein Bias-Netzwerk erforderlich. Ein großer Teil der integrierten Operationsverstärker wird für die SC-Schaltungstechnik eingesetzt. Weitere Einsatzgebiete stellen beispielsweise die gmC-Filter dar. Die erste Stufe eines Operationsverstärkers ist stets eine Differenzstufe. Falls die Anforderungen an die Verstärkung höher ausfallen bzw. andere Spezifikationsgrößen nicht eingehalten werden können, so müssen weitere Stufen eingebaut werden. Ein Überblick zum Design von Operationsverstärkern ist durch die Artikel in [Gray80], [Gray82], [Gray93] und [Lake94] gegeben.

Ein Komparator ist ein Operationsverstärker ohne Frequenzkompensation, weist jedoch eine ähnlich hohe Verstärkung auf. Infolge der fehlenden Kompensation ist ein rückgekoppelter Betrieb nicht möglich, sondern der Komparator darf nur offen zum Vergleich zweier Signale verwendet werden. Andernfalls besteht die Gefahr, dass die resultierende Schaltung instabil wird.

In diesem Kapitel werden die im Rahmen der vorliegenden Arbeit entworfenen integrierten Schaltungen näher beschrieben. Das Layout der Zellen der Verstärker und der Komparatoren wurde mit Hilfe der Modulgeneratorumgebung erstellt, wobei die Module in MOGLAN beschrieben sind. Es sind für einige der vorgestellten Operationsverstärker Prototypen in einer 0.25µm CMOS Technologie der IHP GmbH angefertigt worden. Die Messergebnisse dieser Schaltungen werden in diesem Kapitel ebenfalls präsentiert.

3.1 Zweistufige Operationsverstärker

Ein zweistufiger Verstärker setzt sich grundlegend aus einer Differenzstufe und einer Inverterstufe zusammen. Anhand der Schaltung in Abbildung 3-1 kann die generelle Funktionsweise eines zweistufigen CMOS-Operationsverstärkers verdeutlicht werden.

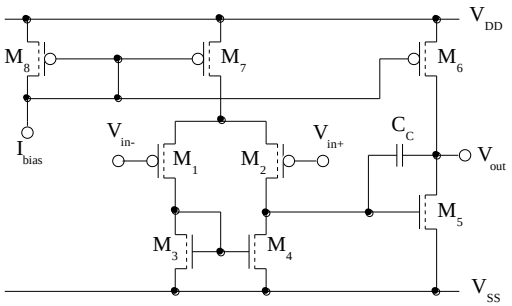


Abbildung 3-1: Zweistufiger CMOS-Operationsverstärker

Die beiden Transistoren M_1 und M_2 bilden mit dem Stromquellentransistor M7 eine Differenzstufe, deren Drainströme durch den Stromspiegel M_3 und M_4 den zweiten Verstärkertransistor M_5 ansteuern. Die Ausgangsspannung greift man ohne spezielle Endstufe am Arbeitswiderstand des Transistors M_5, nämlich dem Stromquellentransistor M_6, ab. Aus Symmetriegründen sind für die beiden Differenzverstärkertransistoren M_1, M_2 sowie für die Stromspiegeltransistoren M_3, M_4 jeweils identische Dimensionierungen zu wählen. Damit sich im Ruhezustand für V_{in-}= V_{in+} = $V_{B/2}$ auch V_{out}=$V_{B/2}$ einstellt, müssen die Drain-Ströme der Transistoren M_5 und M_6 gleich sein. Die

Leerlaufverstärkung ist mit einem geringen Aufwand durch eine Multiplikation der Einzelverstärkungen der beiden Stufen hinreichend genau anzugeben:

$$\frac{V_o}{V_i}(0) \approx \frac{g_{m1}}{g_{ds2}+g_{ds4}} \cdot \frac{g_{m5}}{g_{ds5}+g_{ds6}}. \tag{3-1}$$

Insgesamt weist die Übertragungsfunktion für eine solche Struktur zwei Polstellen auf. Eine Stabilität kann nur dann gewährleistet werden, wenn der Frequenzabstand zwischen den beiden Polen hinreichend groß ist. Bei einem zweistufigen Verstärker liegen die beiden Polstellen jedoch relativ dicht beieinander, so dass die Einführung einer Kompensation notwendig ist [Malo01]. Hierbei wird eine Kompensationskapazität zwischen dem Eingang und dem Ausgang der zweiten Stufe eingefügt.

Die einzelne Kapazität ist dennoch nicht ausreichend, um eine Frequenzkompensation zu erzielen, da diese in der Übertragungsfunktion eine Nullstelle auf der rechten Seite der s-Ebene zur Folge hat. Die aus der Nullstelle resultierende negative Phasen-Verschiebung kann die zweite Polstelle maskieren und es können Oszillationen auftreten [Gray80].

Um das Problem der Nullstelle in der rechten Seite der s-Ebene zu lösen, stehen drei verschiedene Techniken zur Verfügung, die anhand der nachfolgenden drei entworfenen Operationsverstärker näher beschrieben werden.

3.1.1 Operationsverstärker mit Zero-Nulling Widerstand

Das Schaltbild des zweistufigen Operationsverstärkers, der das Prinzip des Zero-Nulling Widerstands zur Frequenzkompensation nutzt, ist in Abbildung 3-2 gezeigt.

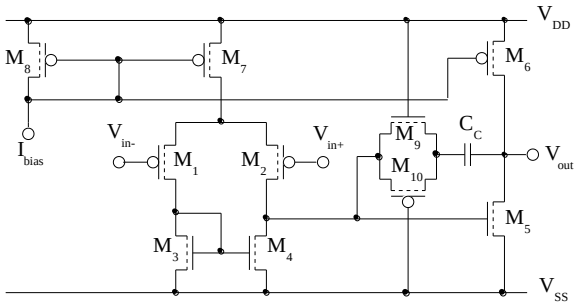


Abbildung 3-2: Schaltbild des Operationsverstärkers mit Zero-Nulling Widerstand

Die erste Stufe des Verstärkers besteht aus einer p-Kanal Differenzstufe (M_1, M_2) sowie der zugehörigen aktiven Last (M_3, M_4). Die zweite Stufe wird durch den n-Kanal Transistor M_5 und den p-Kanal Transistor M_6 realisiert. Die Stromversorgung des Verstärkers erfolgt mit Hilfe einer Stromspiegelschaltung, bestehend aus den p-Kanal Transistoren M_6 - M_8. Das Verhältnis dieser Transistoren lautet 1:1:10. Der Zero-Nulling Widerstand ist mittels eines Passgates (M_9, M_{10}) umgesetzt worden und bildet mit der in Reihe geschalteten Kapazität C_C das Frequenzkompensationsnetzwerk. Der Widerstand ermöglicht eine Verschiebung der durch die Kapazität entstehenden Nullstelle. Eine Verschiebung der Nullstelle in die linke Seite der s-Ebene ermöglicht eine Erhöhung der Bandbreite des Operationsverstärkers, während ein Aufeinanderliegen der Nullstelle und der zweiten Polstelle zu einer Kompensation des zweiten Pols führen kann. Auf diese Weise wird das gesamte System einem Ein-Pol-System angenähert.

3.1.1.1 Layout des Operationsverstärkers mit Zero-Nulling Widerstand

Das Layout des Operationsverstärkers mit Zero-Nulling Widerstand zur Frequenzkompensation ist in der Abbildung 3-3 gezeigt. Die Fläche des in einer 0.25µm CMOS Technologie erstellten Verstärkers beträgt 400 × 188µm^2. Der überwiegende Teil der Layoutfläche ist durch den Stromspiegel, bestehend aus den Transistoren M_6-M_8, und der Kompensationskapazität belegt.

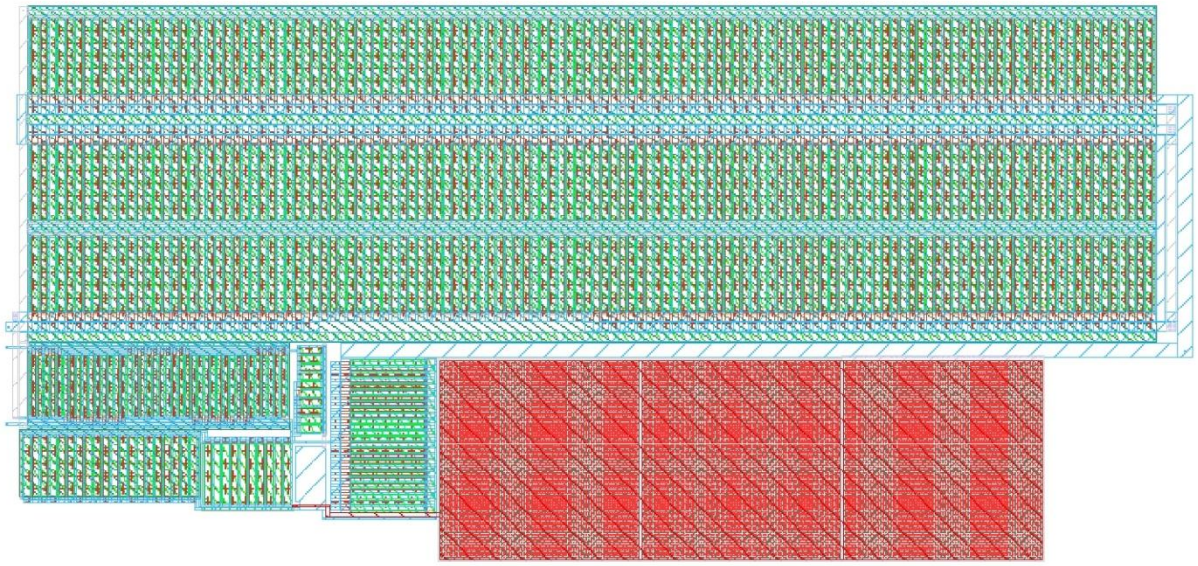


Abbildung 3-3: Layout des Operationsverstärkers aus Abbildung 3-2

3.1.1.2 Simulationsergebnisse des Operationsverstärkers mit Zero-Nulling Widerstand

Die Ergebnisse einer Postlayout-Simulation für eine Lastkapazität von 8 pF sind in Tab. 3-1 zusammengefasst. Bei einer Versorgungsspannung von 3 V kann eine Verstärkung von 92 dB und eine Bandbreite von 46 MHz erzielt werden. Der simulierte Phasengang und die Verstärkungskennlinie sind der Abbildung 3-4 zu entnehmen.

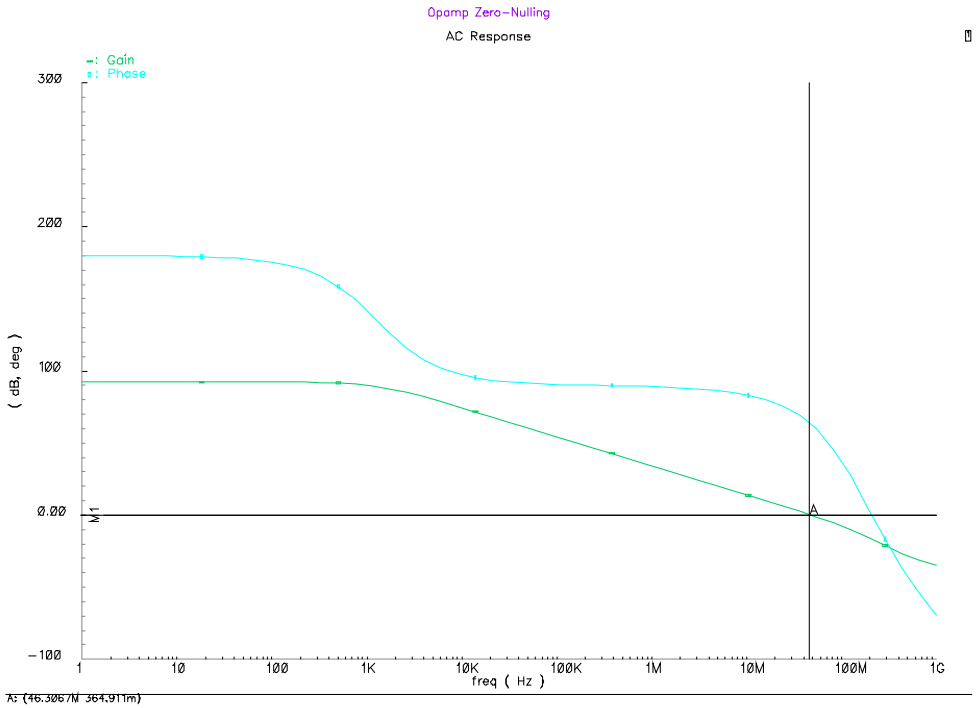


Abbildung 3-4: Bode Diagramm des Operationsverstärkers aus Abbildung 3-2

Tabelle 3-1: Simulationsergebnisse für den Operationsverstärker aus Abbildung 3-2

Parameter	Simulationsergebnis	Einheiten
Leerlaufspannungsverstärkung	92	dB
Bandbreite	46	MHz
Phasenreserve	63	°
Positive Slew Rate	24	V/µs
Negative Slew Rate	30	V/µs

3.1.2 Operationsverstärker mit Eins-Stromverstärker

In Abbildung 3-5 ist das Schaltbild des zweistufigen Operationsverstärkers mit einem Eins-Stromverstärker zur Frequenzkompensation dargestellt.

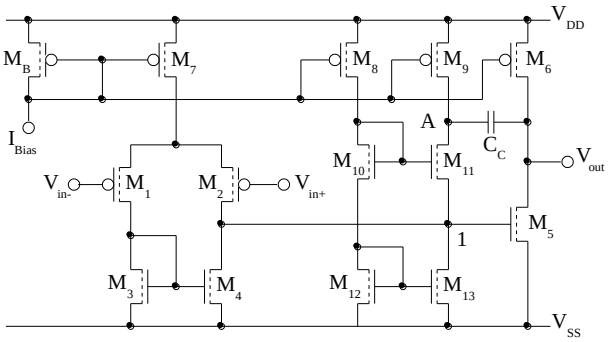


Abbildung 3-5: Schaltbild des Operationsverstärkers mit Eins-Stromverstärker

Die erste Stufe des Verstärkers ist eine p-Kanal Differenzstufe, welche Gleichtaktspannungen unterdrückt und Differenzspannungen an den Transistoren M_1 und M_2 verstärkt. Die aktive Last der Differenzstufe wird mittels der Transistoren M_3 und M_4 realisiert. Die zweite Stufe besteht aus dem n-Kanal-Transistor M_5 und dem zugehörigen Last-Transistor M_6. Die Transistoren M_B, M_6, M_7, M_8 und M_9 bilden einen Stromspiegel, um einen ordnungsgemäßen Betrieb der Schaltung sicherzustellen. Ferner weist die zweite Stufe des Verstärkers einen zehnfach größeren Biasstrom als die erste Stufe auf. Die Kompensation wird durch die Kapazität C_C und die anpassenden Stromquellen M_8, M_9 und M_{12}, M_{13} sowie den Kaskodentransistoren M_{10} und M_{11} erreicht.

Die Stromquellen kompensieren den Strom, der in den Knoten A injiziert wird. Infolge der niedrigen Impedanz an seinem Source-Anschluss liefert der Transistor M_{11} einen Signalstrom an den Knoten 1 und dient auf diese Weise als ein Strom-Buffer.

Ein weiterer Vorteil der präsentierten Schaltung besteht darin, dass aufgrund der verwendeten Kompensationsstruktur das System sich wie ein Ein-Pol-System verhält, obwohl zwei Polstellen vorhanden sind. Dies hat eine bessere Betriebsspannungsunterdrückung zur Folge [Ahuj83]. Das Frequenzkompensationskonzept sowie das Modell der Betriebsspannungsunterdrückung ist der Abbildung 3-6 zu entnehmen.

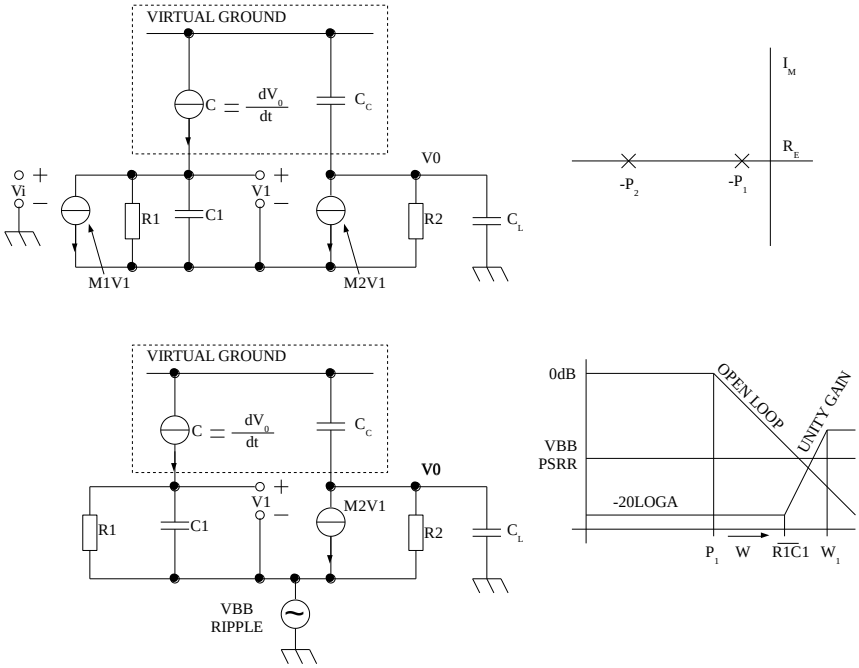


Abbildung 3-6: Frequenzkompensationskonzept, Lage der Polstellen, Kleinsignalmodell der Betriebsspannungsunterdrückung und korrespondierende Frequenzantwort

Der verhältnismäßig geringe Eingangsleitwert am Source-Anschluss des Transistors M_{11} kann jedoch zu auch zu einem Pol-Nullstellen-Doublet bei hoher Frequenz führen. Um diesem Effekt entgegenzuwirken, ist es notwendig das Pol-Nullstellen-Doublet in einen Frequenzbereich zu verschieben, der oberhalb der Transitfrequenz liegt. Dies kann durch eine Vergrößerung der Steilheit des Transistors M_{11} erzielt werden [Malo01].

3.1.2.1 Layout des Operationsverstärker mit Eins-Stromverstärker

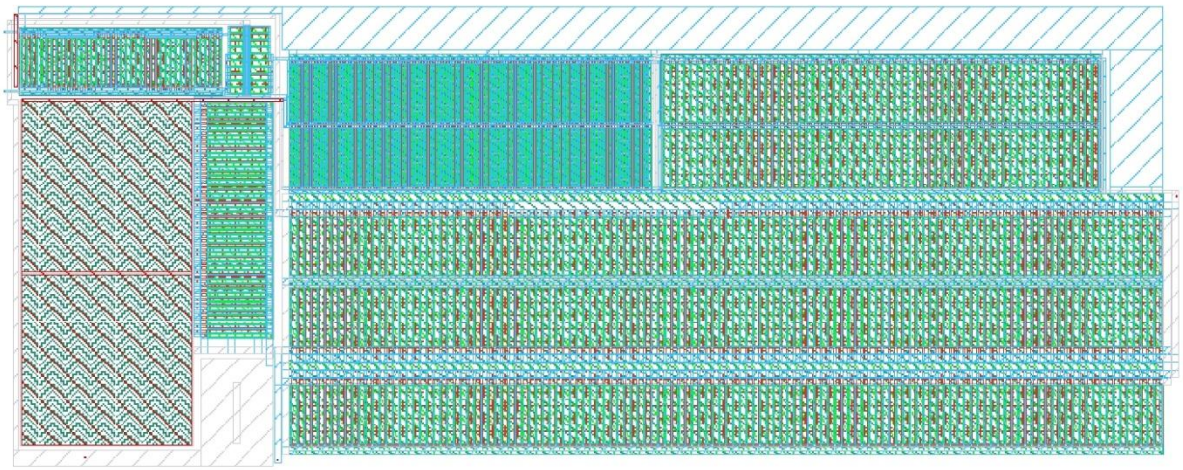


Abbildung 3-7: Layout des Operationsverstärkers aus Abbildung 3-5

Das Layout des Operationsverstärkers mit Eins-Stromverstärker zur Frequenzkompensation ist in Abbildung 3-7 dargestellt. Für die Layouterstellung ist eine 0.25µm CMOS Technologie der Firma IHP GmbH verwendet worden. Die Fläche des Verstärkers beträgt 510 × 200µm^2.

3.1.2.2 Messergebnisse des Operationsverstärker mit Eins-Stromverstärker

Die Messungen des Operationsverstärkers sind bei einer Betriebsspannung von ±1.5 V durchgeführt worden. Die Messergebnisse sind in Tab. 3-2 zusammengefasst. Es wurden eine Verstärkung von 81 dB und eine Bandbreite von 50 MHz ermittelt. In Abbildung 3-8 sind der aus der Messung resultierende Phasengang und die Verstärkungskennlinie dargestellt.

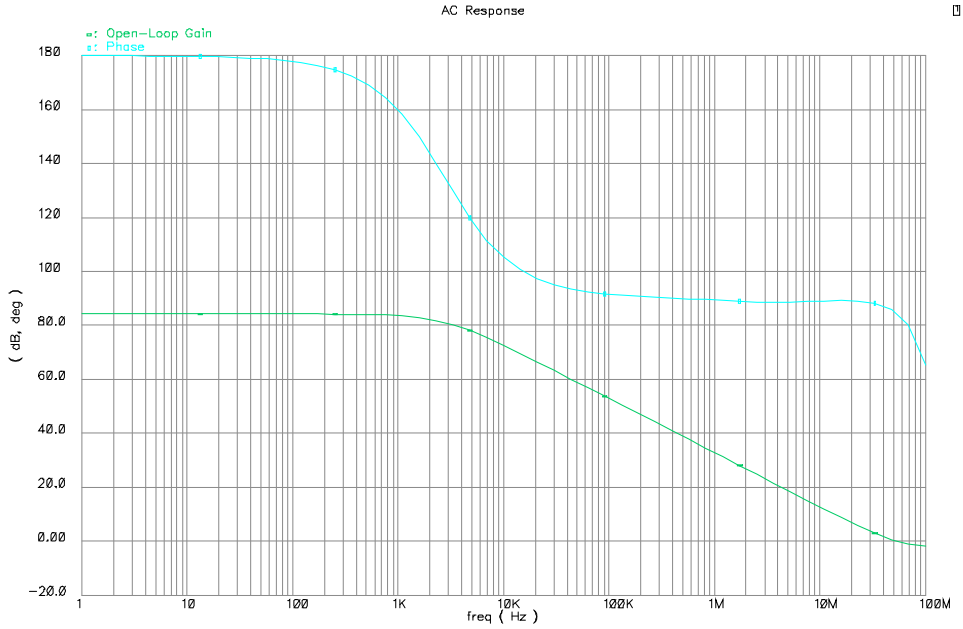


Abbildung 3-8: Bode Diagramm des Operationsverstärkers aus Abbildung 3-5

3.1.3 Operationsverstärker mit Eins-Verstärker

Der Aufbau des zweistufigen Operationsverstärkers mit einem Source-Folger zur Frequenzkompensation ist in Abbildung 3-9 dargestellt.

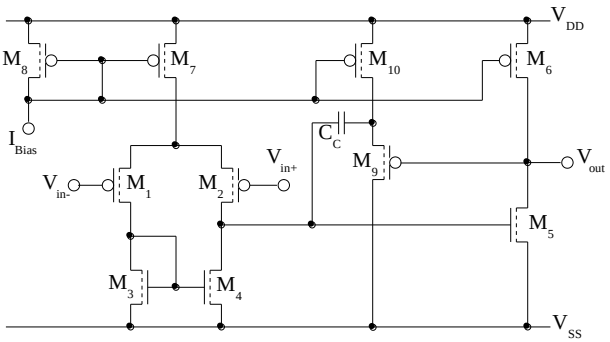


Abbildung 3-9: Schaltbild des Operationsverstärkers mit Eins-Verstärker

Tabelle 3-2: Messergebnisse des Operationsverstärkers mit Eins-Stromverstärker

Parameter	Messergebnisse	Einheiten
Versorgungsspannung	±1.5	V
Leerlaufspannungsverstärkung	81	dB
Bandbreite	50	MHz
Phasenreserve	80	°
Eingangs-Offset-Spannung	-1.3	mV
Ausgangshub	±1.2	V
Eingangshub	±1.5	V

CMRR	> 71	dB
PSRR+ bei		
1 kHz	64	dB
10 kHz	70	dB
100 kHz	52	dB
PSRR- bei		
1 kHz	65	dB
10 kHz	70	dB
100 kHz	52	dB
Positive Slew Rate	3.7	V/µs
Negative Slew Rate	3.9	V/µs
I_{Bias}	250	µA
Fläche (Länge * Weite)	510 * 200	µm²

Die Transistoren M_1 und M_2 bilden mit den Stromquellentransistor M_7 die Eingangsdifferenz-stufe. Mit Hilfe des Stromspiegels, bestehend aus den Transistoren M_3 und M_4, wird die aktive Last der Differenzstufe realisiert. Die Ausgangsstufe wird durch den Transistor M_5 und dessen Ausgangswiderstand M_6 geformt. Zur Lösung des Nullstellenproblems wird zwischen die Kompensationskapazität C_C und dem Ausgangsknoten ein Einsverstärker in Form des Source-Folgers M_9 geschaltet. Dadurch wird der Strom von Ausgangsknoten zur Kompensationskapazität durch den hinzugefügten Treiber bereitgestellt. Ein Nachteil dieser Form der Frequenzkompensation besteht darin, dass der Einsverstärker den erzielbaren Aussteuerungsbereich des zweistufigen Operationsverstärkers einschränkt. Wenn die Eingangsspannung am Source-Folger unter einen Wert von $U_{th}+2U_{ds,sat}$ absinkt, arbeitet der Transistor im linearen Bereich und ermöglicht keinen weiteren ordnungsgemäßen Betrieb [Malo01]. Ein weiterer zu beachtender Punkt betrifft das Kompensationsnetzwerk bei hohen Frequenzen. Hier kann es zu einem Pol-Nullstellen-Doublet kommen und damit zu einer Verringerung der Phasenreserve.

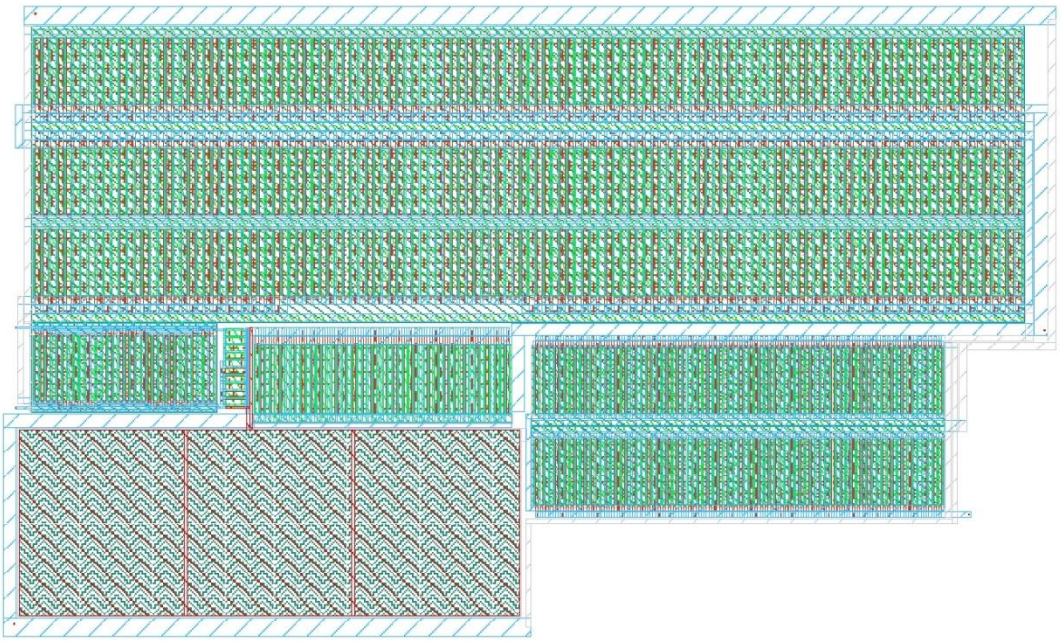


Abbildung 3-10: Layout des Operationsverstärkers aus Abbildung 3-9

3.1.3.1 Layout des Operationsverstärkers mit Eins-Verstärker

Das Layout des Operationsverstärkers mit Eins-Verstärker zur Frequenzkompensation ist in Abbildung 3-10 gezeigt. Der Verstärker ist einer 0,25µm CMOS Technologie von IHP GmbH entworfen und gefertigt worden. Der Flächenbedarf des Operationsverstärkers beläuft sich auf 400 × 240µm^2.

3.1.3.2 Messergebnisse des Operationsverstärkers mit Eins-Verstärker

Die Messergebnisse für den Operationsverstärker mit Eins-Verstärker zur Frequenzkompensation sind in Tab. 3-3 zusammengefasst. Die Messungen sind bei einer Versorgungsspannung von ±1.5 V durchgeführt worden. Der Verstärker erreicht eine Leerlaufspannungsverstärkung von 91 dB und eine Bandbreite von 70 MHz. Die Kennlinien für die Verstärkung und den Phasengang sind der Abbildung 3-11 zu entnehmen

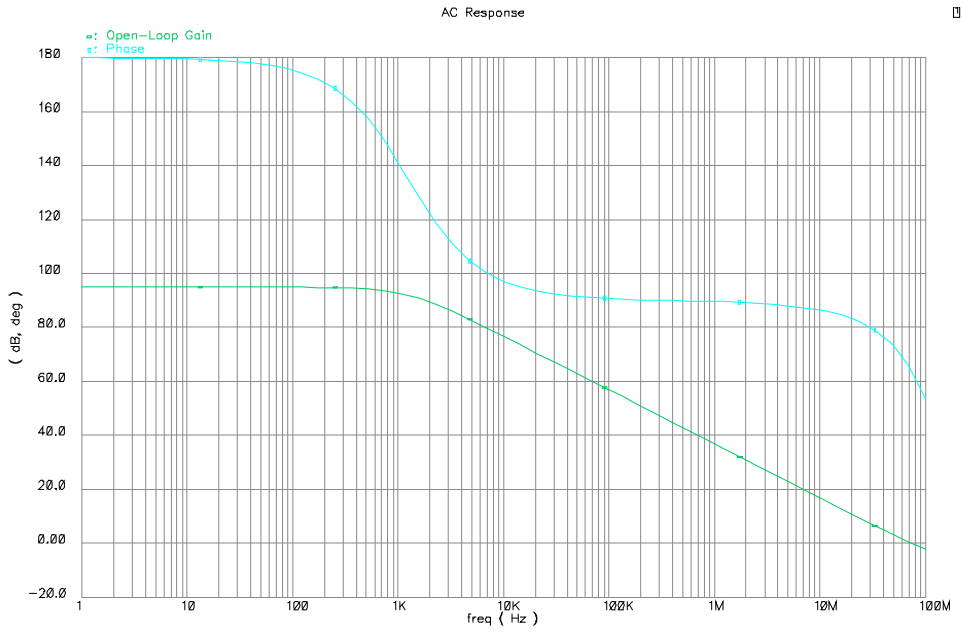


Abbildung 3-11: Bode Diagramm des Operationsverstärkers aus Abbildung 3-9

Tabelle 3-3: Messergebnisse des Operationsverstärkers mit Eins-Verstärker

Parameter	Messergebnisse	Einheiten
Versorgungsspannung	±1.5	V
Leerlaufspannungsverstärkung	91	dB
Bandbreite	70	MHz
Phasenreserve	64	°
Eingangs-Offset-Spannung	0.9	mV
Ausgangshub	±1.3	V
Eingangshub	±1.5	V
CMRR	> 74	dB
PSRR+ bei		
1 kHz	26	dB
10 kHz	45	dB
100 kHz	61	dB
PSRR- bei		
1 kHz	61	dB
10 kHz	70	dB
100 kHz	54	dB
Positive Slew Rate	4	V/µs

Negative Slew Rate	4.3	V/µs
I_{Bias}	450	µA
Fläche (Länge * Weite)	400 * 240	µm²

3.2 Zweistufiger Kaskoden-Operationsverstärker

Es werden das Design und die Messergebnisse eines zweistufigen Kaskoden-Operationsverstärkers mit zwei Eingangsstufen und einer verschachtelten Miller-Kompensation [Esch95] beschrieben. Die Betriebsspannungsunterdrückung im hochfrequenten Bereich kann durch die Verwendung der Kaskode in der Eingangsstufe verbessert werden, erschwert jedoch die Frequenzkompensation des Verstärkers [Ribn84]. Neben dem Aufbau des Operationsverstärkers und dessen Kleinsignalmodell werden nachfolgend das Layout und die Ergebnisse der Messungen des Verstärkers präsentiert.

3.2.1 Design des Kaskoden-Operationsverstärkers

Das Schaltbild des Kaskoden-Operationsverstärkers ist der Abbildung 3-12 zu entnehmen. Die grundlegende Idee basiert auf einen Operationsverstärker, der erstmals in [Ribn84] vorgestellt wurde. Durch die Verwendung einer weiteren Eingangsstufe ist es möglich, die Gleichspannungsverstärkung des Operationsverstärkers weiter zu erhöhen. Das Problem der Frequenzkompensation wird mittels einer verschachtelten Miller-Kompensation [Esch95] gelöst.

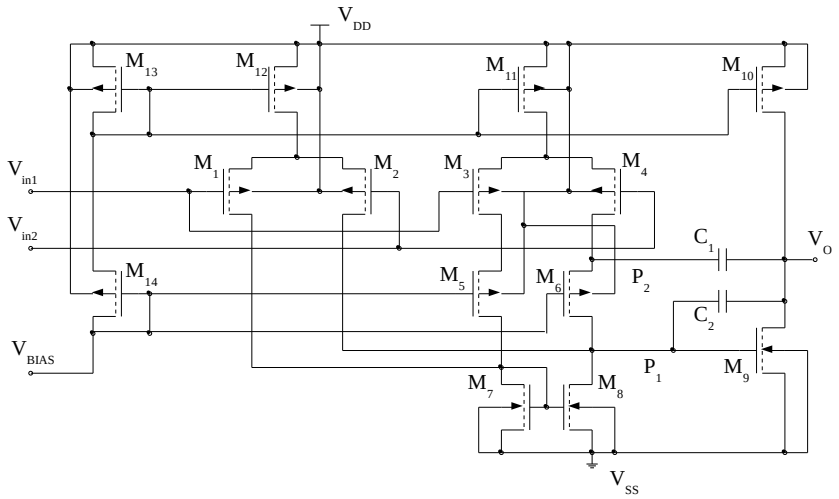


Abbildung 3-12: Schaltbild des Kaskoden-Operationsverstärkers

Aus dem Schaltbild des Verstärkers ist ersichtlich, dass sich dieser aus zwei p-Kanal Differenz-stufen, die durch die Transistoren M_1, M_2 und M_3, M_4 gebildet werden und einer p-Kanal Kaskode zusammensetzt. Die Verwendung der Kaskodentransistoren M_5 und M_6 ermöglicht es, eine Verbindung des Ausgangs mit dem Source des Transistors M_6, unter Verwendung der Kompensationskapazität C_1, herzustellen. Infolge des geringen Eingangswiderstands der Kaskode, kann auf den für gewöhnlich verwendeten Kompensationswiderstand verzichtet werden. Die Ausgangsstufe wird durch den n-Kanal Transistor M9 und den p-Kanal Transistor M_{10}, der Teil des Stromspiegels ist, bestehend aus den Transistoren M_{10}-M_{13}, realisiert. Die Anwendung des Prinzips der verschachtelten Miller-Kompensation gestattet eine Verbindung des Gates des Transistors M_9 mit dem Ausgang unter Benutzung der Kompensationskapazität C_2.

3.2.2 Kleinsignalmodell des Kaskoden-Operationsverstärkers

Um den vorliegenden Operationsverstärker zu entwerfen und dessen Leerlauffrequenzgang zu bestimmen, ist es sinnvoll mit einem geeigneten Kleinsignalmodell zu arbeiten. Für diesen Zweck ist

das in Abbildung 3-13 gezeigte Kleinsignalersatzschaltbild erstellt worden. Hierbei sind einige Elemente zusammengefasst und diverse Elemente vernachlässigt worden, um die Komplexität des Modells zu reduzieren.

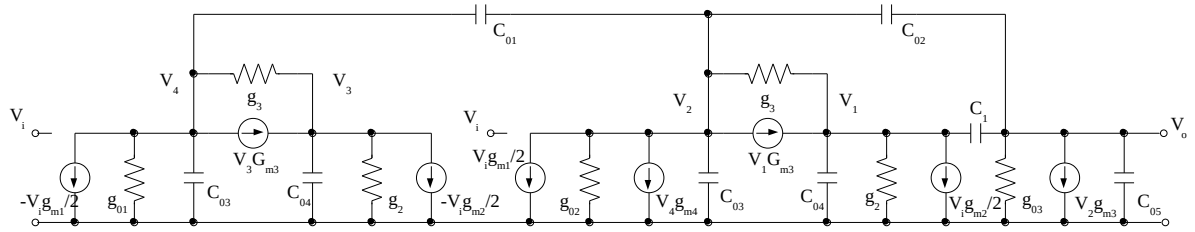


Abbildung 3-13: Kleinsignalersatzschaltbild des Kaskoden-Operationsverstärkers

Die folgende Beschreibung des Ersatzschaltbildes kann anhand der Abbildung 3-13 nachvollzogen werden. Die Stromquellen $\pm V_i g_{m1}/2$ bzw. $\pm V_i g_{m2}/2$ ergeben sich aus den Differenzstufen M_1, M_2 und M_3, M_4. Die Spannungen V_1 und V_2 im Ersatzschaltbild entsprechenden den Spannungen auf beiden Seiten des Kaskodentransistors M_6 aus dem Schaltbild in Abbildung 3-12. Auf die gleiche Weise repräsentieren V_3 und V_4 die Spannungen auf beiden Seiten des Kaskodentransistors M_5. Der Leitwert g_{01} ergibt sich aus der Summation des Ausgangsleitwerts des Transistors M_1 und der Steilheit des Transistors M_7. Die Summe der Ausgangsleitwerte von M_2 und M_8 sind im Leitwert g_{02} enthalten. Die abhängige Quelle V_1G_{m3} beinhaltet die Steilheit des Kaskodentransistors M_6, während g_3 den zugehörigen Drain-Source Leitwert modelliert. Der Strom der gesteuerten Quelle wird mit Hilfe der Spannung V_1 kontrolliert. Analog dazu repräsentiert die Stromquelle V_3G_{m3} die Steilheit des Kaskodentransistors M_5. Im Gegensatz zum Transistor M_6 erfolgt hier die Steuerung des Stroms mittels der Spannung V_3. Die effektive Steilheit G_{m3} stellt in beiden Fällen die Summe aus der Gate-Source Steilheit g_m und der Substratsteilheit g_{mb} des jeweiligen Transistors der Kaskodenstruktur dar. Die Ausgangsleitwerte der Transistoren M_2 und M_4 sind im Leitwert g_2 enthalten. Der Treiber der Ausgangsstufe des Verstärkers M_9 wird durch die Stromquelle V_2g_{m3} in Parallelschaltung mit dem Leitwert g_{03}, der die Ausgangsleitwerte der Transistoren M_9 und M_{10} beinhaltet, umgesetzt.

Dem Modell sind Kleinsignal-Kapazitäten hinzugefügt worden, um den Frequenzgang in wirksamer Weise modellieren zu können. Eine Verbindung des Ausgangs mit dem Source-Anschluss des Transistors M_6 erfolgt somit durch die Kompensationskapazität C_1. Die Gate-Drain Kapazität des Transistors M_8 wird mit Hilfe von C_{01} dargestellt. Die Kompensationskapazität C_2 und die Gate-Drain Kapazität des Transistors M_9 sind in C_{02} enthalten. Die Kapazität C_{03} modelliert die Gate-Drain Kapazitäten der Kaskodentransistoren. Die Sperrschichtkapazität sowie die Gate-Source Kapazität der Transistoren der Kaskode werden mittels der Kapazität C_{04} realisiert. Die parasitäre Kapazität des Transistors M_{10} ist durch C_{05} gegeben.

Nach Auswertung des Kleinsignalersatzschaltbildes ergibt sich für die Leerlaufverstärkung des Operationsverstärkers:

$$\frac{V_o}{V_i}(0) = \frac{g_{m3}\{g_{m1}(g_2+g_3-G_{m3})+g_{m2}(g_3-G_{m3})\}}{2g_{03}\frac{(g_2+g_3-G_{m3})}{g_{m4}+g_{01}+g_3+\frac{g_3(G_{m3}-g_3)}{(g_2+g_3-G_{m3})}}\left(g_{01}+g_3+\frac{g_3(G_{m3}-g_3)}{(g_2+g_3-G_{m3})}\right)\left(g_{02}+g_3+\frac{g_3(G_{m3}-g_3)}{(g_2+g_3-G_{m3})}\right)}. \tag{3-2}$$

Weiterhin liefert die Analyse des Kleinsignalmodells nach Anwendung geeigneter Verfahren eine Übertragungsfunktion der folgenden Form:

$$\frac{V_o}{V_i}(s) = \frac{s^4a_{01}+s^3a_{02}+s^2a_{03}+sa_{04}+a_{05}}{s^5a_{11}+s^4a_{12}+s^3a_{13}+s^2a_{14}+sa_{15}+a_{16}}, \tag{3-3}$$

wobei gilt:

$$a_{01} = -g_{m2}C_{04}\delta_{25}C_1\delta_{22} + \frac{1}{2}g_{m1}C_{02}\delta_{24}C_{04}\delta_{25}, \tag{3-4}$$

$$a_{02} = -g_{m4}\left(g_{m2}C_1C_{01} + \frac{1}{2}g_{m1}C_{02}C_{04}\right)\delta_{04} - \frac{1}{2}(g_{m1}g_{m3}\delta_{24} - g_{m1}C_{02}\delta_{11})C_{04}\delta_{25} + (\delta_{11}\delta_{25} + C_{04}\delta_{13})\left(\frac{1}{2}g_{m1}C_{02}\delta_{24} - g_{m2}C_1\delta_{22}\right) - C_{04}\delta_{25}(g_{m2}C_{02}\delta_{12} + g_{m2}C_1\delta_{14}), \tag{3-5}$$

$$a_{03} = -\frac{1}{2}g_{m1}g_{m3}C_{04}\delta_{11}\delta_{25} + (\delta_{11}\delta_{25} + C_{04}\delta_{13})\left(-\frac{1}{2}(g_{m1}g_{m3}\delta_{24} - g_{m1}C_{02}\delta_{11}) - (g_{m2}C_{02}\delta_{12} + g_{m2}C_1\delta_{14})\right) + (\delta_{11}\delta_{13} - g_3\delta_{12})\left(\frac{1}{2}g_{m1}C_{02}\delta_{24} - g_{m2}C_1\delta_{22}\right) + g_{m4}\left(\frac{1}{2}g_{m1}g_{m3}C_{04} + C_{02}\left(g_{m2}\delta_{12} - \frac{1}{2}g_{m1}\delta_{11}\right)\right)\delta_{24} - g_{m4}\left(g_{m2}C_1C_{02} + \frac{1}{2}g_{m1}C_{02}C_{04}\right)\delta_{11} + g_{m2}g_{m3}C_{04}\delta_{12}\delta_{25}, \tag{3-6}$$

$$a_{04} = (\delta_{11}\delta_{25} + C_{04}\delta_{13})\left(-\frac{1}{2}g_{m1}g_{m3}\delta_{11} + g_{m2}g_{m3}\delta_{12}\right) - \frac{1}{2}(\delta_{11}\delta_{13} - g_3\delta_{12})\big((g_{m1}g_{m3}\delta_{24} - g_{m1}C_{02}\delta_{11}) - (g_{m2}C_{02}\delta_{12} + g_{m2}C_1\delta_{14})\big) - g_{m3}g_{m4}\left(g_{m2}\delta_{12} - \frac{1}{2}g_{m1}\delta_{11}\right)\delta_{24} + g_{m4}\left(\frac{1}{2}g_{m1}g_{m3}C_{04} + C_{02}\left(g_{m2}\delta_{12} - \frac{1}{2}g_{m1}\delta_{11}\right)\right)\delta_{11}\,, \tag{3-7}$$

$$a_{05} = g_{m3}g_4\left(g_{m2}\delta_{12} - \frac{1}{2}g_{m1}\delta_{11}\right)\delta_{11} + (\delta_{11}\delta_{13} - g_3\delta_{12})\left(-\frac{1}{2}g_{m1}g_{m3}\delta_{11} + g_{m2}g_{m3}\delta_{12}\right), \tag{3-8}$$

$$a_{11} = C_{04}\delta_{25}(\delta_{22}(C_1^2 - \delta_{21}\delta_{23}) + C_{02}^2\delta_{24}), \tag{3-9}$$

$$a_{12} = -(g_{m3}C_{02}\delta_{24} - C_{02}^2\delta_{11})C_{04}\delta_{25} + (\delta_{11}\delta_{25} + C_{04}\delta_{13})\left(C_{02}^2\delta_{24} + \delta_{22}(C_1^2 - \delta_{21}\delta_{23})\right) + g_{m4}C_{01}\delta_{24}(C_1^2 - \delta_{21}\delta_{23}) + C_{04}\delta_{25}(\delta_{14}(C_1^2 - \delta_{21}\delta_{23}) - \delta_{22}(g_{03}\delta_{21} + \delta_{11}\delta_{13}) + C_1C_{02}\delta_{12}) \tag{3-10}$$

$$a_{13} = (\delta_{11}\delta_{13} - g_3\delta_{12})(\delta_{22}(C_1^2 - \delta_{21}\delta_{23}) + C_{02}^2\delta_{24}) - g_3C_{02}C_{04}\delta_{11}\delta_{25} + (\delta_{11}\delta_{25} + C_{04}\delta_{13})\left((\delta_{14}(C_1^2 - \delta_{21}\delta_{23}) - \delta_{22}(g_{03}\delta_{21} + \delta_{11}\delta_{13}) - C_1C_{02}\delta_{12}) - (g_{m3}C_{02}\delta_{24} - C_{02}^2\delta_{11})\right) + C_{04}\delta_{25}\big(\delta_{14}(-g_{03}\delta_{21} - \delta_{11}\delta_{13}) - g_{03}\delta_{11}\delta_{22} - \delta_{12}(g_{m3}C_1 - g_3\delta_{23})\big) + g_{m4}\delta_{11}(C_1^2C_{01} - C_{01}\delta_{21}\delta_{23}) - g_{m4}\delta_{24}(g_{03}C_{01}\delta_{21} + C_{01}\delta_{11}\delta_{23}), \tag{3-11}$$

$$a_{14} = (\delta_{11}\delta_{25} + C_{04}\delta_{13})\left(-g_{m3}C_{02}\delta_{11} + \left(g_{03}\delta_{11}\delta_{14}\big((-\delta_{23} - \delta_{23}) - \delta_{22}\big) - \delta_{12}(g_{m3}C_1 - g_3\delta_{23})\right)\right) + (\delta_{11}\delta_{13} - g_3\delta_{12})\left(-(g_{m3}C_{02}\delta_{24} - C_{02}^2\delta_{11}) + (\delta_{14}(C_1^2 - \delta_{21}\delta_{23}) - \delta_{22}(g_{03}\delta_{21} + \delta_{11}\delta_{23}) + C_1C_{02}\delta_{12})\right) + g_{m4}g_{03}\delta_{11}\big((-C_{01}\delta_{21} - C_{01}\delta_{11}\delta_{23}) - \delta_{24}C_{01}\big) + g_{03}C_{04}\delta_{25}(-\delta_{11}\delta_{14} + g_3\delta_{12}), \tag{3-12}$$

$$a_{15} = \delta_{11}(\delta_{11}\delta_{13} - g_3\delta_{12})\left(\left(g_{03}\delta_{14}\left((-g_{03}\delta_{21} - \delta_{11}\delta_{23}) - \big(\delta_{22} - \delta_{12}(g_{m3}C_1 - g_3\delta_{23})\big)\right) - g_{m3}C_{02}\right)\right) + g_{03}(\delta_{11}\delta_{25} + C_{04}\delta_{13})(-\delta_{11}\delta_{14} + g_3\delta_{12}) - g_{m4}g_{03}C_{01}\delta_{11}^2\,, \tag{3-13}$$

$$a_{16} = (\delta_{11}\delta_{13} - g_3\delta_{12})(-g_{03}\delta_{11}\delta_{14} + g_3g_{03}\delta_{12}), \tag{3-14}$$

$$\begin{aligned}\delta_{11} &= g_2 + g_3 - G_{m3}\\ \delta_{12} &= g_3 - G_{m3}\\ \delta_{13} &= g_{01} + g_3\\ \delta_{14} &= g_{02} + g_3\end{aligned} \quad , \tag{3-15}$$

$$\begin{aligned}\delta_{21} &= C_1 + C_{04}\\ \delta_{22} &= C_{01} + C_{02} + C_{03}\\ \delta_{23} &= C_{01} + C_{02} + C_{05}\\ \delta_{24} &= C_1 + C_{04} - g_3 C_1\\ \delta_{25} &= C_{01} + C_{03}\end{aligned} \quad , \tag{3-16}$$

Infolge der Komplexität von Gl. (3-3) finden numerische Verfahren zur Lösung der Transferfunktion Anwendung. Obgleich der Nenner ein Polynom 5.Grades darstellt, zeigt die numerische Lösung, dass lediglich 3 Polstellen für weitere Betrachtungen von Interesse sind. Der dominante Pol liegt bei einer Frequenz von 93 Hz vor. Die weiteren beiden Polstellen sind bei 7.9 MHz bzw. 13.6 MHz zu finden. Die Lösung des Zählerpolynoms der Übertragungsfunktion liefert die nachfolgenden drei relevanten Nullstellen. So ist die erste Nullstelle bei 180 kHz und die verbleibenden beiden Nullstellen liegen 9 MHz bzw. 11.7 MHz.

3.2.3 Layout des Kaskoden-Operationsverstärkers

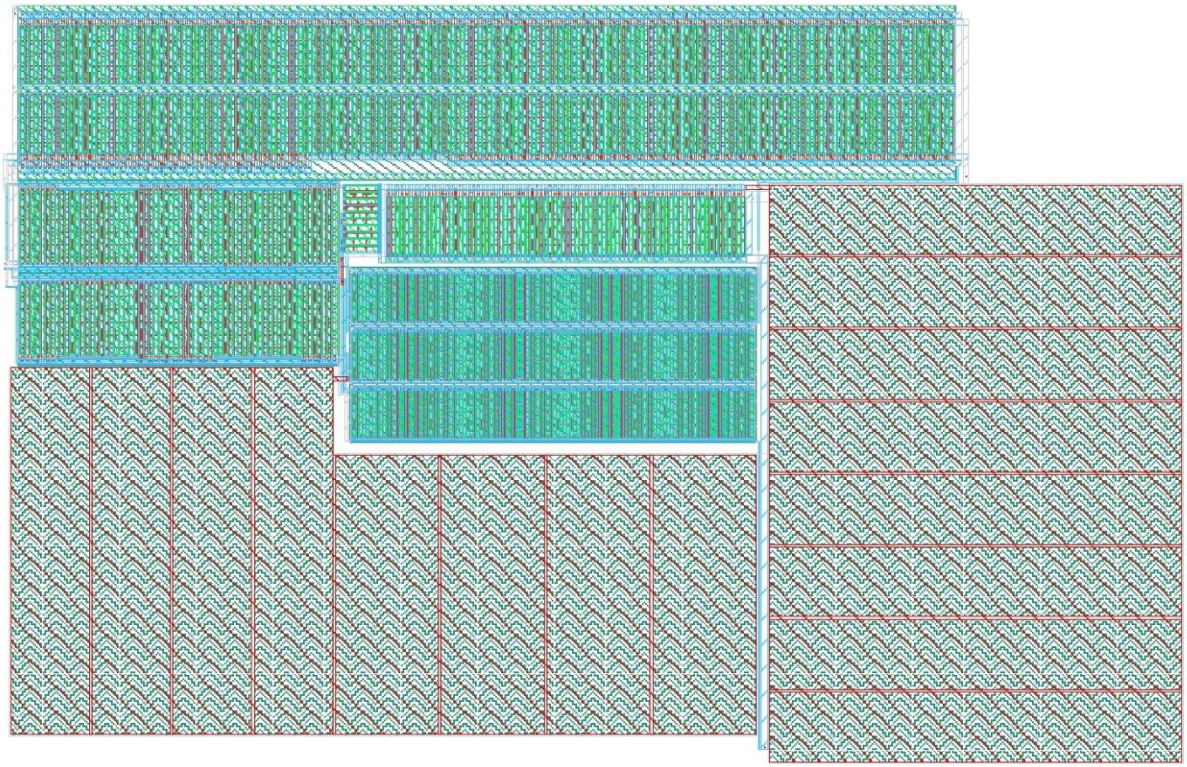


Abbildung 3-14: Layout des Kaskoden-Operationsverstärkers

In Abbildung 3-14 ist das Layout des Operationsverstärkers dargestellt. Für das Design und die Fertigung des Verstärkers ist eine 0,25µm CMOS Technologie der Firma IHP GmbH verwendet worden. Für das Design und die Anfertigung der Layouts ist das Designpaket ALADIN und dessen Modulgeneratorumgebung benutzt worden.

Der durch die Transistoren M_{10} - M_{13} umgesetzte Stromspiegel ist in der oberen Hälfte des Layouts positioniert. Unterhalb dieses Moduls sind linksseits die Eingangsdifferenzstufen angeordnet. In der

Mitte des Layouts sind die aktive Last (M_7, M_8), der n-Kanal Ausgangstransistor und die Kaskode zu finden. Die verbleibende Fläche des Layouts wird durch die beiden Kompensationskapazitäten C_1 und C_2 belegt.

3.2.4 Messergebnisse des Kaskoden-Operationsverstärkers

Die Messungen, deren Ergebnisse in diesem Abschnitt präsentiert werden, sind mit einer Versorgungsspannung von ±1.5 V und einem Einsverstärker durchgeführt worden. Dieser Pufferverstärker weist eine Bandbreite von 250 MHz und eine Eingangskapazität von 2pF auf. Die gemessenen Eckdaten des Operationsverstärkers sind der Tab. 3-4 zu entnehmen.

Die Verstärkungskennlinie und der Phasengang des Operationsverstärkers sind in dem Bode Diagramm in Abbildung 3-15 dargestellt. Das Antwortzeitverhalten für einen rechteckförmigen Eingangsimpuls ist in Abbildung 3-16 gezeigt.

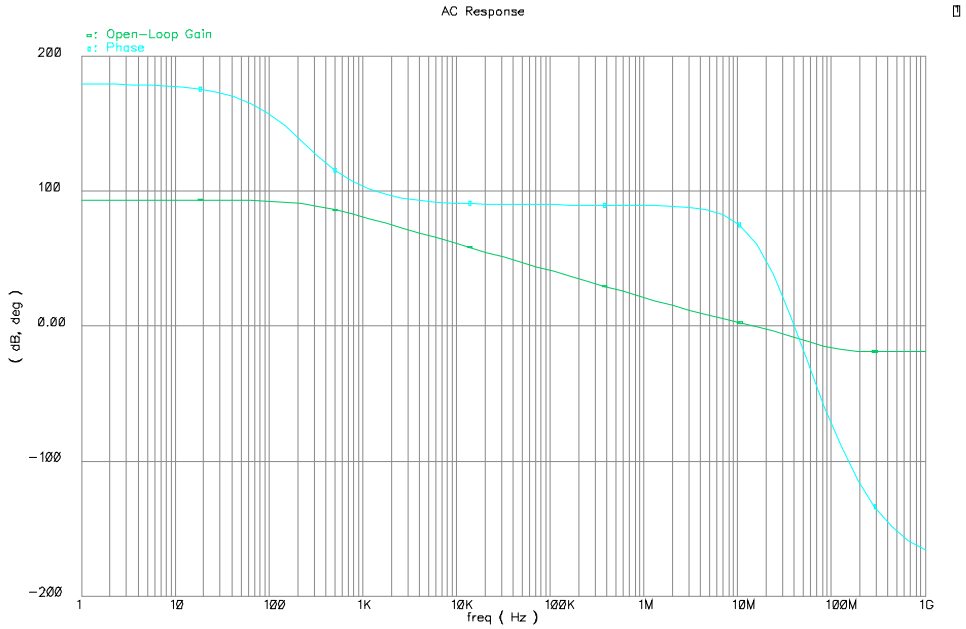


Abbildung 3-15: Bode Diagramm des Operationsverstärkers aus Abbildung 3-12

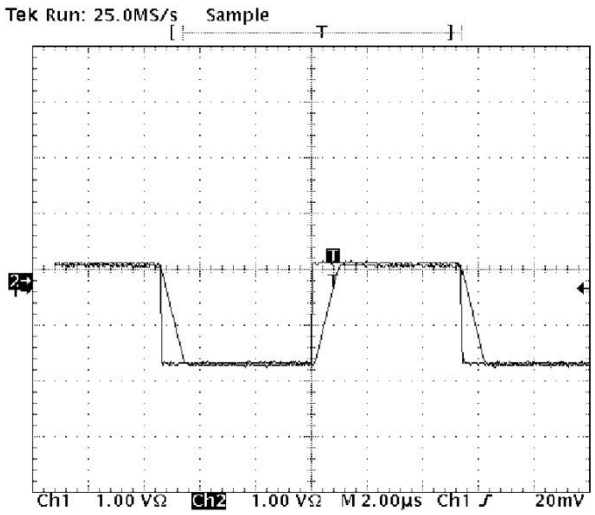


Abbildung 3-16: Zeitverhalten des Kaskodenverstärkers für einen Rechteckimpuls

Nachträgliche Postlayout-Simulationen haben gezeigt, dass die Größe der Kompensationskapazitäten reduziert werden kann. Die Verwendung eines vor die Kapazität C_2 platzierten Passgates, welches die Funktion eines Kompensationswiderstands übernehmen soll, ermöglicht eine Reduktion der Kapazitätswerte von C_1 und C_2 auf 40% der ursprünglichen Größe.

Außerdem ist es möglich die Versorgungsspannung auf ±1 V zu reduzieren. In diesem Fall beträgt die aus der Simulation ermittelte Leerlaufspannungsverstärkung 98 dB und die Phasenreserve beläuft sich auf 61.4°.

Tabelle 3-4: Messergebnisse des Kaskoden-Operationsverstärkers

Parameter	Messergebnisse	Einheiten
Versorgungsspannung	±1.5	V
Leerlaufspannungsverstärkung	97	dB
Bandbreite	14.3	MHz
Phasenreserve	62	°
Eingangs-Offset-Spannung	-2.1	V
Ausgangshub	±1.06	V
Eingangshub	±1.5	V
CMRR	> 67	dB
PSRR+ bei		
1 kHz	80	dB
10 kHz	67	dB
100 kHz	57	dB
PSRR- bei		
1 kHz	76	dB
10 kHz	65	dB
100 kHz	56	dB
Positive Slew Rate	2.6	V/µs
Negative Slew Rate	2.5	V/µs
Fläche (Länge * Weite)	0.17	µm²

3.3 Einstufiger Operationsverstärker

Mit Hilfe einer einzelnen Verstärkerstufe kann in der Regel eine Verstärkung von etwa 40 dB erzielt werden. Um jedoch eine Steigerung auf 80 dB oder mehr zu erreichen, ist eine Kaskadierung von zwei verstärkenden Stufen notwendig. Dabei haben zwei solche Stufen in der Übertragungsfunktion zwei Polstellen zur Folge, was eine zusätzliche Frequenzkompensation erforderlich macht (siehe Kap. 3.1). Das benötigte Kompensationsnetzwerk erhöht die Komplexität der Schaltung und reduziert die Flexibilität des Designs.

Die Verwendung einer Kaskode mit einer entsprechenden Last erlaubt es eine hohe Verstärkung mit einer Verstärkerstufe zu erreichen, ohne den Nachteil von zwei dicht benachbarten Polstellen. Aus diesem Grund stellt die Verwendung eines einstufigen Operationsverstärkers mit einer Kaskode eine interessante Alternative zu einem zweistufigen Operationsverstärker dar.

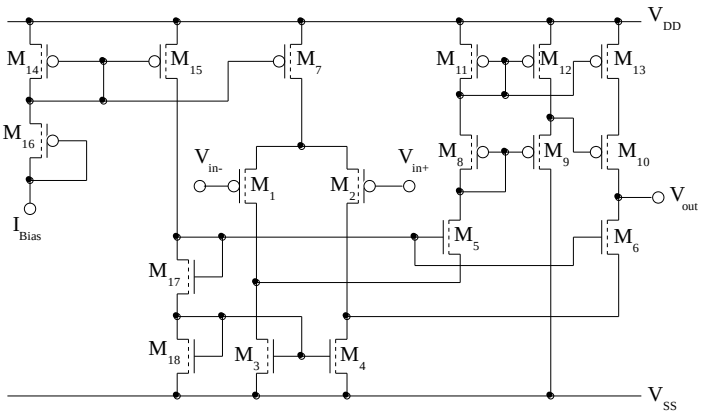


Abbildung 3-17: Schaltbild eines einstufigen Operationsverstärkers mit gefalteter Kaskode

Im Nachfolgenden werden das Design und die Simulationsergebnisse eines einstufigen Operationsverstärkers mit einer gefalteten Kaskode beschrieben. Das Schaltbild des Verstärkers ist der Abbildung 3-17 zu entnehmen. Der Stromquellentransistor M_7 und die beiden Transistoren M_1 und M_2 bilden eine Differenzverstärkerstufe. Die Transistoren M_3 und M_4 dienen lediglich zur Arbeitspunkteinstellung. Der Weg der Verstärkung führt über die Differenzstufe mit den Transistoren M_1 und M_2 auf die Kaskodenstufe in Gate-Schaltung mit den Transistoren M_5 und M_6. Die Kaskodenstufe ihrerseits ist weiterhin verbunden mit einem verbesserten Stromspiegel bestehend aus den Transistoren M_8 - M_{13}. Die High-Swing-Kaskode als Stromspiegel weist einen deutlich größeren Ausgangswiderstand als ein einfacher Stromspiegel auf, während der Aussteuerungsbereich der beiden Stromspiegelvarianten vergleichbar ist [Fran94]. Der HighSwing-Stromspiegel ist besonders für Low-Voltage-Anwendungen geeignet. Der Ausgang des Verstärkers wird an dem hochohmigen Verbindungspunkt zwischen Kaskodenausgang und Stromspiegelausgang herausgeführt. Bei Bedarf kann an dieser Stelle auch eine Frequenzkompensation bzw. eine Endstufe angeschlossen werden.

Ein Vorteil dieser Schaltung ist darin gegeben, dass der Operationsverstärker lediglich einen hochohmigen Knoten aufweist und damit die Frequenzkompensation, sofern diese überhaupt notwendig ist, durch die im Schaltbild angegebene Kapazität C_L vorgenommen werden kann. Weiterhin wird der Bereich der Ausgangsspannung durch die Verwendung der gefalteten Kaskode auf näherungsweise U_B-$3U_{ds,sat}$ ausgedehnt, da die Kaskodenstufe von der Betriebsspannung aus betrachtet parallel zum Eingangsdifferenzverstärker arbeitet. Eine näherungsweise Berechnung der Leerlaufverstärkung ist mit einem geringen Aufwand verbunden, wenn die Verstärkerstufen als rückwirkungsfrei angenommen werden. Infolgedessen ist es möglich, die Verstärkung der Differenzstufe und der Kaskodenstufe zu multiplizieren. Die Aufspaltung und spätere Zusammenführung der Verstärkerzweige können außer Acht gelassen werden. Damit ergibt sich für die Leerlaufspannungsverstärkung:

$$\frac{V_o}{V_i}(0) \approx \frac{g_{m1}}{g_{ds1}+g_{ds3}+g_{m6}} \cdot \frac{g_{m6}}{g_{ds5}+g_{ds9}} \approx \frac{g_{m1}}{g_{ds5}+g_{ds9}}. \tag{3-17}$$

Der effektive Innenwiderstand ist sehr hoch, da beide daran beteiligten Stufen Kaskodenstufen sind. Aus diesem Grund wird die Leerlaufverstärkung voraussichtlich zwischen 500 und 2000 liegen.

3.3.1 Layout des einstufigen Operationsverstärkers

Der einstufige Operationsverstärker mit einer gefalteten Kaskode ist in einer 0,25µm CMOS Technologie der Firma IHP GmbH entworfen worden. Das in Abbildung 3-18 gezeigte Layout umfasst eine Fläche von 161 ×92µm^2.

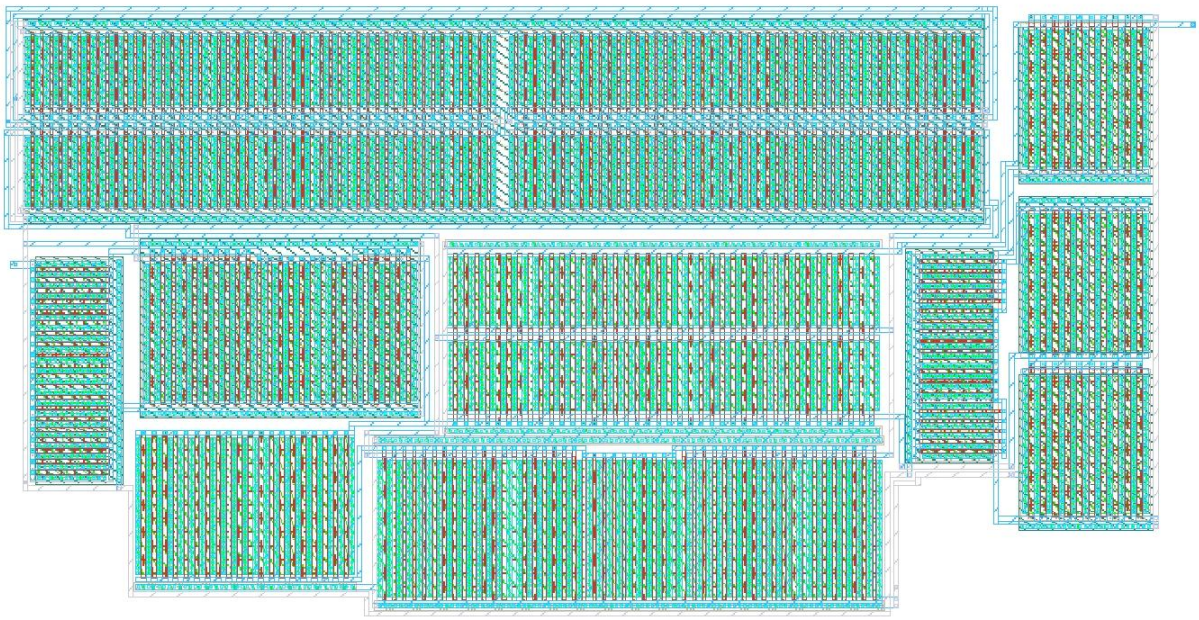


Abbildung 3-18: Layout des einstufigen Operationsverstärkers

3.3.2 Simulationsergebnisse des einstufigen Operationsverstärkers

Die Simulationsergebnisse des einstufigen Operationsverstärkers mit Berücksichtigung der parasitären Elemente aus dem Layout sind in Tab. 3-5 zusammengefasst.

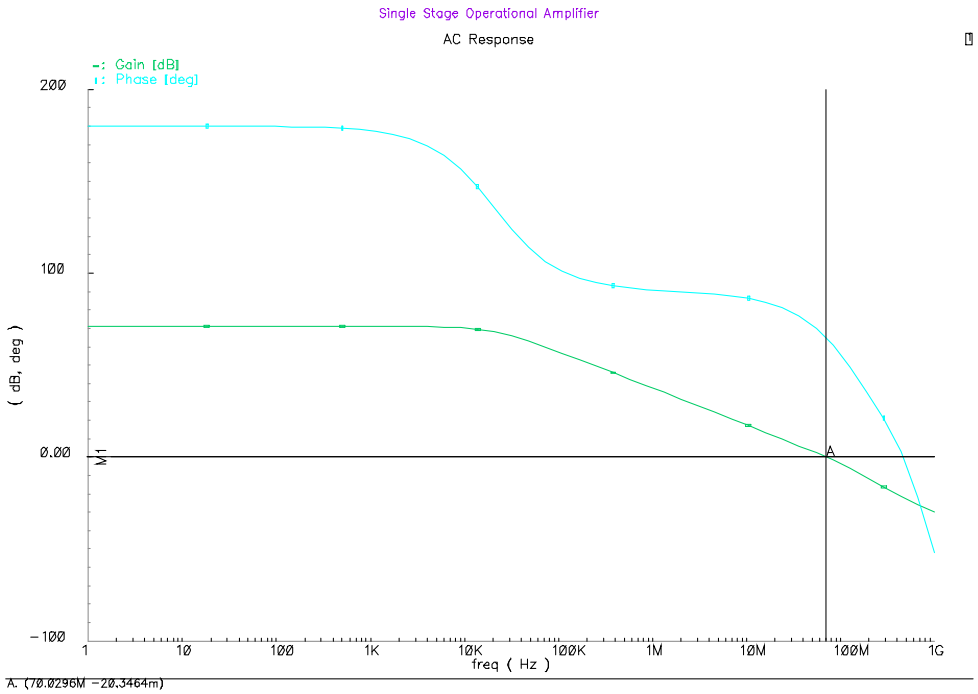


Abbildung 3-19: Bode Diagramm des einstufigen Operationsverstärkers

Tabelle 3-5: Simulationsergebnisse des einstufigen Operationsverstärkers

Parameter	Simulationsergebnis	Einheiten
Leerlaufspannungsverstärkung	71	dB
Bandbreite	70	MHz
Phasenreserve	65	°
Positive Slew Rate	41	V/µs
Negative Slew Rate	47	V/µs

Der Verstärker wurde für eine Versorgungsspannung von 3V und eine kapazitive Last von 9 pF dimensioniert. Die DC-Verstärkung beläuft sich auf 71 dB und die Bandbreite beträgt 70 MHz. Die Verstärkungs-kennlinie und der Phasengang sind im Bode Diagramm in Abbildung 3-19 dargestellt. Aus der simulierten Übertragungsfunktion lässt sich eine Phasenreserve von 65° ablesen.

3.4 Operationsverstärker mit gefalteter Kaskode

Es ist ein Redesign für einen Operationsverstärker, der mit einer niedrigen Versorgungsspannung nach dem Prinzip der gefalteten Kaskode [Roew02] betrieben wird, in einer 0.25µm CMOS Technologie durchgeführt worden. Für den ursprünglichen Verstärker wurde eine 0.8µm CMOS Technologie verwendet. Der Verstärker erzielte eine Leerlaufverstärkung von 72.4 dB und eine Bandbreite von 13.5 MHz.

Nachstehend werden der Aufbau und das Kleinsignalmodell des Verstärkers beschrieben. Anschließend werden das Layout und die Messergebnisse dargestellt.

3.4.1 Design des Operationsverstärkers

Das Schaltbild des Operationsverstärkers mit einer gefalteten Kaskode ist in Abbildung 3-20 dargestellt. Der Verstärker setzt sich aus zwei komplementären, differentiellen Eingangsstufen, bestehend aus den Transistoren M_1, M_2 und M_3, M_4, sowie den beiden komplementären, aktiven Lasten, die durch die Stromspiegel M_5, M_6 und M_9, M_{10} realisiert werden, zusammen. Diese Stromspiegel sind mit den Kaskodentransistoren M_7 und M_8 verbunden. Wenn die differentielle Eingangsspannung Null ist und das Verhältnis aller verwendeten Stromspiegel dem Wert 1:1 entspricht, dann beträgt der Strom durch die differentiellen Eingangsstufen $1/2 \cdot I_{bias}$. Weiterhin fließt in diesem Fall kein Strom durch die Kaskode und der Verstärker befindet sich somit im dynamischen Arbeitsbereich. Um diesen Effekt zu vermeiden, muss ein zusätzlicher vordefinierter Strom, der unter Verwendung von zwei MOS Dioden erzeugt wird, bereitgestellt werden [Vall94]. Unter dieser Voraussetzung sind der Strom sowie der gespiegelte Strom im Ausgangszweig abhängig von der Versorgungsspannung, wodurch jedoch die Wahl der minimalen Betriebsspannung und der maximale Ausgangshub eingeschränkt werden. Erfolgt die Bereitstellung des zusätzlichen Stroms mit Hilfe von asymmetrischen Stromspiegeln als aktive Lasten, so ist möglich, den genannten Nachteilen entgegen zu wirken [Roew02]. Um einen maximalen Ausgangsspannungshub zu erzielen, werden die Kaskoden unabhängig eingestellt.

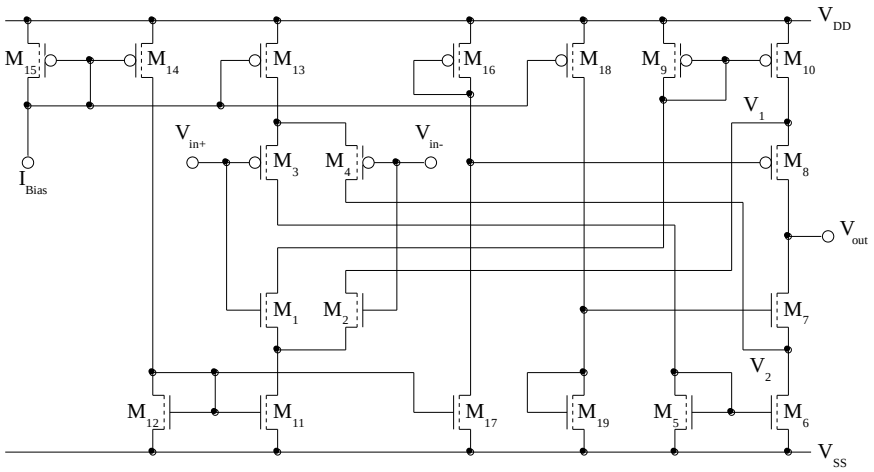


Abbildung 3-20: Schaltbild des Operationsverstärkers mit gefalteter Kaskode

3.4.2 Kleinsignalmodell des Operationsverstärkers mit gefalteter Kaskode

In diesem Abschnitt wird eine kurze Beschreibung eines adäquaten Kleinsignalmodells (siehe Abbildung 3-21) für den Operationsverstärker mit der gefalteten Kaskode gegeben.

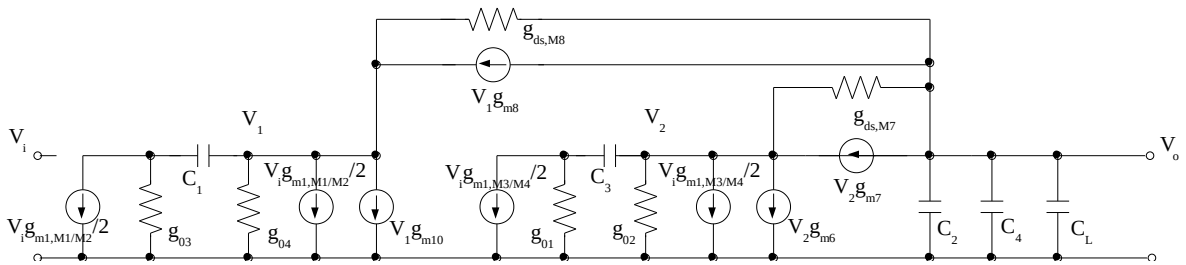


Abbildung 3-21: Kleinsignalersatzschaltbild des Operationsverstärkers aus Abbildung 3-20

Um ein leicht zu handhabendes Kleinsignalersatzschaltbild zu erhalten, sind einige Elemente zusammengefasst und diverse kleinere Elemente ausgelassen worden. Die Stromquellen $V_i g_{m,M1/M2}$ und $V_i g_{m,M3/M4}$ ergeben sich aus den Eingangsdifferenzstufen M_1, M_2 bzw. M_3, M_4. Der Leitwert g_{01} repräsentiert die Summe aus dem Ausgangsleitwert des Transistors M_1 und der Steilheit des Transistors M_5. Die Summe der Ausgangsleitwerte von M_4 und M_6 sind im Leitwert g_{02} enthalten. Die abhängige Quelle $V_1 g_{m8}$ beinhaltet die Steilheit des Kaskodentransistors M_8, während $g_{ds,M8}$ den zugehörigen Ausgangsleitwert modelliert. Die effektive Steilheit g_{m8} ist hierbei die Summe aus der Gate-Source Steilheit g_m und der Substratsteilheit g_{mb}. Der Strom der abhängigen Quelle wird durch die Spannung V_1 kontrolliert. Auf die gleiche Weise stellt die Stromquelle $V_2 g_{m7}$, welche mit dem Ausgangsleitwert $g_{ds,M7}$ parallel geschaltet ist, die Steilheit der Kaskode M_7 dar. In diesem Fall wird der Strom mit Hilfe der Spannung V_2 gesteuert. Die Ausgangsleitwerte der Transistoren M_2 und M_{10} bilden den Leitwert g04. Die Summe aus der Steilheit von M_9 und dem Ausgangsleitwert von M_1 sind in g_{03} enthalten. Die abhängigen Quellen $V_1 g_{m,M10}$ und $V_2 g_{m,M6}$ repräsentieren die Steilheit der Transistoren M_{10} bzw. M_6.

Um eine effektive Frequenzantwort zu modellieren, werden einige Kleinsignalkapazitäten ebenfalls mit berücksichtigt. So wird die kapazitive Last am Ausgang mittels der Kapazität C_L erfasst. C_2 und C_4 beschreiben die Gate-Drain Kapazitäten der Transistoren M_8 und M_7. Infolge dessen, das in der Regel die Weite eines PMOS Transistors größer als die Weite eines NMOS Transistors gewählt wird, sind die

Kapazitäten C_2 und C_4 verschieden. Die verbleibenden Kapazitäten C_1 und C_3 resultieren schließlich aus den Transistoren M_6 und M_{10}.

Eine Analyse des gezeigten Kleinsignalmodells liefert, nach Anwendung einer geeigneten Algebra, die folgende Übertragungsfunktion:

$$\frac{V_o}{V_i}(s) = \frac{1}{2}\left(1+\frac{g_{m,M6}}{g_{01}}\right)\cdot\frac{\left(\frac{sC_2}{g_{m8}}+1\right)g_{m,M1M2}+\left(\frac{sC_4}{g_{m7}}+1\right)g_{m,M3M4}}{\frac{g_{02}g_{ds,M7}}{g_{m7}}+\frac{g_{04}g_{ds,M8}}{g_{m8}}+\left(\frac{C_2C_4}{g_{m7}g_{m8}}s^2+\frac{C_2g_{m7}+C_4g_{m8}}{g_{m7}g_{m8}}s+1\right)C_Ls}. \tag{3-18}$$

Der Ausgangswiderstand berechnet sich näherungsweise wie folgt:

$$r_{out} = \frac{1}{\frac{g_{02}g_{ds,M7}}{g_{m7}}+\frac{g_{04}g_{ds,M8}}{g_{m8}}}. \tag{3-19}$$

Falls ein sehr großer Ausgangsstrom vorliegt, erzielen die Steilheiten g_{m7} und g_{m8} ebenfalls sehr große Werte, so dass die nachfolgenden Polstellen in etwa dem Wert Eins entsprechen:

$$\begin{aligned} P_1(s) &= \frac{C_2C_4}{g_{m7}g_{m8}}s^2+\frac{C_2g_{m7}+C_4g_{m8}}{g_{m7}g_{m8}}s+1 \\ P_2(s) &= \frac{C_2}{g_{m8}}+1 \\ P_3(s) &= \frac{C_4}{g_{m7}}s+1 \end{aligned} \quad . \tag{3-20}$$

In diesem Fall ergibt sich für die Frequenz des dominanten Pols der Übertragungsfunktion

$$f = \frac{1}{2\cdot\pi\cdot r_{out}\cdot C_L}, \tag{3-21}$$

und die Transitfrequenz beträgt annähernd

$$GBW = \frac{V_o}{V_i}(0)\cdot f \approx \frac{g_{m,M1M2}+g_{m,M3M4}}{2\cdot\pi\cdot C_L}. \tag{3-22}$$

3.4.3 Layout des Operationsverstärkers mit gefalteter Kaskode

In Abbildung 3-22 ist das Layout des Operationsverstärkers gezeigt. Unter der Verwendung der Modulgeneratorumgebung von ALADIN sind Modulgeneratoren geschrieben worden, um das Layout des dargestellten Verstärkers mit gefalteter Kaskode automatisch zu erzeugen. Wie der Abbildung zu entnehmen ist, ergibt sich ein kompaktes Layout.

Die p-Kanal-Differenzstufe (M_3, M_4) ist im oberen linken Bereich des Layouts zu finden. Unterhalb von diesem Modul befindet sich die komplementäre n-Kanal-Differenzstufe (M_1, M_2). Um den Offset zu reduzieren und das Matching-Verhalten zu verbessern, ist für die Eingangstransistoren eine mäanderförmige Struktur verwendet worden. Die Kaskodentransistoren M_7 und M_8 sind der mittleren, rechten Hälfte positioniert. Die aktiven Lasten M_5, M_6 und M_9, M_{10} sowie die MOS Dioden M_{16} und M_{19} sind in der Mitte des Layouts platziert. Der durch die Transistoren M_{13}, M_{14}, M_{15} und M_{18} realisierte Stromspiegel ist im oberen rechten Bereich angeordnet, während der Stromspiegel, der durch die Transistoren M_{11}, M_{12} und M_{17} gebildet wird, den unteren, rechten Bereich des Layout einnimmt.

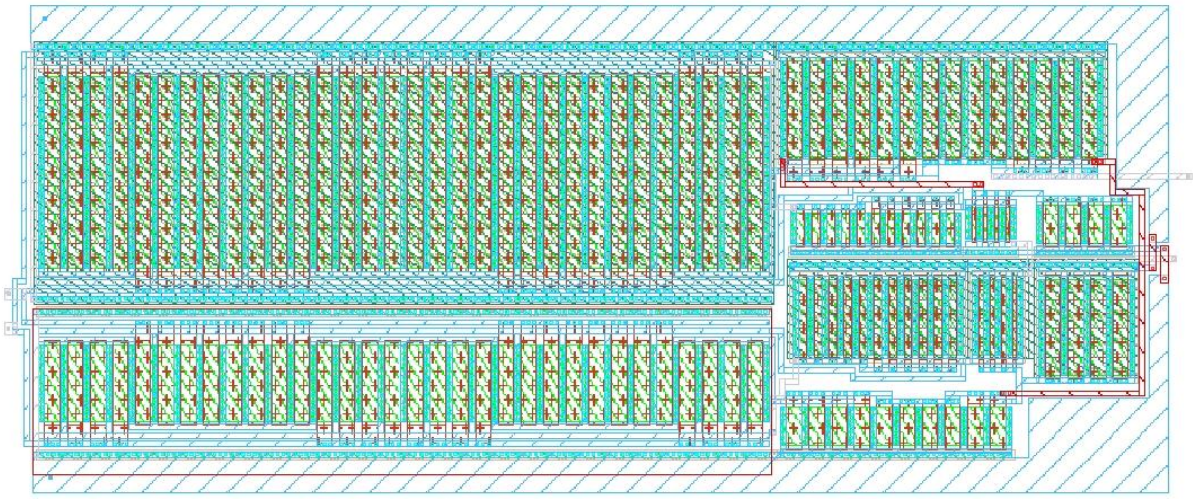


Abbildung 3-22: Layout des Operationsverstärkers aus Abbildung 3-20

3.4.4 Messergebnisse des Operationsverstärkers mit gefalteter Kaskode

Alle in diesem Abschnitt präsentierten Messungen sind bei einer Betriebsspannung von ±1.5 V und unter Verwendung eines Eins-Verstärkers, der eine Bandbreite von 250 MHz und einer Eingangskapazität von 2 pF aufweist, durchgeführt worden. Die Ergebnisse sind der folgenden Tabelle 3-7 zu entnehmen.

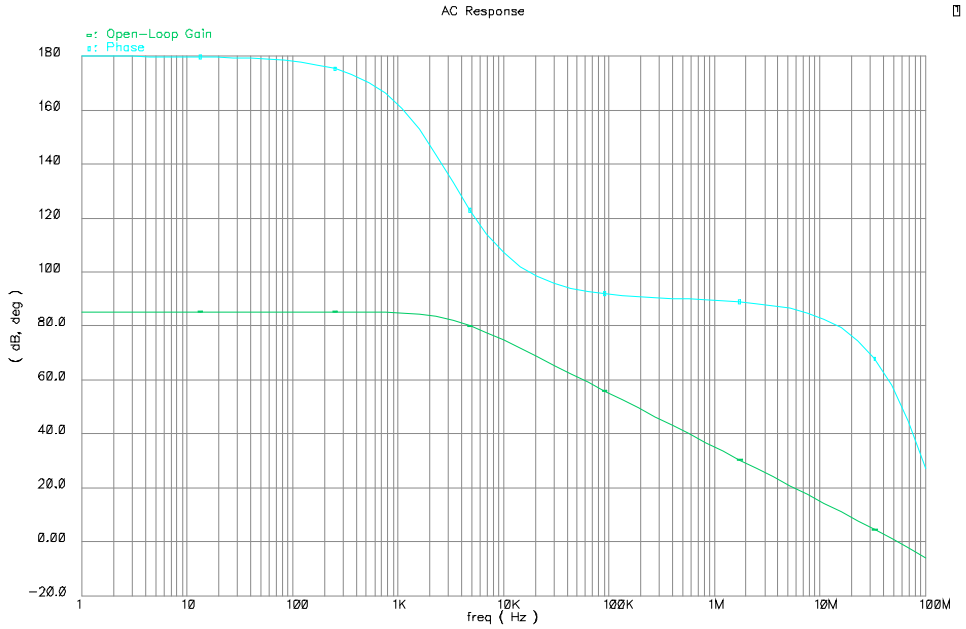


Abbildung 3-23: Bode Diagramm des Operationsverstärkers aus Abbildung 3-20

Tabelle 3-6: Messergebnisse des Operationsverstärkers mit gefalteter Kaskode

Parameter	Messergebnisse	Einheiten
Versorgungsspannung	±1.5	V
Leerlaufspannungsverstärkung	85	dB
Bandbreite	53	MHz
Phasenreserve	55	°
Eingangs-Offset-Spannung	-1.4	mV
Ausgangshub	±1.35	V

Eingangshub	±1.5	V
CMRR	> 60	dB
PSRR+ bei		
1 kHz	80	dB
10 kHz	70	dB
100 kHz	55	dB
PSRR- bei		
1 kHz	79	dB
10 kHz	69	dB
100 kHz	50	dB
Positive Slew Rate	5.3	V/µs
Negative Slew Rate	6.4	V/µs
I_{Bias}	400	µA
Fläche (Länge * Weite)	150 * 60	µm²

Die Übertragungsfunktion und der Phasengang sind der Abbildung 3-23 zu entnehmen. Das Zeitverhalten für einen Rechteckeingangsimpuls ist in Abbildung 3-24 dargestellt.

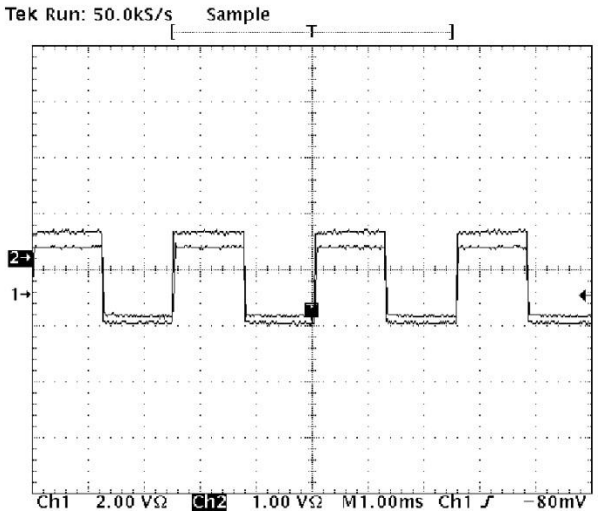


Abbildung 3-24: Zeitverhalten des Operationsverstärkers aus Abbildung 3-20 für einen Rechteckimpuls

3.5 Analoger Leistungsoperationsverstärker

Im Nachfolgenden werden das Design und die Simulationsergebnisse eines analogen Leistungsoperationsverstärkers präsentiert. Der Verstärker ist in der Lage sowohl kapazitive als auch ohmsche Lasten anzusteuern. Die Implementierung basiert auf den in [Fish85] vorgestellten CMOS Leistungsverstärker. Das Schaltbild des analogen Leistungsverstärkers ist der Abbildung 3-25 zu entnehmen. Innerhalb der Struktur werden zwei zweistufige Operationsverstärker A_1 und A_2 und ein einstufiger Vorverstärker A_3 (siehe Kap. 3.3) verwendet. Die Operationsverstärker A_1 und A_2 verwenden zur Frequenzkompensation das Prinzip des Zero-Nulling-Widerstands (siehe Kap. 3.1). In Abbildung 3-26 bzw. Abbildung 3-27 ist der Aufbau der Verstärker A_1 und A_2 dargestellt. Der Ausgangsstrom des Leistungsverstärkers wird über die Transistoren M_1 und M_2 geregelt, während die Kontrolle des Ruhestroms mit Hilfe der Transistoren M_3-M_6 erfolgt.

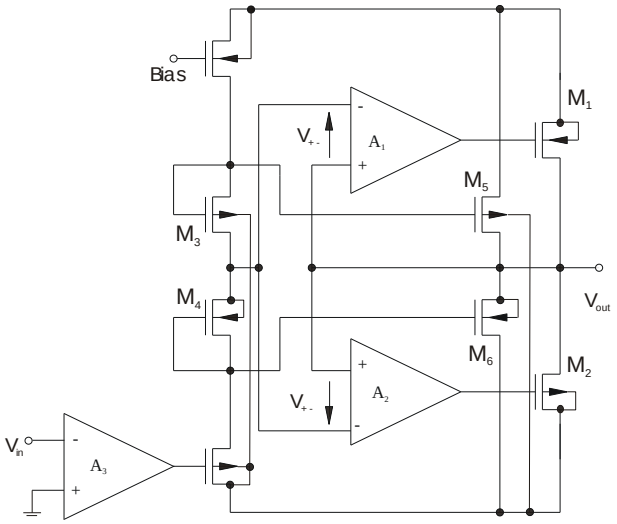


Abbildung 3-25: Schaltbild des linearen Leistungsverstärkers

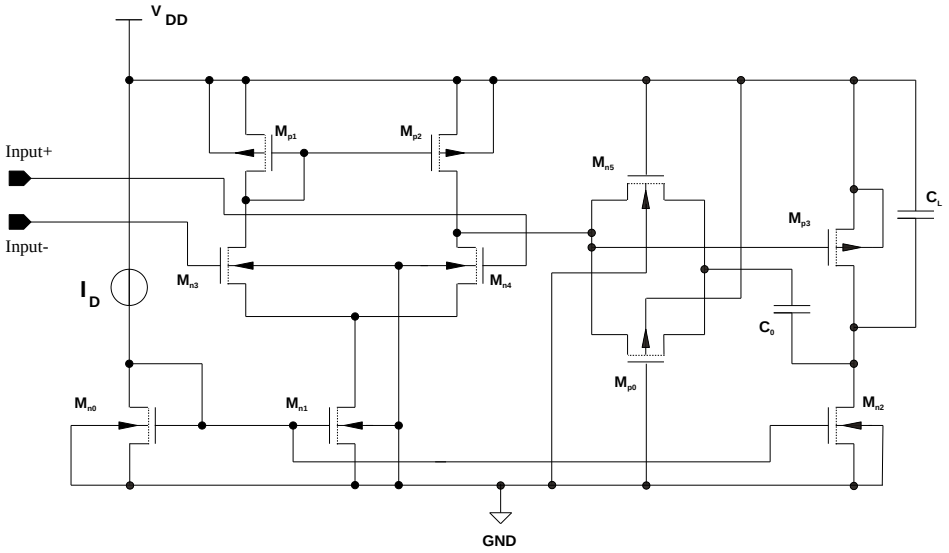


Abbildung 3-26: Schaltbild des Operationsverstärkers A_1

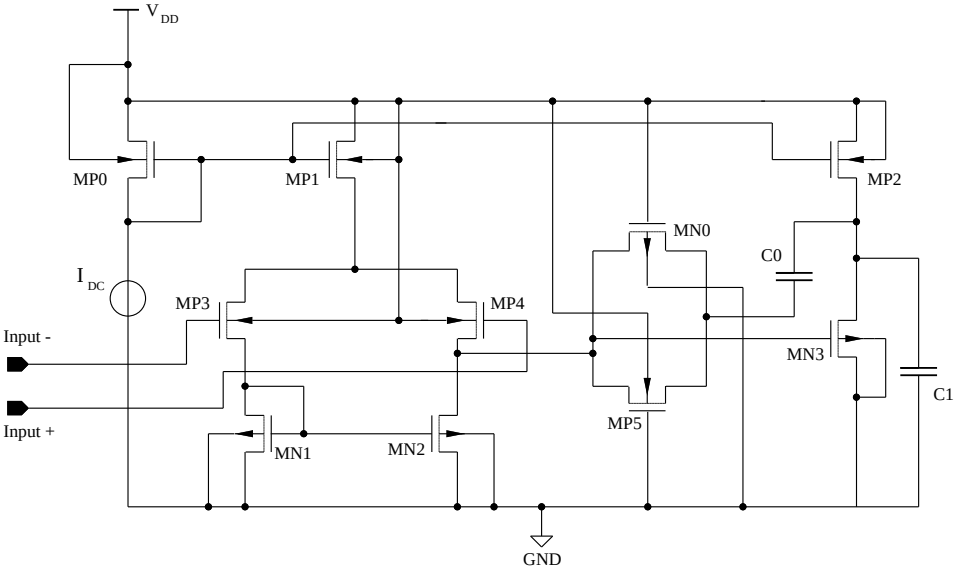


Abbildung 3-27: Schaltbild des Operationsverstärkers A_2

Die Transistoren M_1 und M_2 werden in den Ruhezustand versetzt, wenn die Verstärker A_1 und A_2 eine geringe Offset-Spannung aufweisen. Aus diesem Grund erfolgt die Regelung des Ruhestroms durch die Transistoren M_3-M_6. Der Ausgangsruhestrom verhält sich proportional zum Strom durch die Transistoren M_3 und M_4. Dieser Strom lässt sich mittels einer Funktion des Größenverhältnisses von M_5 zu M_3 und M_6 zu M_4 beschreiben. Beispielsweise fallen bei Volllast in negativer Richtung etwa 95% des erforderlichen Laststroms über M_2 ab und die verbleibenden 5% über M_6. Eine Kontrolle des Stroms durch M_6 findet in diesem Fall genau dann statt, wenn der Verstärker A_2 die Source-Spannungen von M_4 und M_6 angleicht, dabei eine vergleichbare Gate-Source-Spannung über beide Transistoren erzeugt und damit die Ströme entsprechend aufteilt. Auf analoge Weise erfolgt die Kontrolle der Ströme für den positiven Fall.

Ein limitierender Faktor für den Ausgangssteuerbereich der vorliegenden Schaltung liegt in der Schwellenspannung der Transistoren M_4 und M_6, der auf Back-Bias Effekte zurückzuführen ist [Fish85].

Obwohl die Transistoren M_5 und M_6 einen gewissen Anteil am Laststrom haben, liegt der tatsächliche Nutzen in der Kontrolle des Ruhestroms. Gleichfalls reduziert sich durch diese Transistoren die Phasenverschiebung, die sich infolge des Feed-forward Pfades von den Verstärkers A_1 und A_2 zum Ausgang bei hohen Frequenzen einstellt. So benötigen die Operationsverstärker A_1 und A_2 zwar eine minimale Kompensation der Phase, um eine stabile Funktion in einem geschlossenen Regelkreis zu gewährleisten, jedoch tendiert die Gesamtschaltung eher dazu, die Frequenzcharakteristik von M_5 und M_6 anzunehmen anstelle der zusammengesetzten Source-Folger. Der Verstärker ermöglicht damit insgesamt auch einen stabilen Betrieb bei Verwendung einer rein kapazitiven Last.

3.5.1 Layout des analogen Leistungsverstärkers

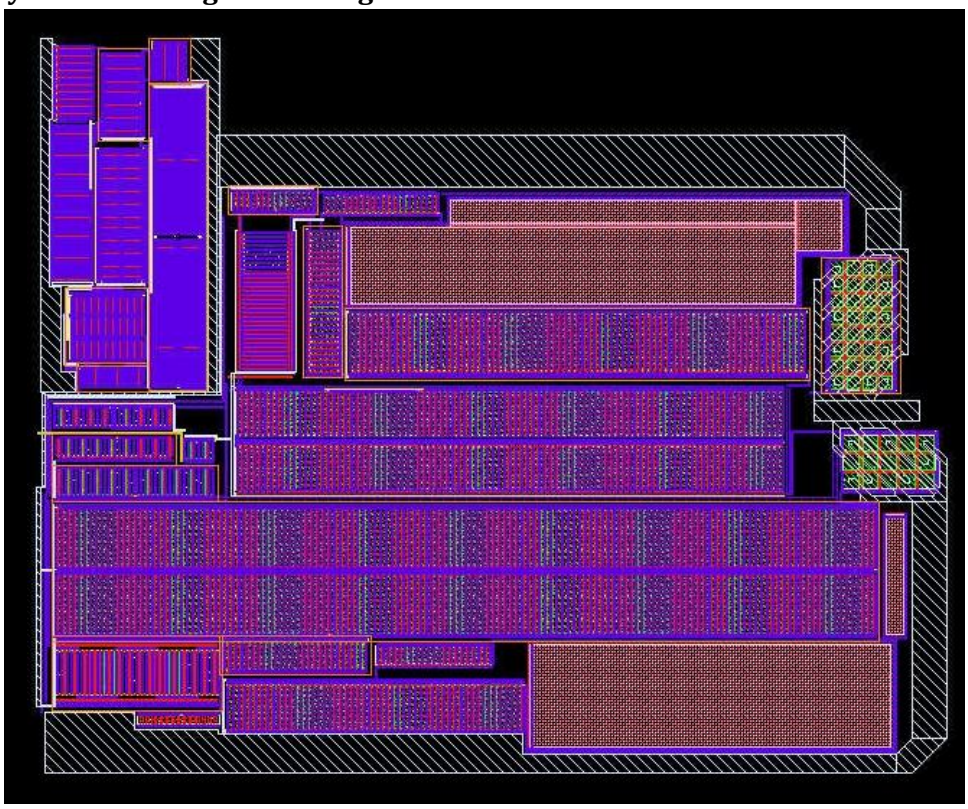


Abbildung 3-28: Layout des analogen Leistungsverstärkers

Unter Verwendung des Designpakets ALADIN sind Modulgeneratoren zur Erstellung des in Abbildung 3-28 gezeigten Layouts des Leistungsverstärkers geschrieben worden. Die Layoutfläche des Verstärkers beträgt 530µm × 430µm^2. Für die Erstellung des Layouts ist eine 0.35µm CMOS Technologie der Firma austriamicrosystems verwendet worden.

Der Vorverstärker A_3 ist in der oberen linken Hälfte platziert worden. Rechtsseits davon befindet sich der Operationsverstärkers A_1. Unterhalb der beiden Verstärker ist der Operationsverstärker A_2 arrangiert. Die Platzierung der Stromspiegel, bestehend aus den Transistoren M_3 und M_5 bzw. M_4 und M_6, erfolgt zwischen den Verstärkern A_2 und A_3. Die Leistungstransistoren M_1 und M_2, für die jeweils die Struktur eines Waffeltransistors gewählt wurde, sind an rechten Rand des Layouts angeordnet worden. Zur Reduzierung des Effekts der Elektromigration sind für einige Leitungen, die eine besonders hohe Stromdichte aufweisen, 45° Strukturen gewählt worden.

3.5.2 Simulationsergebnisse des analogen Leistungsverstärkers

Aus den Tabellen können die Ergebnisse der Postlayout-Simulationen der Operationsverstärker A_1-A_3 entnommen werden. Ferner geht aus den Tabellen hervor, dass ein Betrieb der Verstärker in einem Bereich der Versorgungsspannung von 2.7 V bis 3.3 V gewährleistet ist.

Tabelle 3-7: Simulationsergebnisse für Operationsverstärker A_1 unter Verwendung einer 14 pF Last

VDD [V]	**2.7**	**3**	**3.3**
Leerlaufspannungsverstärkung [dB]	83	84	86
Bandbreite [MHz]	64	66	64
Phasenreserve [°]	55	53	52
Slew Rate [V/µs]	84	90	97
Fläche [µm²]	337 * 181		

Tabelle 3-8: Simulationsergebnisse für Operationsverstärker A_2 unter Verwendung einer 7 pF Last

VDD [V]	**2.7**	**3**	**3.3**
Leerlaufspannungsverstärkung [dB]	83	84	85
Bandbreite [MHz]	58	60	59
Phasenreserve [°]	74	71	71
Slew Rate [V/µs]	76	74	71
Fläche [µm²]	492 * 146		

Tabelle 3-9: Simulationsergebnisse für Operationsverstärker A_3 unter Verwendung einer 7 pF Last

VDD [V]	**2.7**	**3**	**3.3**
Leerlaufspannungsverstärkung [dB]	67	68	69
Bandbreite [MHz]	98	100	101
Phasenreserve [°]	71	71	72
Slew Rate [V/µs]	69	78	78
Fläche [µm²]	210 * 94		

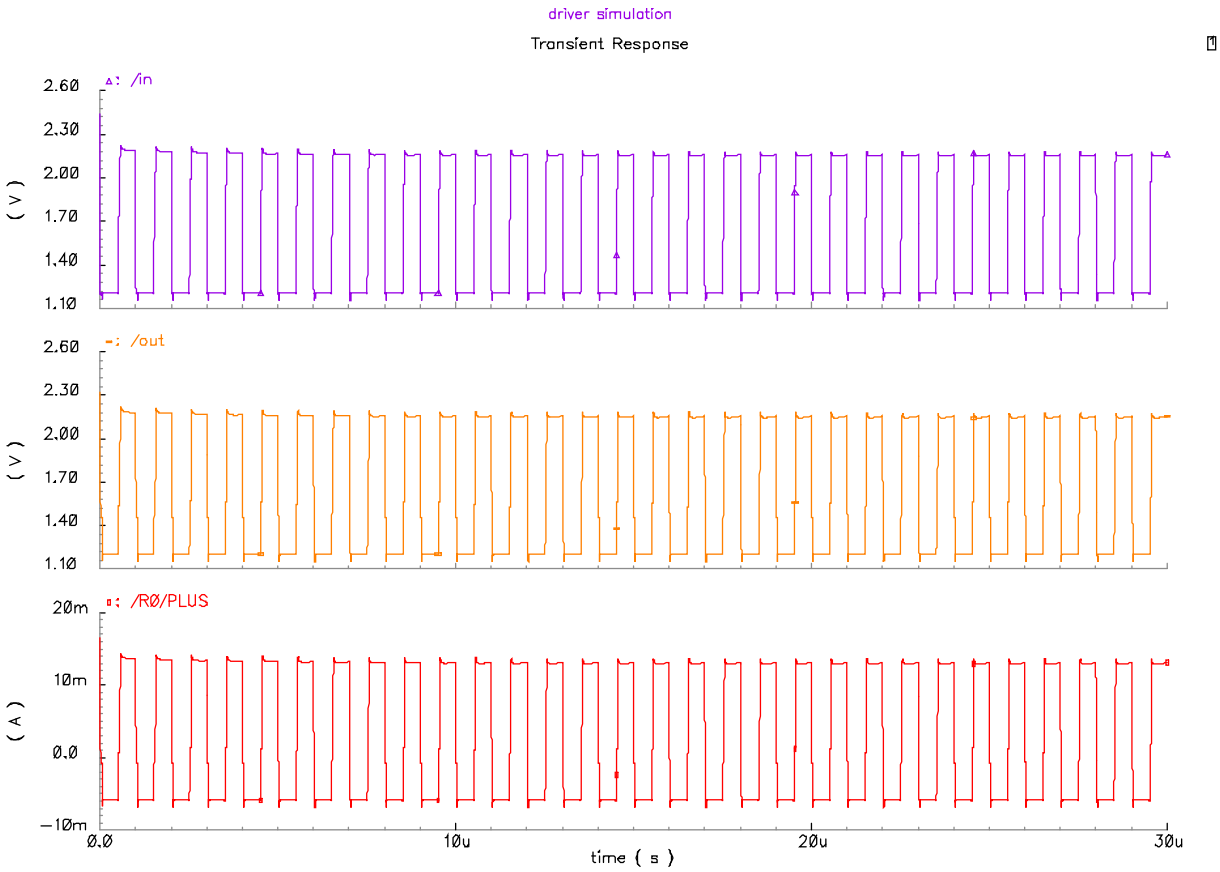


Abbildung 3-29: Einschwingverhalten des Leistungsoperationsverstärkers

Das Einschwingverhalten des analogen Leistungsverstärkers ist in der Abbildung 3-29 dargestellt. Die erste Kurve von oben repräsentiert das Eingangssignal, die mittlere das Ausgangssignal. Die letzte Kennlinie zeigt den resultierenden Strom durch einen 50Ω Ausgangswiderstand. Für die Simulation ist eine Betriebsspannung von 3 V verwendet worden. Die maximale Abtastfrequenz beläuft sich nach Simulation auf mehr als 1 MHz. Insgesamt steht ein Aussteuerbereich von ±0.7 V bei einer Versorgungsspannung von ±1.5 V zur Verfügung.

3.6 Komparatoren

Im Rahmen einer Diplomarbeit [Zhan08] wurde für verschiedene CMOS-Komparatoren mit Hilfe der Modulgeneratorumgebung das Layout erstellt. Die Komparatoren wurden in einer 0.25µm CMOS Technologie der Firma IHP GmbH entworfen und integriert. Die Schaltungen sind für eine Versorgungsspannung von ± 1 V ausgelegt.

3.6.1 Dreistufiger Komparator

Der Aufbau eines dreistufigen Komparators ist der Abbildung 3-30 zu entnehmen. Die ersten beiden Stufen entsprechen denen eines zweistufigen Operationsverstärkers, jedoch ohne dessen Kompensationsnetz.

Im Gegensatz zum Operationsverstärker kann die Eingangsstufe sich beim Anlegen eines Eingangssignals bereits in der Sättigung befinden. Der Ausgangshub für den Komparator beträgt ±1 V. Neben der verstärkenden Eingangsdifferenzstufe verfügt der Komparator über eine weitere verstärkende Stufe. Um den Ausgangspegel auf V_{DD} bzw. V_{SS} regeln zu können, wird ferner ein CMOS Inverter verwendet.

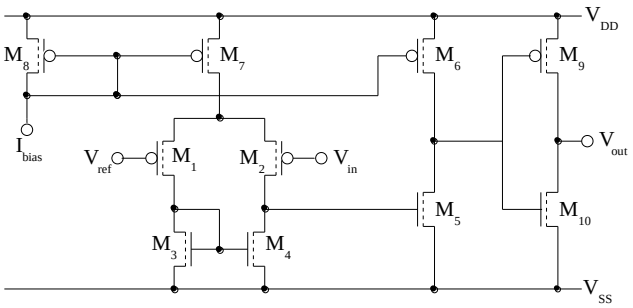


Abbildung 3-30: Schaltbild des dreistufigen Komparators

3.6.1.1 Layout des dreistufigen Komparators

Das Layout des Komparators ist in Abbildung 3-31 gezeigt. Für die Differenzstufe, realisiert durch die Transistoren M_1 und M_2, ist eine meanderförmige Struktur gewählt worden. Um eine Symmetrie zu sichern, sind im Layout auf beiden Seiten der Differenzstufe zusätzlich Dummy-Transistoren hinzugefügt worden. Das Modul befindet sich links unten. Oberhalb der Differenzstufe ist der Stromspiegel (M_B, M_6, M_7) angeordnet. Der Inverter (M_8, M_9) ist rechts oben zu finden. Darunter liegen der Transistor M_5 und die aktive Last der Differenzstufe (M_3, M_4).

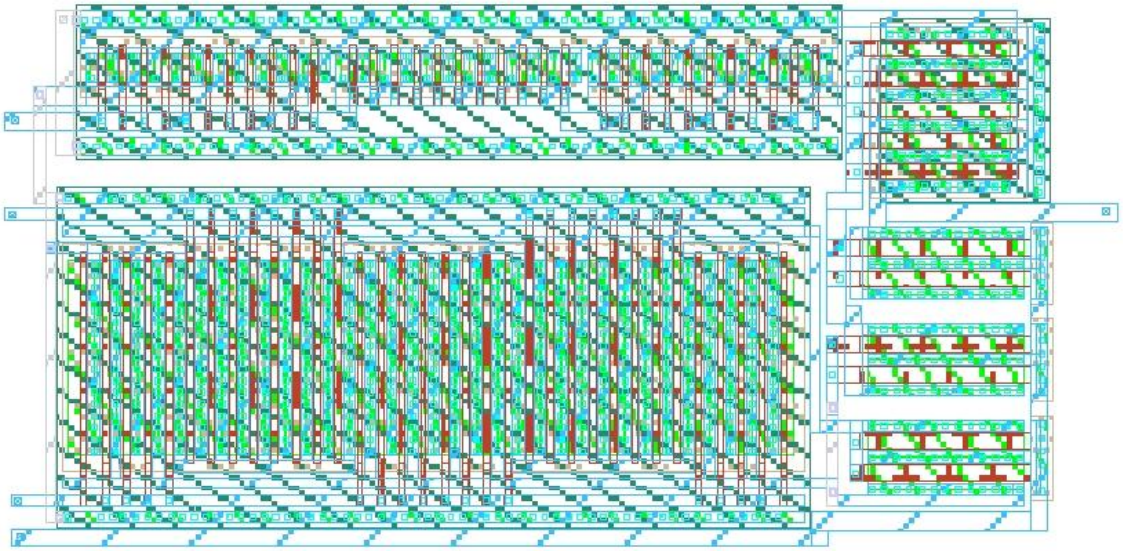


Abbildung 3-31: Layout des dreistufigen Komparators

3.6.1.2 Simulationsergebnisse

Die Simulation des dreistufigen Komparators erfolgt mit einer Betriebsspannung von 2 V. Die minimale Eingangsspannung beträgt ±1 mV. Damit ist es möglich, eine Auflösung von 9-Bit zu erzielen. Die Ausgangscharakteristik bei Verwendung eines Rechteckeingangsimpulses ist in Abbildung 3-32 gezeigt. Der Komparator weist nach Simulation eine Antwortzeit von 30 ns auf.

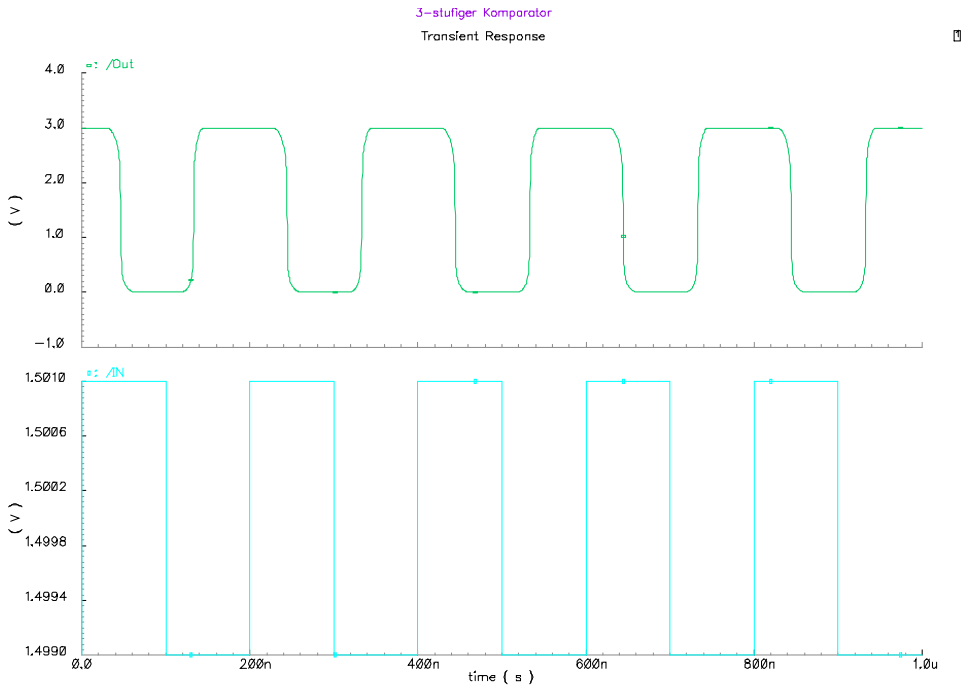


Abbildung 3-32: Ein- und Ausgangskennlinie des dreistufigen Komparators

3.6.2 Dreistufiger SC-Komparator

Die Eingangsoffset-Spannung stellt ein sehr großes Problem beim Entwurf von Komparatoren dar. Insbesondere in hoch genauen Applikationen, wie beispielsweise bei A/D-Umsetzern, kann ein großes Eingangsoffset nicht toleriert werden. Ein gutes Design trägt dazu bei, dass ein systematisches Offset nahezu eliminiert werden kann, während statistische Offsets verbleiben und nicht vorhergesagt werden können [Palo03]. Eine Möglichkeit zur Verringerung des Offsets ist durch eine Vergrößerung der Eingangstransistoren gegeben, jedoch wird auf diese Weise die resultierende Schaltung infolge der erhöhten parasitären Kapazitäten langsamer. Eine andere Lösungsmöglichkeit besteht in der Verwendung von Schalter-Kondensator-Komparatoren. Bei dieser zeitdiskret arbeitenden Schaltung wird der Eingangsoffset mit Hilfe eines Selbstabgleichs minimiert. Die Funktionsweise wird anhand von Abbildung 3-33 erläutert.

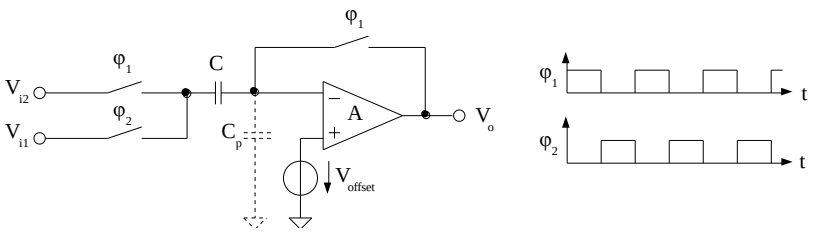


Abbildung 3-33: Komparator mit Selbstabgleich

Die Eingangsspannung V_{i1} - V_{i2} wird auf zwei Phasen verteilt eingelesen. Während der ersten Phase ϕ_1 wird auf der Kapazität C die Spannung V_{i2} - V_{offset} gespeichert und der Verstärker wird als Spannungsfolger beschaltet. In der zweiten Phase ϕ_2 wird die Eingangsspannung V_{i1} eingelesen und der Verstärker arbeitet als Komparator. Mit der parasitären Kapazität C_p und der Kapazität C ergibt sich ein kapazitiver Spannungsteiler, der zu einem vom Kapazitätsverhältnis abhängigen Fehler führt. Infolgedessen hängt die Ausgangsspannung des Verstärkers lediglich von der Eingangsspannung ab und ist bei einer hinreichenden Verstärkung A und einem großen Kapazitätsverhältnis C/C_p nahezu frei von einer Offsetspannung.

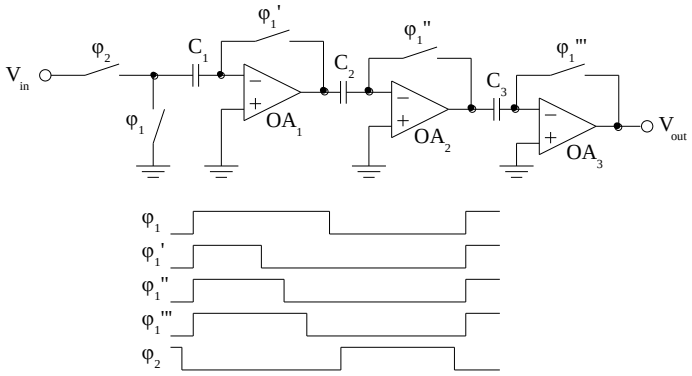


Abbildung 3-34: Blockschaltbild des dreistufigen SC-Komparators

Das Problem bei SC-Komparatoren liegt in der Ladungsinjektion. Die Realisierung der Schalter erfolgt mit Hilfe von Transistoren. Wenn diese ausgeschaltet werden, treten unerwünschte Ladungen auf, die in die Schaltung injiziert werden. Zur Umgehung dieses Problems werden für jeden Schalter zwei komplementäre Transistoren verwendet, so dass es zu einer gleichzeitigen Injektion von negativen und positiven Ladungen kommt. Auf diese Weise soll die Ladungsinjektion in erster Näherung kompensiert werden. Eine weitere Minimierung des Fehlers infolge von Ladungsinjektionen besteht in der Anwendung von volldifferentiellen Designtechniken für Komparatoren [John97]. Diese Variante wird anhand eines dreistufigen SC-Komparators beschrieben, dessen Aufbau und das zugehörige Phasenverhältnis der Takte in Abbildung 3-34 dargestellt sind.

Bei einem Wechsel des Taktes ϕ_1' von einem High- zu einem Low-Pegel injiziert der zugehörige Schalter, durch parasitäre Kapazitäten, Ladungen. Diese werden sowohl in dem invertierenden Eingang als auch in dem Ausgang der ersten Stufe gespeichert. Dadurch wird am Ausgang der ersten Stufe ein temporärer Glitch erzeugt. Die am invertierenden Eingang injizierten Ladungen $Q_{inj,1}$ fließen in die Kapazität C_1 und verursachen eine Offsetspannung $V_{off,1} = Q_{inj,1}/C_1$. Der Offset wird durch den Verstärker OA_1 um den Faktor A_0 vergrößert. Da der zur Taktphase ϕ_1'' korrespondierende Schalter noch geöffnet ist, wird auf der Kapazität C_2 die Spannung $A_0V_{off,1}$ gespeichert. Auf diese Weise wird der Effekt durch den Eingangsoffset reduziert. Entsprechend wird der Offset der nächsten Stufe auf der Kapazität C_3 gespeichert und dessen Einfluss minimiert.

3.6.2.1 Layout des dreistufigen SC-Komparators

Für den dreistufigen SC-Komparator ist in Abbildung 3-35 das Layout gezeigt. Im oberen Bereich des Layouts sind die analogen Komponenten der Schaltung in Form von drei Differenzstufen und einem Ausgangstreiber, der mittels eines Inverters modelliert wird, platziert worden. Den Strukturen folgen die integrierten Kapazitäten. Diese trennen den analogen Bereich des Komparators vom digitalen. Dieser besteht aus fünf Schaltern und den zugehörigen Taktleitungen. Das Layout der verwendeten Differenzstufen setzt sich jeweils aus einem meanderförmig angeordneten, differentiellen Eingangspaar, sowie einem oberhalb davon platzierten Stromspiegel und einer aktiven Last zusammen.

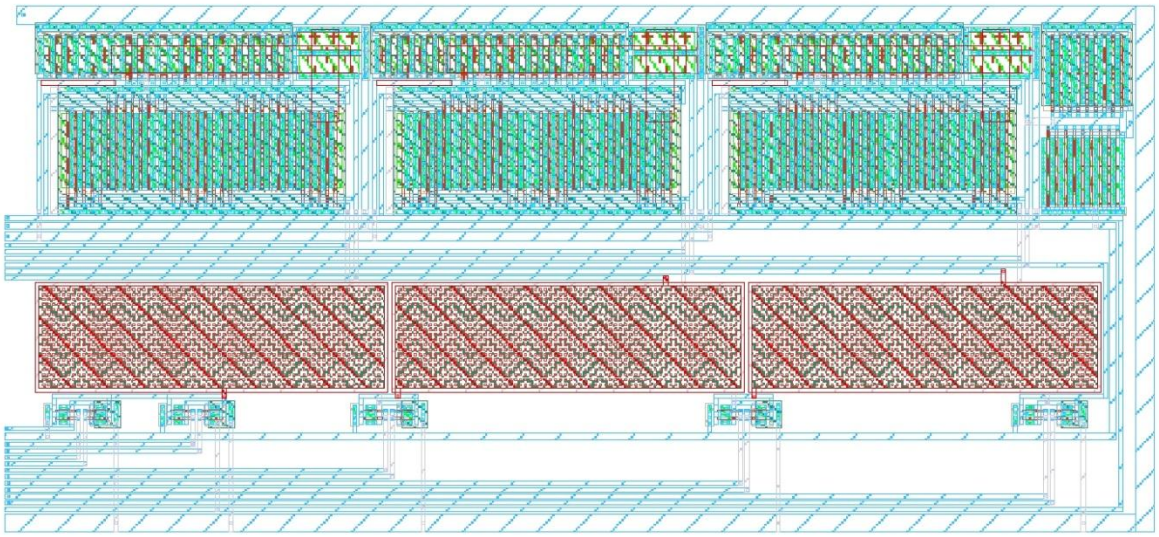


Abbildung 3-35: Layout des dreistufigen SC-Komparators

3.6.2.2 Simulationsergebnisse

Für die Simulation des dreistufigen SC-Komparators werden eine Versorgungsspannung von 2 V und ein Biasstrom von 150 µA eingestellt. Die Taktverhältnisse der fünf verschiedenen Schalter ist der Abbildung 3-36 zu entnehmen. Um Ladungsinjektionen zu vermeiden und um den Offset zu minimieren, dürfen sich die Takte clk_1 und clk_2 nicht überlappen. Es muss ferner die Bedingung $clk_1 > clk'_1 > clk''_1 > clk'''_1$ erfüllt sein. Das simulierte Übergangsverhalten des Komparators wird in Abbildung 3-37 präsentiert. Die Antwortzeit beträgt nach Auswertung der Simulationsdaten ca. 10 ns.

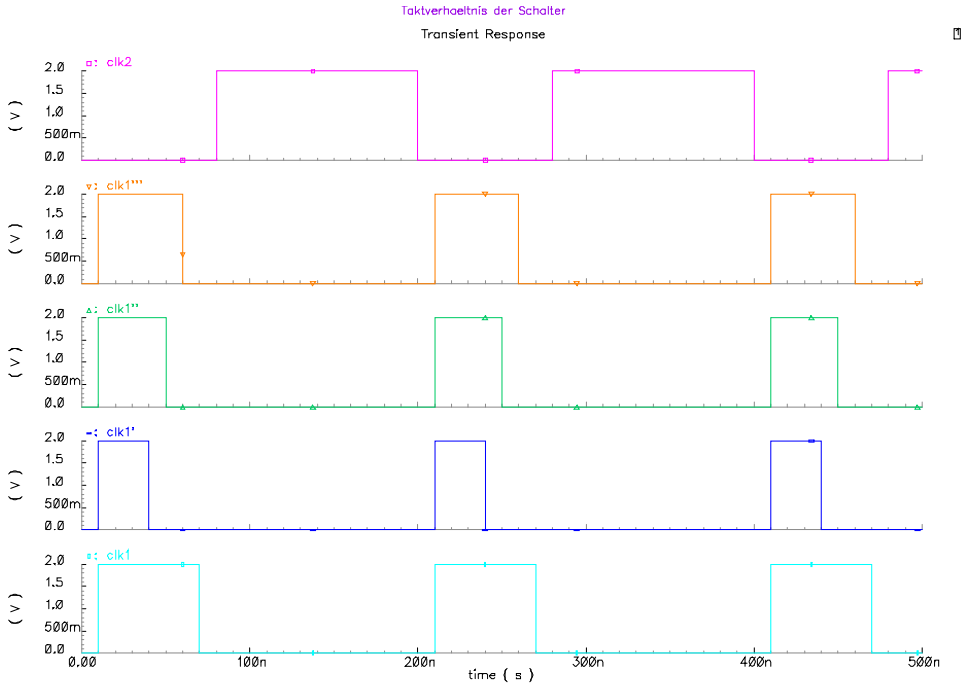


Abbildung 3-36: Taktverhältnis der verwendeten Schalter

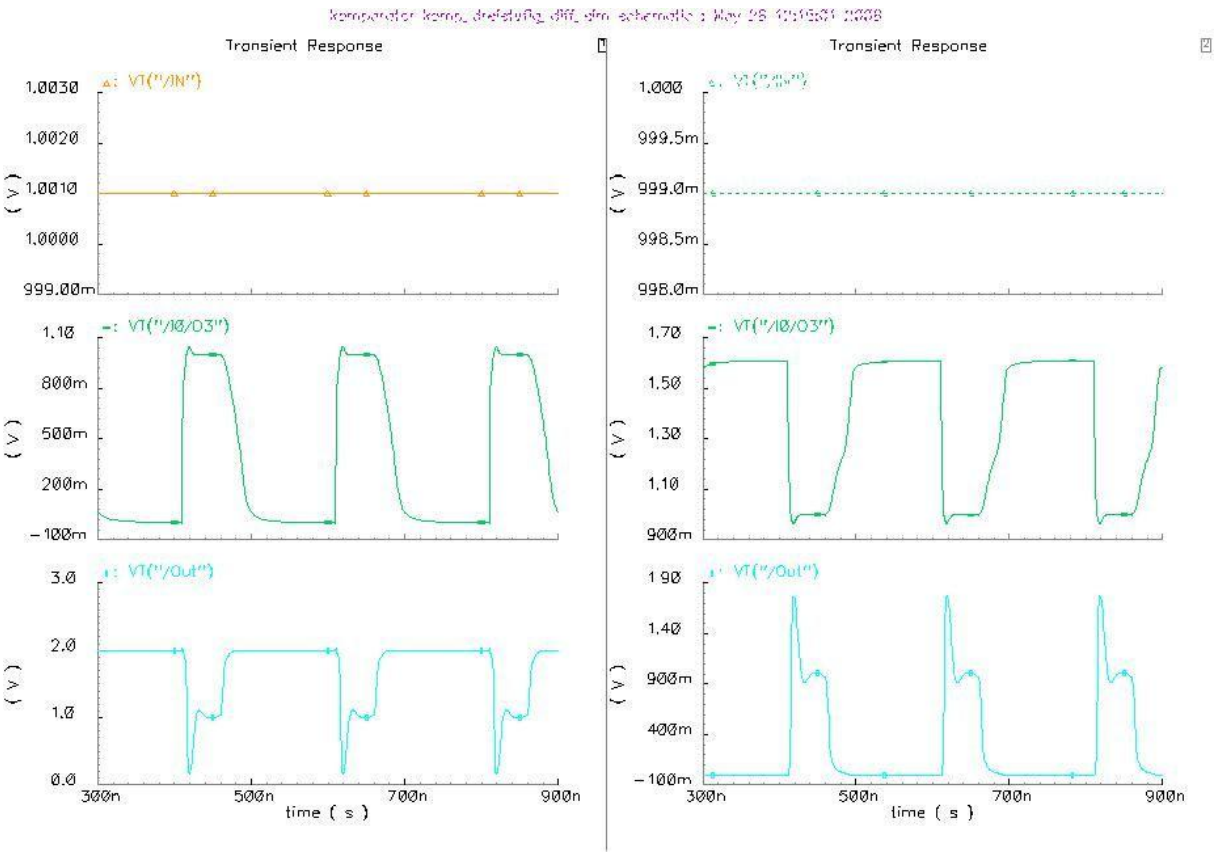


Abbildung 3-37: Simulationsergebnisse des SC-Komparators

3.6.3 Regenerativer Komparator

Das Design und die Simulationsergebnisse eines regenerativen Komparators werden in diesem Abschnitt beschrieben. Der Komparator besteht aus einem volldifferentiellen Vorverstärker, einem Latch und einer Inverterstufe (siehe Abbildung 3-38).

Der Einsatz des Vorverstärkers ermöglicht es, eine höhere Auflösung zu erzielen. Auf diese Weise weist der Ausgang ein betragsmäßig größeres Signal auf als der Komparator-Eingang. Dennoch ist der Spannungspegel zu gering, um eine anschließende digitale Schaltungskomponente treiben zu können. Aus diesem Grund wird eine Track-und-Latch Stufe eingesetzt. Diese verstärkt das Ausgangssignal während der Track- bzw. Latch-Phase. Die in der Latch-Phase ermöglichte, positive Rückkopplung überführt das Analogsignal in ein digitales Signal. Ferner minimiert die Track-und-Latch Stufe die Anzahl der andernfalls erforderlichen Verstärkerstufen, um eine hinreichende Auflösung zu erreichen.

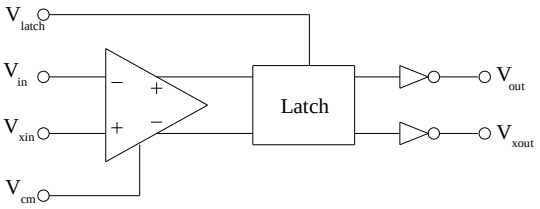


Abbildung 3-38: Struktur des regenerativen Komparators

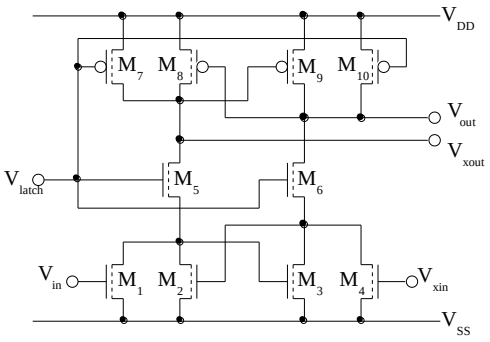


Abbildung 3-39: Track-und-Latch Stufe

3.6.3.1 Layout des regenerativen Komparators

In Abbildung 3-40 ist das Layout des regenerativen Komparators dargestellt. Die Track-und-Latch Stufe ist auf der rechten Seiten positioniert worden. Unterhalb von dieser Struktur wurden die Inverterstufen platziert, um ein möglichst kompaktes Layout zu erzielen. Die verbleibende Fläche wird durch den volldifferentiellen Vorverstärker belegt.

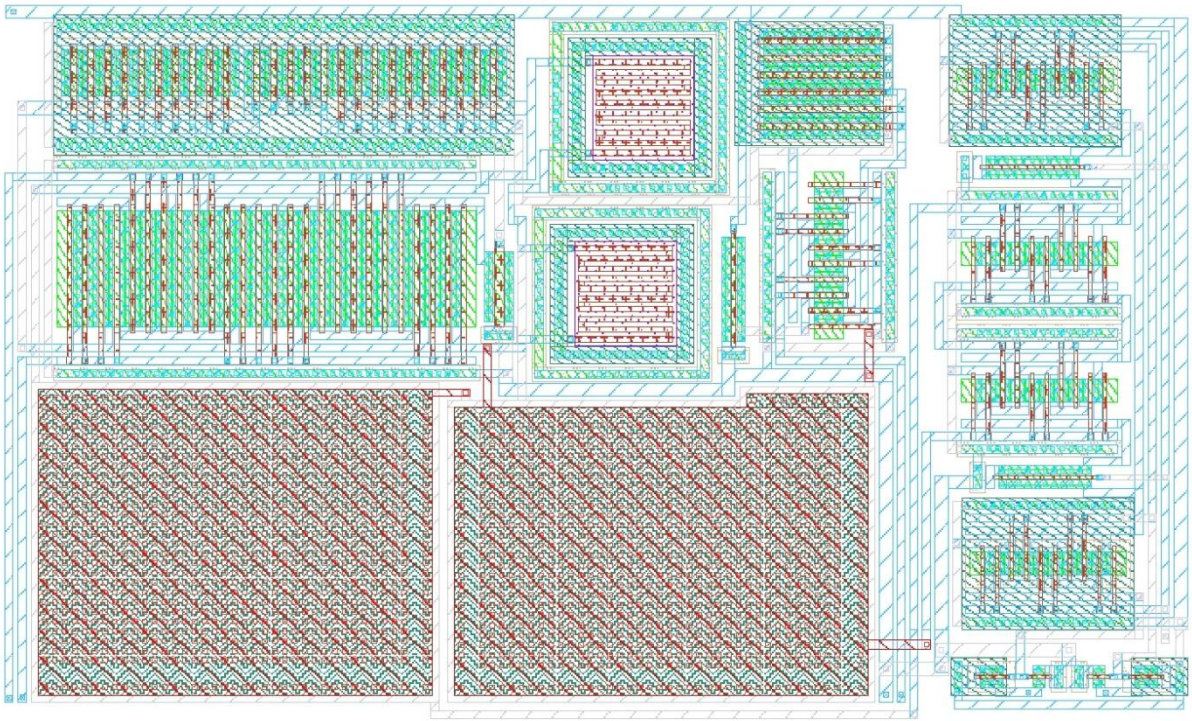


Abbildung 3-40: Layout des regenerativen Komparators

3.6.3.2 Simulationsergebnisse

Das simulierte Übergangsverhalten des regenerativen Komparators ist in Abbildung 3-41 gezeigt. Für die Simulation ist eine Betriebsspannung von 2 V verwendet worden. Das Taktsignal des Latches, welches eine Anstiegszeit von 1 ns aufweist, wird durch die Kennlinie 1, die Ausgänge des Latches werden mittels der Kennlinien 2 und 3 dargestellt. Infolge der Sättigungsspannung wird für den High-Pegel lediglich ein Spannungsniveau von 1,7 V erzielt (siehe Kennlinie 3). Aus diesem Grund wird eine Inverterstufe benutzt, so dass die Logikniveaus von 0 V bzw. 2 V erreicht werden können (siehe Kennlinien 4 und 5). Insgesamt benötigt der Komparator, nach Simulation, ca. 1,3 ns um das entsprechende Ausgangsniveau zu erreichen.

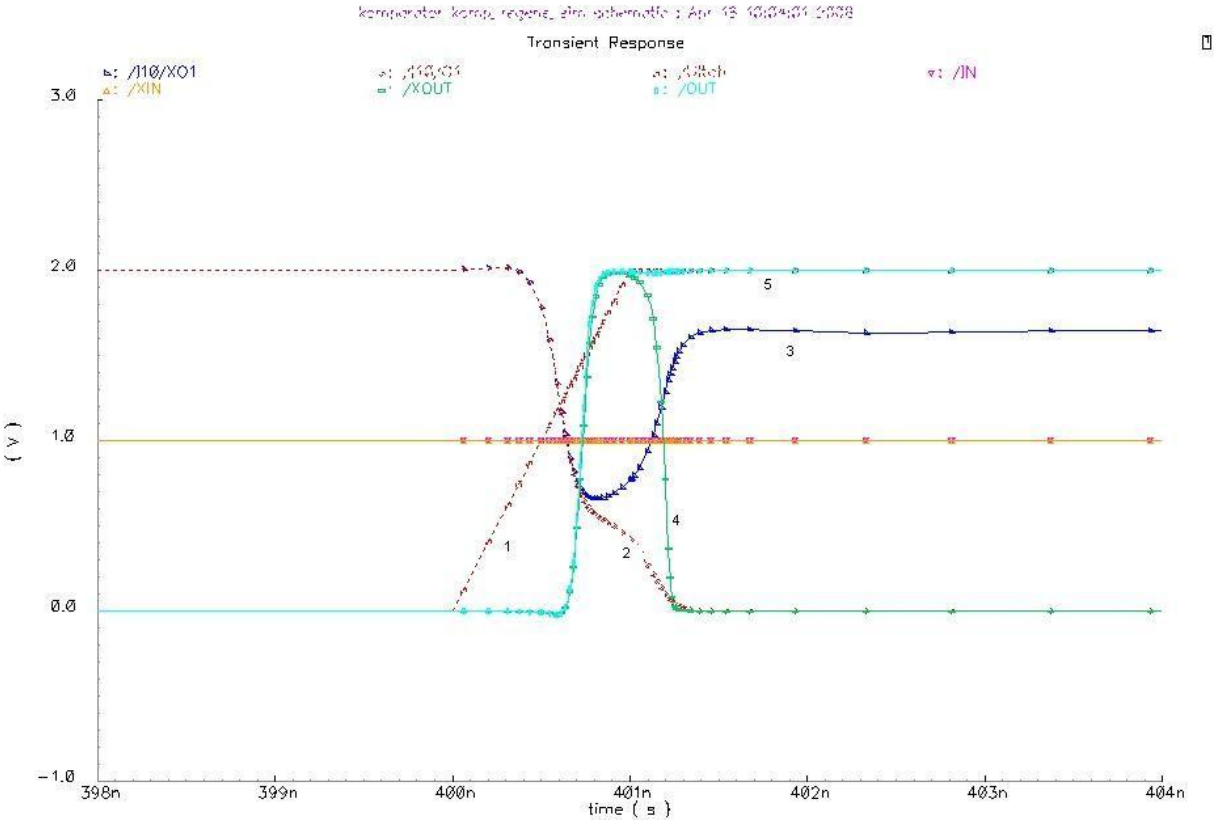


Abbildung 3-41: Simulationsergebnisse des regnerativen Komparators

4 Bekannte Konzepte zur Platzierung und Verdrahtung

Das Ziel der Platzierung und Verdrahtung bei der Layout-Synthese integrierter Schaltungen ist die optimale Anordnung der Module sowie der Verlauf der zugehörigen Verbindungen bzgl. einer gegebenen Kostenfunktion unter Einhaltung technologiespezifischer Randbedingungen. Dazu sind eine Menge von Zellen und eine Netzliste gegeben, die die Verbindungen zwischen den Zellen eindeutig beschreibt. Mathematisch gesehen stellt das Layoutproblem ein gemischt kombinatorisch kontinuierliches Optimierungsproblem mit Nebenbedingungen dar. Eine exakte Lösung in polynomialer Laufzeit unter Verwendung deterministischer Verfahren ist nicht möglich, da es sich um ein NP-vollständiges Problem handelt [Brüc93].

Generell ist bei der Platzierung eine Vielzahl an Randbedingungen zu beachten. Beispielsweise darf kein Übersprechen zwischen den Leitungen auftreten, die maximale Verlustleistung pro Fläche ist hinreichend gering zu halten und auf eine Minimierung der parasitären Effekte ist zu achten. Die Liste der Nebenbedingungen lässt sich weiter fortsetzen, in Abhängigkeit von den weiteren Anforderungen, die an das zu erstellende Layout gestellt werden. In der Kostenfunktion sind damit unterschiedliche Parameter zu berücksichtigen, wobei das primäre Optimierungsziel in der Regel die Minimierung der Gesamtfläche darstellt.

Mit der Platzierung sollte die Verdrahtung einhergehen. Infolge der Komplexität der entsprechenden Verfahren werden beide Vorgänge sequentiell ausgeführt. Diese Vorgehensweise reduziert den Umfang des Layoutproblems. Die Nachteile bestehen darin, dass es unter Umständen zum einen für eine vorgegebene Platzierung keine Verdrahtungsmöglichkeit gibt, zum anderen ist die Bewertung der Platzierungsqualität nur eingeschränkt möglich. Für einige Bewertungskriterien wird die Verdrahtung benötigt. Bei einer getrennten Verdrahtung und Platzierung stehen diese jedoch nicht zur Verfügung, so dass die Verdrahtung während der Platzierungsphase abgeschätzt werden muss. Die Abschätzung kann jedoch dazu führen, dass eine Platzierung bereits im Vorfeld verworfen wird, obwohl diese nach Ausführung der Verdrahtung im Hinblick auf die Kostenfunktion günstiger als eine andere wäre.

In den nachfolgenden Abschnitten werden derzeitige Lösungsansätze und -verfahren zur Platzierung und Verdrahtung integrierter Schaltungen näher beschrieben. Ein allgemeiner Überblick zur Platzierung, in der Regel mit einem Schwerpunkt auf digitale Schaltungen, ist in [Breu72, Brüc93, Cohn91, Laut87, Marw93, Otte80] gegeben. Ein Bezug zu analogen Schaltungen ist in [Cohn94, Lovr91, Meye94, Zhan03] zusammengefasst.

4.1 Platzierungsverfahren

Das Ziel der Platzierung ist eine überlappungsfreie Anordnung von Zellen, die eine automatische Verdrahtung ermöglicht. Als Eingabedaten dienen eine Netzliste und physikalische Zellen. Die Netzliste stellt die Verbindungen der Bauelemente dar und repräsentiert damit den Schaltungsgraphen. Es ist nun die geometrische Anordnung der Bauelemente und der zugehörigen Verbindungen auf der Layoutfläche gesucht. Die Platzierung bildet damit ein Optimierungsproblem mit Nebenbedingungen. Zu einer gegebenen Anzahl Zellen ist eine Anordnung zu ermitteln, die unterschiedlichen Optimierungskriterien genügt. Jedes Design weist in seiner Spezifikation Eigenschaften auf, die einen entscheidenden Einfluss auf die jeweiligen Kriterien ausüben. Ein Ziel ist beispielsweise die Minimierung der Layoutfläche, die sich aus der Fläche für die Zellen und der Verdrahtung zusammensetzt. Die Zellen sind derart zu platzieren, dass die Fläche und die

Gesamtverdrahtungslänge minimal sind. Da die tatsächliche Verdrahtungslänge während der Platzierungsphase nicht zur Verfügung steht, muss diese abgeschätzt werden. Die genaue Berechnung ist in den meisten Fällen zu zeitaufwendig. Es wird in der Regel der euklidische oder der Manhattan-Abstand verwendet. Bei digitalen Schaltungen ist bei einer ausreichenden Anzahl an Verdrahtungsebenen eine Verbindung über die Zellen möglich. Auf diese Weise entfällt die Verdrahtungsfläche. Infolge der resultierenden Streukapazitäten ist diese Vorgehensweise bei analogen Schaltungen nicht zu empfehlen.

Ein weiteres Optimierungskriterium besteht darin, eine Anordnung der Zellen so zu ermitteln, dass eine automatische Verdrahtung gewährleistet ist.

Weitere Kriterien sind die Reduzierung von Streukapazitäten oder technologiebedingte Randbedingungen. Streukapazitäten sind parasitäre Effekte, die aus der elektromagnetischen Kopplung von Leitungen hervorgehen. Insbesondere bei Signalleitungen kann bei einer Verfälschung des Signalpegels die Funktionsfähigkeit der Schaltung beeinträchtigt werden. Mit den technologiespezifischen Randbedingungen ist die Einhaltung der Entwurfsregeln der verwendeten Technologie gemeint.

Die Vorgehensweisen zur Platzierung lassen sich grundlegend in zwei Gruppen unterteilen, die iterativen und die konstruktiven Verfahren. Beim konstruktiven Ablauf werden schrittweise die Zellen auf der anfangs leeren Layoutfläche angeordnet. Bereits gesetzte Zellen werden in der Regel nicht mehr zurückgenommen. Dazu zählen die im Folgenden beschriebene kräftegesteuerte Platzierung und die Min-Cut-Platzierung. Im Gegensatz dazu führen die iterativen Verfahren Veränderungen an einer gegebenen Initialplatzierung durch und bewerten diese mittels einer Kostenfunktion. Basierend auf dem Ergebnis der Beurteilung wird die Modifikation angewendet oder verworfen. Der Vorgang wird bis zum Erreichen eines definierten Abbruchkriteriums wiederholt. Der Simulated-Annealing-Algorithmus und die genetischen Algorithmen stellen jeweils eine Implementierung der iterativen Platzierung dar. Diese werden ebenfalls in den folgenden Abschnitten näher beschrieben und diskutiert.

4.1.1 Kräftegesteuerte Platzierung

Die kräftegesteuerte Platzierung ist das am weitesten verbreitete konstruktive Verfahren. Hierbei wird eine Initialplatzierung iterativ verbessert [Kaya88, Eise98, Hu02, Vorw04]. Dazu wird das Platzierungsproblem in ein klassisches mechanisches Feder-Masse-System überführt. Die Zellen der Schaltung entsprechen den Massen und die Netzverbindungen den Federn. Die Kraft verhält sich dabei proportional zur Entfernung der Zellen zueinander. Gesucht sind diejenigen Positionen der Zellen, bei denen es zu der geringsten Abweichung vom Kräftegleichgewicht kommt. Während des Verfahrens wird jeweils eine Zelle zur Optimierung der Gesamtverdrahtungslänge in Richtung des Schwerpunkts der errechneten Kräfte zwischen den Zellen verschoben.

Die Berechnung der Kosten eines Netzes erfolgt in der Regel durch den quadratischen euklidischen Abstand zwischen je zwei Zellen, d.h. die kräftegesteuerte Platzierung basiert auf der quadratischen Optimierung. Damit ergeben sich die nachstehenden Verbindungskosten

$$c_{ij}(k) = w_k \cdot \left\| \vec{x}_i - \vec{x}_j \right\|_2^2, \tag{4-1}$$

wobei w_k dem Netzgewicht der Verbindung k zwischen den Zellen i und j entspricht, x_i die Position der Zelle i und x_j die Position der Zelle j beschreibt. Einige Implementierungen verwenden auch

lineare Kostenfunktionen zur Berechnung der Verbindungskosten. Eine Gegenüberstellung von linearer und quadratischer Kostenfunktion sowie eine Diskussion der Vor- und Nachteile sind in [Sigl91] aufgeführt. Eine Anwendung der kräftegesteuerten Platzierung ist beispielsweise im Synthese-Werkzeug SALIM zu finden [Kaya88]. Die Verbindungen zwischen den zu platzierenden Zellen werden dabei als Federn modelliert.

Die Standardimplementierung der kräftegesteuerten Platzierung weist bei der Berechnung der anziehenden und abstoßenden Kräfte eine Laufzeit von O(n· log n) auf, wobei n die Anzahl der zu platzierenden Zellen ist [Klei88]. Ausgehend von der Tatsache, dass der Zeitbedarf von der Dauer der Lösung des linearen Gleichungssystems abhängig ist, kann die Laufzeitkomplexität durch Verwendung eines iterativen Verfahrens zur Berechnung der Gleichungen auf $O(n^{3/2})$reduziert werden [Spir90]. Damit ist die kräftegesteuerte Platzierung eines der schnellsten Platzierungsverfahren.

Der Nachteil des Verfahrens besteht darin, dass nach Abschluss der Platzierung ein sogenannter Legalisierungsschritt ausgeführt werden muss. Während der Ausführung der Platzierung, basierend auf den Resultaten aus dem linearen Gleichungssystem, treten bei den Zellen häufig Überlappungen auf. Diese Überlappungen werden in einem abschließenden Schritt, der Legalisierung, aufgelöst. Dabei kann die ursprünglich erzielte Platzierung signifikant verändert werden. Ein weiterer Nachteil ist dadurch gegeben, dass auf Nebenbedingungen, wie besondere Anforderungen für bestimmte Netze, nur geringfügig eingegangen werden kann. Die einzige Möglichkeit eine Gleichbehandlung von Zellen bzw. Netzen zu umgehen, ist durch eine Erhöhung der Verbindungsgewichte für kritische Netze gegeben. Auf diese Weise wird die Anziehungskraft der entsprechenden Zellen erhöht. In [Chou01, Eise98, Kong02] sind Ansätze für eine stärkere Anbindung von Randbedingungen in die kräftegesteuerte Platzierung vorgestellt worden. Dennoch ist keine der Modifikationen in der Lage, eine Garantie für die Erstellung eines funktionsfähigen Layouts zu geben. Es wird zwar ein Platzierungsergebnis erreicht, welches besser als das aus der Standardimplementierung ist, jedoch können Platzierungen auftreten, die nicht unter Einhaltung der gegebenen Restriktionen verdrahtbar sind [Malo04].

4.1.2 Min-Cut-Platzierung

Die Min-Cut-Platzierung ist ein konstruktives Top-Down-Platzierungsverfahren. Die Vorgehensweise besteht in einer rekursiven Aufteilung der zu platzierenden Zellen und der Layoutfläche, bis jede Teilfläche maximal eine Zelle einschließt. Die Zellen und die zugehörigen Verbindungen werden bei diesem Verfahren als Graph repräsentiert. Der generelle Ablauf des Verfahrens ist in Algorithmus 4-1 skizziert. Bei der Aufteilung einer Partition in Teilpartitionen wird darauf geachtet, dass möglichst wenige Netzverbindungen zwischen den Zellen der resultierenden Teilflächen existieren. Das Optimierungskriterium ist damit durch eine Minimierung der Verbindungsschnitte gegeben. Auf diese Weise soll die Gesamtverdrahtungslänge reduziert werden. Der Optimierungsprozess erfolgt mit Hilfe von Heuristiken, wie der nach Kerninghan und Lin [Kern70] bzw. der nach Fiduccia und Mattheyses [Fidu82]. Beide genannten Verfahren spalten den Graph der zu platzierenden Zellen in zwei Teilgraphen unter Berücksichtigung minimaler Schnittkosten. Die Platzierungsfläche wird sequentiell mit Schnittlinien durchzogen. Die Zellen der Teilgraphen werden auf die entstehenden Teilflächen derart verteilt, dass die Anzahl der die Schnittlinien kreuzenden Netze minimiert werden. Diese Tatsache hat dem Verfahren den Namen "Min-Cut-Platzierung" gegeben.

Algorithmus 4-1: Min-Cut-Verfahren

```
Aufteilung der Layoutfläche in zwei annähernd gleich große Teilflächen mit
vertikaler bzw. horizontaler Schnittrichtung;
solange jede Teilfläche enthält mehr als eine Zelle
   Anwendung eines geeigneten Algorithmus (Kernighan-Lin oder Fiduccia-
   Mattheyses) zur optimierten Verteilung der Zellen auf die beiden
   Teilflächen;
   Aufteilung in eine neue Teilfläche und entsprechende Initialzuordnung der
   Zellen;
   alternierender Wechsel zwischen senkrechter und waagerechter Schnittrichtung;
solange_ende
```

In [Sarr97] wird die Min-Cut-Platzierung mit dem Simulated-Annealing-Verfahren (siehe Kap. 4.1.3) kombiniert. Es wird zunächst eine globale Anordnung der Zellen mittels der Min-Cut-Partitionierung durchgeführt. Jeder Layoutteilfläche werden entsprechend der Netzlisten-Partitionierung Zellen zugewiesen, die auf dem Mittelpunkt der Fläche platziert werden. Die Optimierung der Globalplatzierung erfolgt mit einem Simulated-Annealing-Ansatz. Jeweils zwei Zellen aus verschiedenen Partitionen werden für die Vertauschung ausgewählt. Die Detailplatzierung innerhalb einer Teilfläche erfolgt getrennt von den anderen Partitionen. Es wird dabei ebenfalls auf das Simulated-Annealing-Verfahren zurückgegriffen.

In [Wang00] erfolgt die Globalplatzierung ebenfalls unter Verwendung des Min-Cut-Verfahrens, während die Detailplatzierung mit Hilfe eines Greedy-Algorithmus durchgeführt wird. Die Zellen werden auf der Layoutfläche verteilt. Anschließend werden die Netzlängen durch heuristische Vertauschungsoperationen optimiert.

Das Platzierungswerkzeug PALM [Lovr91], welches Bestandteil des Designpakets ALSYN [Meye93] ist, verwendet ebenfalls das Min-Cut-Verfahren. In jedem Teilungsschritt werden die Zellen derart auf die resultierenden Teilmengen verteilt, so dass die Flächen näherungsweise gleich groß sind. Der Teilungsalgorithmus basiert auf dem Verfahren von Fiduccia und Mattheyses (siehe Kap. 4.1.2.2). Es wird eine modifizierte Datenstruktur verwendet, die den Einsatz gewichteter Netze erlaubt.

4.1.2.1 Kerninghan-Lin Heuristik

Die Basis des Partitionierungsalgorithmus ist das Verfahren nach B. Kerninghan und S. Lin [Kern70]. Der Algorithmus wird heute in erster Linie zur Verbesserung von bereits berechneten Partitionen verwendet. Die Anzahl der Verbindungen einer gegebenen Bipartition (Halbierungsplatzierung) wird durch paarweises Vertauschen der Zellen unter Berücksichtigung von vorgeschriebenen Randbedingungen minimiert. Die Beschränkungen können beispielsweise die Schnittgröße bzw. die Balance hinsichtlich der Größe der beiden entstandenen Teilflächen enthalten.

Um eine Initialplatzierung zu erhalten, wird bei dem Verfahren im ersten Schritt die Menge der Zellen mit ihren zugehörigen Verbindungsnetzen in zunächst zwei gleich große Partitionen geteilt. Im Fall einer ungeraden Anzahl an Zellen wird eine entsprechende Dummy-Zelle erstellt. Es wird nun jeweils eine Zelle der einen Partition mit einer Zelle aus der gegenüberliegenden Partition vertauscht, um eine minimale Schnittgröße der Netze zu erzielen. Das Risiko in einem lokalen Minimum zu verharren ist dann gegeben, wenn ausschließlich diejenigen Zellen vertauscht werden, die die größte Schnittgrößenreduzierung zur Folge haben. Zur Vermeidung dieses Problems haben Kerninghan und Lin vorgeschlagen, immer dasjenige Zellenpaar auszuwählen, welches die größte Schnittgrößenreduktion oder die kleinste Schnittgrößenerhöhung mit sich bringt. Im Anschluss an die Vertauschungsoperation werden die getauschten Zellen in der jeweils neuen Partition fixiert und stehen damit für einen weiteren Austausch nicht weiter zur Verfügung. Der Vorgang wird solange

wiederholt, bis alle Zellen fixiert sind. Nach jeder Vertauschung werden die Schnittgröße und das zugehörige Zellenpaar gespeichert. Am Ende wird die kleinste aufgetretene Schnittgröße bestimmt und die korrespondierende Partitionierung rekonstruiert. Mit dieser Vorgehensweise können lokale Minima überwunden werden [Kern70]. Die Komplexität des Verfahrens liegt hinsichtlich der Laufzeit in der Größenordnung $O(n^3)$. Nach [Vahi97] kann mit Hilfe von verbesserten Verfahren die Laufzeit auf $O(n^2 \log n)$ reduziert werden.

4.1.2.2 Fiduccia-Mattheyses Heuristik

Eine Erweiterung des von Kerninghan und Lin vorgeschlagenen Verfahren geht auf Fiduccia und Mattheyses zurück [Fidu82]. Zu jedem Zeitpunkt wird stets nur eine einzelne Zelle in eine andere Partition verschoben. Auf diese Weise entstehen unterschiedlich große Partitionen. Die Schnittgrößenberechnung wird derart modifiziert, dass im Gegensatz zum Verfahren nach Kerninghan und Lin Verbindungskanten mit mehr als zwei Anschlusspunkten verarbeitet werden können.

Um überflüssige Knotenverschiebungen zu vermeiden, wird bei diesem Algorithmus ebenfallseine Zelle in ihrer neuen Partition fixiert. Zur Überwindung der lokalen Minima ist auch hier eine kurzfristige Erhöhung der Schnittgröße beim Auswahlprozess zulässig. Im Anschluss an die Verschiebung sämtlicher Zellen wird diejenige Partition ausgewählt, deren Schnittgröße den niedrigsten Wert zur Folge hat. Das Verfahren wird iterativ solange ausgeführt, bis sich keine weitere Verbesserung einstellt.

Im Vergleich zur vorhergehenden Heuristik ist das Verfahren nach Fiduccia und Mattheyses schneller, da die Laufzeit durch die Anzahl der Verbindungskanten E definiert ist, d.h. der Algorithmus weist die Laufzeit $O(|E|)$ auf. Der Implementierungsaufwand ist in diesem Fall infolge der komplexeren Strukturen jedoch deutlich größer.

4.1.3 Simulated Annealing

Das Simulated Annealing bildet das Ausglühen fester Körper nach. Es ist der bekannteste heuristische Ansatz zur Platzierung [Caso87, Naha86, Sech86, Sech88]. Während des Vorgangs werden Spannungen innerhalb des Körpers entfernt. Nach Abschluss des Verfahrens ergibt sich ein Zustand minimaler Energie.

Bezogen auf das Platzierungsproblem wird eine Initialplatzierung mittels Vertauschungen von einzelnen Zellen iterativ verbessert. Änderungen, die sich günstig auf die Kostenfunktion auswirken, werden immer akzeptiert. Mit Hilfe einer definierten Temperaturkurve und einer Zufallsfunktion wird entschieden, welche Modifikationen, die zu einer Verschlechterung führen, angenommen werden. Durch die Akzeptanz von Verschlechterungen der Kosten, können lokale Minima der Kostenfunktion überwunden werden, um das globale Optimum zu finden. Mit abnehmender Temperatur ist die Wahrscheinlichkeit geringer, dass eine Verschlechterung angenommen wird. Ein Zustand des Gleichgewichts ist stets dann erreicht, wenn eine vorgegebene Anzahl an Durchläufen bei einer konstanten Temperatur erfolgt ist, ohne zu einer Verbesserung zu führen. Die Temperatur wird abgesenkt, wenn der Gleichgewichtszustand erreicht wird. Die Abbruchkriterien für das Verfahren sind durch das Erreichen einer definierten Endtemperatur bzw. Überschreiten einer maximalen Anzahl an Iterationen gegeben. Ferner kann als Abbruch eine vorgegebene Anzahl an Durchläufen gewertet werden, bei der keine verbesserte Lösung erzielt werden kann. Insgesamt ergibt sich damit der in Algorithmus 4-2 skizzierte Ablauf.

Algorithmus 4-2: Simulated Annealing - Prinzipieller Verfahrensablauf

```
bestimme Anfangstemperatur T0;
erstelle Initialplatzierung P0;
T:=T0;
P:=P0;
wiederhole
   solange Gleichgewicht nicht erreicht
      Pneu:=VeränderePlatzierung(P);
      wenn AkzeptierePlatzierung(P,Pneu,T) dann
         P:=Pneu;
      wenn_ende
   solange_ende
   berechne T neu;
bis Abbruchkriterium erfüllt wiederhole_ende
```

Durch das Zulassen einer Verschlechterung der Kostenfunktion können lokale Optima verlassen werden. In [Brüc93] ist gezeigt worden, dass durch eine geeignete Wahl der Gleichgewichtsbedingungen und eines entsprechenden Temperaturverlaufs das Verlassen des globalen Optimums verhindert werden kann.

Dem Simulated Annealing Algorithmus stehen grundlegend drei verschiedene Möglichkeiten zur Darstellung der Platzierungskonfiguration zur Verfügung. Eine Repräsentation ist die ebene Darstellung, welche in [Jeps84] erstmals vorgestellt wurde. Die zu platzierenden Zellen werden durch absolute Koordinaten in einer rasterfreien Ebene dargestellt. Die Modifikationen der Platzierung lassen sich damit auf Translationen und Rotationen der Zellen zurückführen. Während der Platzierungsphase ist ein mögliches illegales Überlappen einzelner Zellen erlaubt, solange keine Einschränkungen bzgl. der relativen Position einer Zelle zu einer anderen Zellengemacht werden. Die unerlaubten Überlappungen werden durch einen gewichteten Term der Kostenfunktion berücksichtigt. Dieser Anteil ist während der Optimierungsphase auf Null zu reduzieren. Die Anwendung der Absolutkoordinaten ist gut geeignet, um Symmetrie-Bedingungen und das Matching-Verhalten von Zellen zu berücksichtigen. Daher ist diese Datenstruktur bei den Werkzeugen KOAN/ANAGRAM II [Cohn91, Cohn94], LAYLA [Lamp95] und PUPPY-A [Mala96] zu finden. Ein entscheidender Nachteil der Darstellung wird in [Sun95] aufgezeigt. Infolge der Komplexität der Kostenfunktion ist der Anteil der nicht gestatteten Überlappungen der endgültigen Platzierung nicht notwendigerweise gleich Null. Es wird damit eine Nachoptimierung notwendig, die die Lücken und Überlappungen entfernt. Dies führt zum einen zu einer erhöhten Laufzeit und zum anderen kann die Lösung negativ beeinflusst werden.

Eine weitere Repräsentation der Platzierung ist durch ein graphenbasiertes Modell gegeben, das "Slicing" Modell. Es wurde zuerst in [Otte80] vorgestellt. Im Gegensatz zur vorherigen Datenstruktur werden die Positionen der Zellen durch die relative Lage zueinander beschrieben. Die Zellen werden in eine Menge von Schnitten eingeteilt, die das Layout rekursiv horizontal und vertikal halbieren. Die Richtung und die untergeordneten Schnittmengen werden in einer Baumstruktur bzw. in einem Äquivalent zur umgekehrten polnischen Notation gespeichert [Wong86]. Auf diese Weise führt der Simulated-Annealing-Algorithmus keine direkten Verschiebungen der Zellen durch, sondern ändert lediglich die relative Lage der Zellen zueinander. Die Änderung erfolgt durch die Modifikation der Einträge der verwendeten Datenstruktur. Damit kann es nicht mehr zu einer unerwünschten Überlappung der Zellen kommen. Eine Anwendung findet das Modell im Layoutsystem ILAC [Rijm89]. Die Menge der erzielbaren Layouttopologien wird durch die relative Anordnung jedoch derart eingeschränkt, dass die Dichte des Layouts abnimmt. Dieser Effekt tritt besonders bei Zellen mit sehr

unterschiedlichen Größen, wie es bei analogen Layouts üblich ist, auf [Bala99]. Ferner lassen sich Randbedingungen wie Symmetrie und Matching nur mit erhöhtem Aufwand miteinbeziehen. Eine Möglichkeit Symmetrien zu berücksichtigen, wird in [Mala91] durch die Einführung von virtuellen Symmetrie-Achsen aufgezeigt.

Der letzte Ansatz zur Repräsentation einer Platzierungskonfiguration ist durch die "SequencePair"-Topologie gegeben [Bala99, Koud06, Mura96, Tam06, Zhan08a]. Die grundlegende Idee für die Darstellung der Platzierung mittels Sequence-Pair besteht darin, ein Paar von zwei Sequenzen der n Zellen zu verwenden. Für jedes Paar von zwei Zellen gibt dieses Sequence-Pair eine horizontale bzw. vertikale Beziehung an. Sei α_A^{-1} die Position der Zelle A in der Folge α und β_A^{-1} entsprechend die Stelle der Zelle A in der Sequenz β, dann lassen sich folgende topologische Beziehungen der Zellen A und B aus der Sequence-Pair-Darstellung ableiten:

- wenn $\alpha_A^{-1} < \alpha_B^{-1}$und $\beta_A^{-1} < \beta_B^{-1}$ gilt, dann befindet sich die Zelle A linksseits von der Zelle B,
- wenn $\alpha_A^{-1} < \alpha_B^{-1}$und $\beta_B^{-1} < \beta_A^{-1}$ gilt, dann befindet sich die Zelle A oberhalb von der Zelle B.

Auf diese Weise ergibt sich ein Satz von Randbedingungen für die Platzierung der Zellen. Jede mögliche Anordnung der Sequenzen beschreibt eine zulässige Platzierung. Die Berechnung einer Lösung weist die Laufzeit $O(n^2)$ auf, wobei n die Anzahl der zu platzierenden Zellen repräsentiert. Die genannte topologische Darstellung eignet sich nicht nur für die Anwendung des Simulated-Annealing-Verfahrens, sondern auch für die genetischen Algorithmen.

Insgesamt liefert das Simulated-Annealing-Verfahren bei hinreichender Laufzeit gute Ergebnisse. Es ist in der Lage lokale Optima zu überwinden und einfach zu implementieren. Die Nachteile bestehen darin, dass es sehr rechenzeitintensiv ist und die Qualität der Ergebnisse abhängig ist von der Wahl der den Ablauf steuernden Parameter. Es erfordert damit Erfahrungen und einige zeitintensive Testläufe. Es werden in der Standardimplementierung alle Netze und Zellen gleichbehandelt. Damit ist es schwierig auf die besonderen Anforderungen einiger Schaltungen einzugehen. Nur über eine Modifikation der Kostenfunktion können bevorzugte Behandlungen erreicht werden [Malo04, Swar95].

4.1.4 Genetische Algorithmen

Die genetischen Algorithmen sind ein heuristisches Lösungsverfahren, das nach dem Prinzip der Evolutionstheorie arbeitet [Mich94]. Es ist ein kontinuierlicher Optimierungsprozess, dessen Erfolge durch Anpassungs- und Überlebensfähigkeit gekennzeichnet sind. In der Natur haben diejenigen Lebewesen, die am besten an die Umgebung angepasst sind, eine größere Überlebenswahrscheinlichkeit. Der Kernbegriff der Evolutionstheorie ist das Prinzip "survial of the fittest" [Darw80]. Die genetischen Algorithmen, die zur Platzierung integrierter Schaltungen verwendet werden, simulieren diesen Evolutionsprozess. Die möglichen Lösungen des Platzierungsproblems werden hierbei als Chromosomen oder Individuen bezeichnet. Sie werden oft durch eine Kette von Symbolen, den Genen, dargestellt. Die Menge der möglichen Lösungen bildet eine Population. Infolgedessen, dass die genetischen Algorithmen ein iteratives Verfahren darstellen, wird jeder Iterationsschritt als Generation bezeichnet. Die Bewertung der Individuen erfolgt nach jeder Iteration mittels der Zuordnung einer Qualität, der sogenannten Fitness. Es wird damit die Qualität der Platzierung beurteilt. Aus den besten Individuen werden im Laufe einer Generation durch Anwendung von genetischen Operatoren neue Lösungen erzeugt. Es stehen insgesamt drei genetischen Operatoren, nämlich Selektion, Mutation und Crossover, zur Verfügung. Den aus den

Operatoren resultierenden Individuen, den Nachkommen, werden Fitnesswerte zugeordnet. Die neue Generation setzt sich damit aus einer bestimmten Anzahl an Individuen aus der vorherigen Generation und den Nachkommen zusammen. Die Anzahl der Lösungen einer Population ist begrenzt. Daher werden am Ende einer Generation Individuen mit einer schlechten Fitness gelöscht und durch die Nachkommen mit einer höheren Fitness ersetzt. Auf diese Weise ergeben sich im Verlauf der Generationen zunehmend bessere Platzierungsergebnisse. Im Nachfolgenden werden die drei zur Verfügung stehenden genetischen Operatoren näher beschrieben.

- *Crossover*: Der Crossover-Operator benötigt zwei Individuen aus einer Generation zur Erstellung der Individuen der nachfolgenden Generation. Es wird ein zufällig ausgewählter Schnitt an beiden Elternindividuen durchgeführt. Die linke Hälfte des einen Elternteils wird mit der rechten Hälfte des anderen Elternteils kombiniert und umgekehrt. Auf diese Weise ergeben sich zwei neue Lösungen.
- *Mutation*: Die Nachkommen werden durch eine zufallsbasierte Vertauschung von Genen innerhalb eines Individuums erstellt. Die Mutation wird durch das Verhältnis der Anzahl der Vertauschungen zur Anzahl der Gene, der sogenannten Mutationsrate, gesteuert. Es wird damit der Prozentsatz der Population spezifiziert, die der Mutation unterzogen werden. Die Mutation sollte nur kleine Änderungen hervorrufen, in der Summe jedoch so viel, dass die Individuen über der Laufzeit des Algorithmus die gesamte Wertelandschaft abdecken. Am Anfang des Algorithmus ist es sinnvoller eine größere Änderung zu zulassen. Zum Ende des Verfahrens hin sollten nur noch kleine Änderungen erlaubt werden, um diejenigen Nachkommen, die sich nahe am Optimum befinden, dort zu belassen. Eine globale Mutationsrate von null wird sehr schlechte Ergebnisse abliefern, da einmal durch die Kreuzungsoperation aus der Population entfallene Individuen nicht wieder in die Population zurückkehren können und somit zum Auffinden der optimalen Lösung fehlen. Wenn die Mutationsrate zu hoch gesetzt wird, besteht die Gefahr, dass Individuen nahe beim Optimum von diesem weggedrängt werden und das Verfahren nicht konvergiert.
- *Selektion*: Mit Hilfe des genetischen Operators Selektion werden Individuen für die Reproduktion, d.h. für die nächste Generation, ausgewählt. Die zu berücksichtigende Nebenbedingung besteht darin, dass die Populationsgröße konstant bleiben muss. Je höher die Fitness eines Individuums ausfällt, desto größer ist die Wahrscheinlichkeit, dass es in die Nachfolgegeneration übernommen wird.

In Algorithmus 4-3 ist der Ablauf einer Platzierung unter Verwendung der genetischen Algorithmen in Pseudocode zusammenfassend dargestellt.

Der Ansatz die möglichen Lösungen einer Population, bestehend aus der aktuellen Generation und den erzeugten Nachkommen für den nächsten Iterationsschritt, auf Basis des Zufallsprinzips auszuwählen, bringt Vor- und Nachteile mit sich. Auf der einen Seite besteht durch das Zulassen von schlechteren Lösungen die Möglichkeit lokale Optima zu überwinden. Andererseits kann der Vorgang dazu führen, dass die Fitness der gesamten Population fällt, d.h. es besteht die Gefahr in einem lokalen Optimum hängen zu bleiben.

Bezogen auf die Platzierung repräsentiert jedes einzelne Symbol in der Regel die Koordinaten und den Namen einer Zelle. Aus guten Lösungen können unter Zuhilfenahme komplexer Operatoren bessere Lösungen erzeugt werden. Bei den genetischen Algorithmen können mehrere Lösungen gespeichert und verarbeitet werden, wodurch die besseren Lösungen weiterhin erhalten bleiben. Dadurch steigt jedoch der Speicherplatzbedarf im Vergleich zum Simulated-Annealing-Verfahren deutlich an. Weitere Ausführungen können dem Artikel von [Shah91] entnommen werden. Eine Anwendung der genetischen Algorithmen zur Lösung des Platzierungsproblems ist in [Chan91, Coho87, Coho03, Esbe92, Ebse97, Mura96, Sadi95] zu finden.

Algorithmus 4-3: Ablauf eines genetischen Algorithmus in Pseudocode

```
Generierung einer zufälligen Ausgangspopulation bestehend aus n Individuen,
jedes Individuum repräsentiert eine mögliche Platzierung;
berechne Fitness für jedes Individuum der Ausgangspopulation;
solange Abbruchkriterium nicht erreicht
   solange Anzahl Individuen der neuen Generation < n
      Selektion von zwei Elternindividuen;
      r:=Random[0,1];
      wenn r<P_crossover dann
         führe Crossover-Operator auf ausgewählten Individuen aus;
      wenn_ende
      r:=Random[0,1];
      wenn r<P_mutation dann
         führe Mutations-Operator auf ausgewählten Individuen aus;
      wenn_ende
   solange_ende
   ersetze alte Population mit der neuen Generation;
   berechne Fitness für jedes Individuum der neuen Population;
solange_ende
```

Da die genetischen Algorithmen nur langsam konvergieren, sind in [Zhan03, Zhan06b] Ansätze zur Kombination des genetischen Verfahrens mit dem Simulated-Annealing-Ansatz präsentiert worden. Das Simulated-Annealing-Verfahren bietet den Vorteil einer schnellen Konvergenz zu einem lokalen Optimum. Es besteht jedoch die Gefahr der Stagnation in diesem lokalen Bereich. Die Verbindung beider Verfahren ermöglicht ein gegenseitiges Aufheben der jeweiligen Nachteile.

4.1.5 Sonstige Ansätze zur Platzierung integrierter Schaltungen

Ein Ansatz zur Platzierung analoger Schaltungen mit Hilfe des Branch-and-Bound-Verfahrens ist in [Onod92] präsentiert worden. Die Methode basiert auf einer Gliederung des Lösungsraumes unter Verwendung eines Entscheidungsbaums. Für den Aufbau des Baumes werden die Orientierungen der Zellen und deren topologische Beziehungen untereinander, wie z.B. der minimale bzw. exakte Abstand, betrachtet. Um die dabei auftretende große Anzahl an möglichen Platzierungskombinationen zu begrenzen, werden zusätzliche Randbedingungen in Form der Netzlängenabschätzung und Einschränkungen der topologischen Beziehungen eingesetzt. Bei dem resultierenden Layout können Überschneidungen auftreten. In einer getrennten Phase werden verschiedene Formen der Zellen betrachtet, um die Fläche zu minimieren.

In [Mehr91] ist ein Werkzeug vorgestellt worden, welches die initiale Anordnung der Zellen aus dem Schaltbild ableitet. Die auftretenden Überlappungen zwischen den Zellen werden durch eine Form der Relaxation aufgelöst. Der Ansatz weist einige Einschränkungen auf. So können die Zellen nicht gedreht werden und es werden keine Layoutvarianten für eine Zelle in Betracht gezogen.

Neben dem konventionellen Ablauf einer Layouterzeugung, gegliedert in Synthese, Platzierung, Verdrahtung und Postlayout-Optimierung, existieren noch Ansätze, bei denen zwei oder drei Entwurfsschritte vereinigt werden. Dazu wird in [Lou97, Sale98] die Schaltung im ersten Schritt in baumartige Schaltungsteile gegliedert. Ohne Kenntnisse über Platzierung und Verdrahtung wird ein Technology Mapping, d.h. eine Abbildung der Schaltung auf eine Zielbibliothek, durchgeführt. Zu jedem Schaltungsteil werden verschiedene Gatterflächen und abgeschätzte Verdrahtungsflächen untersucht. Die Bewertung der Mapping-Qualität erfolgt erst nach der Platzierung mit Hilfe einer Kostenfunktion, die die aus der Mappingwahl resultierende Platzierung bewertet. Deshalb werden in diesem Ansatz flächenungünstigere Mapping-Lösungen bis zur Platzierung gespeichert. Die Auswahl

einer Anordnung aus Schaltungs- und Verdrahtungsfläche erfolgt erst bei der Platzierung. Im Anschluss an die Platzierung erfolgt die Detailverdrahtung.

Der in [Sten97] präsentierte Ansatz geht von einer initialen Platzierung aus, die mit einem beliebigen Werkzeug zu erstellen ist. Anschließend erfolgt eine Optimierung des Ausgangslayouts, indem iterativ gleichzeitig Netzlistenänderungen und Platzierungsveränderungen durchgeführt werden. Dazu wird in jedem Iterationsschritt der längste Pfad der Schaltung bestimmt. Durch Ändern der Netzliste wird versucht, eine Verkürzung der Pfadlänge zu erzielen, wobei Zellen entfernt bzw. hinzugefügt werden. Auftretende Überlappungen werden nach einer vorgegebenen Anzahl an Netzlistenänderungen durch einen Legalisierungsschritt aufgelöst.

4.2 Verdrahtungsverfahren

Die Verdrahtung bestimmt die genaue Geometrie der Verbindungswege zwischen den Anschlusspunkten platzierter Zellen. Der Erfolg und die Qualität der Verdrahtung sind im Wesentlichen abhängig von der gewählten Platzierung. Infolge der Komplexität des gemeinsamen Platzierungs- und Verdrahtungsproblems entzieht es sich einer optimalen algorithmischen Lösung. Bereits in der allgemeinen Formulierung ist das Verdrahtungsproblem NP-vollständig [Kram82].

Neben der Aufgabe den exakten Verlauf aller Verbindungsleitungen sowie die Zuordnung aller Leitungssegmente zu den zulässigen Verdrahtungsebenen durchzuführen, gibt es gewisse Randbedingungen zu beachten. Dazu zählen beispielsweise die Einhaltung sämtlicher Design-Regeln, eine Minimierung der Gesamtverdrahtungsfläche und -länge sowie die Reduktion der Häufigkeit von Leitungsknicken. Insbesondere bei analogen Schaltungen können weitere Forderungen, wie beispielsweise die Minimierung von Streukapazitäten oder spezielle Abschirmungen von Signalleitungen durch den Einsatz von Schilden, auftreten.

Neben der Wegsuche hat die Reihenfolge der zu verbindenden Netze einen großen Einfluss auf die Verdrahtbarkeit einer Schaltung. Eine schlecht gewählte Reihenfolge der Netze kann dazu führen, dass die Verdrahtbarkeit der Schaltung nicht mehr gewährleistet ist. Aus diesem Grund gibt es Ordnungsstrategien für die Reihenfolge der Netze. Häufig erfolgt die Sortierung nach der Funktionen der jeweiligen Netze. Innerhalb einer Gruppierung kann ebenfalls eine Ordnung festgelegt werden, wie beispielsweise die Länge der Leitungen. Es ist jedoch nicht möglich, eine allgemeine Ordnungsstrategie zu erstellen, die stets die Verdrahtbarkeit und die Einhaltung der Randbedingungen sichern kann.

Ausgehend von einer vorangegangenen Platzierung wird der Verdrahtungsprozess in der Regel in zwei Teilprobleme aufgeteilt, nämlich der globalen und der lokalen Verdrahtung. Die meisten Ansätze zur Verdrahtung analoger Schaltungen basieren auf lokalen Verdrahtungsalgorithmen, welche ursprünglich für digitale Systeme entworfen worden sind. Ein Überblick und vertiefende Diskussionen zu den jeweiligen Verfahren, die mehrheitlich einen Bezug zur Verdrahtung digitaler Schaltungen aufweisen, können [Breu72, Korn82, Laut87, Leng90, Lips83, Marw93, Moor59] entnommen werden. Ein Bezug zum analogen Layout wird in [Adle01, Garr88, Garr91, Mala90, Meye93, Schr05] hergestellt.

4.2.1 Globale Verdrahtung

Die Aufgabe der globalen Verdrahtung besteht in der Aufteilung der Layoutfläche in Hindernisse

und Verdrahtungsflächen, die den Restriktionen des verwendeten Verdrahtungsverfahrensgenügen. Üblicherweise erfolgt die Aufteilung in rechteckige Flächen, sogenannten Verdrahtungskanälen. Anschließend werden die zu verbindenden Netze den Teilflächen zugewiesen. Falls es bei Kanälen mit einer fest vorgegebenen Breite zu einem Überlauf kommen sollte, wird der betroffenen Verbindung ein anderer Kanal zugewiesen. Entsprechend erfolgt bei flexiblen Kanalbreiten eine Anpassung der Verdrahtungsfläche oder eine Modifikation der Platzierung. Ebenso können bei der globalen Verdrahtung analoge Randbedingungen, wie beispielsweise eine räumliche Trennung zwischen rauschempfindlichen und stark rauschenden Netzen, mit berücksichtigt werden.

Insgesamt verfolgt die globale Verdrahtung im Wesentlichen die Ziele eine Verdrahtungsfläche derart zu bestimmen, dass eine möglichst geringe Gesamtnetzlänge für jedes einzelne Netz realisiert wird. Ferner sollen damit ausweichende Pfade für häufige Netzwege ermittelt und eine Festlegung kleinerer Instanzen zur lokalen Verdrahtung durchgeführt werden.

Die Implementierung des globalen Verdrahtungsverfahrens kann durch einen Kanalschnittgraphen erfolgen. Die Kanten dieses Graphen beschreiben die Nachbarschaft zwischen den vorhandenen Kanälen. Die Knoten entsprechen den Schnittbereichen zwischen den Verdrahtungskanälen. Zusätzlich wird jeder Kante ein Gewicht zugeordnet, welches die Kapazität des entsprechenden Kanals widerspiegelt. Eine Zuweisung der Verdrahtungsflächen zu den Netzen erfolgt mit einem Algorithmus zur Bestimmung des kürzesten Pfades, wie z.B. nach Dijkstra [Corm01] oder mit Hilfe einer Steinerbaum-Heuristik [Chia90], welche auf den Kanalschnittgraphen angewandt werden.

Ein anderes Modell zur Globalverdrahtung basiert auf einer Kombination aus kräftegesteuerter Platzierung und Labyrinth-Verdrahtung [Mo01]. Die gesamte Layoutfläche wird hierbei zunächst in ein grobes Verdrahtungsraster eingeteilt. Jeder Rasterpunkt entspricht einem Kanal bzw. einem Teil des Kanals und erhält eine Kapazität zugewiesen aus der die maximale Anzahl an Pfaden innerhalb des Kanals hervorgeht. Unter Verwendung der kräftegesteuerten Platzierung werden zusätzliche Leitungspunkte erstellt und entsprechend positioniert. Durch diese Punkte verläuft zu einem späteren Zeitpunkt die globale Verdrahtung, die mit Hilfe eines Labyrinth-Verdrahters ausgeführt wird. Ein Leitungspunkt wird erstellt, wenn die Distanz zwischen zwei vorhandenen Rasterpunkten einen vorgegebenen Grenzwert überschreitet, die betroffenen Punkte sich nicht auf einer gemeinsamen orthogonalen Verbindungslinie befinden oder sich eine größere Anzahl an überbelegten Bereichen zwischen den beiden Punkten befindet. Umgekehrt werden zwei Punkte zu einem vereinigt, wenn die Distanz ein definiertes Limit unterschreitet. Nach der Generierung der Leitungspunkte erfolgen eine Berechnung der Kräfte zwischen den Punkten und eine Neupositionierung, um einen Gleichgewichtszustand der Kräfte zu erlangen. Der Vorgang der Generation und Vereinigung der Leitungspunkte, sowie der anschließenden Translation, wird solange wiederholt, bis ein annäherndes Gleichgewicht der Kräfte erzielt werden kann. Im Anschluss wird die eigentliche Globalverdrahtung mit einem Labyrinth-Verdrahter, basierend auf den resultierenden Leitungspunkten, durchgeführt.

Ein weiterer Lösungsansatz zur Globalverdrahtung ist durch die Anwendung eines Verdrahtungsmusters gegeben [Kast00]. Innerhalb dieses Verfahrens wird die Layoutfläche in Kacheln aufgeteilt. Alle Netze, die mehr als zwei Anschlüsse aufweisen, werden mittels einer Steiner-Heuristik zerlegt. Jede Kante einer Kachel verfügt über eine konstante Kapazität an Leitungen. Als Verdrahtungsmuster stehen lediglich L- und Z-Formen zur Verfügung, die ausgehend vom Start- und Zielpunkt bei der Durchführung der Verdrahtung zu verwenden sind. Die Wahl der Verdrahtungsform

ist abhängig von der lokalen Verdrahtungsdichte und der Leitungslänge. Eine Einflussnahme durch den Benutzer ist in Form einer Gewichtung gegeben.

Ein Vorteil dieser Vorgehensweise der Verdrahtung ist die Vorhersagbarkeit der Verbindungslängen. Nachteilig wirkt sich die Beschränkung der Verdrahtungsmuster auf diejenigen Netze aus, die nicht direkt mittels der vorgegebenen Formen verdrahtet werden können.

4.2.2 Lokale Verdrahtung

Im Rahmen der lokalen Verdrahtung wird die exakte geometrische Form der Verbindungsleitungen festgelegt. In Analogie zur Wegsuche in einem Labyrinth sind bei der Detailverdrahtung Pfade für die jeweiligen Verbindungen zu ermitteln. Dabei sind neben der Einhaltung der Entwurfsregeln weitere Randbedingungen zu beachten. Das Hauptmaß für die Bewertung der Verdrahtung ist insbesondere bei digitalen Schaltungen die Länge der Leitungen. So führen längere Verbindungswege einerseits zu größeren parasitären Effekten, wie beispielsweise den Leitungslaufzeiten, andererseits vergrößert sich der Platzbedarf für die Verdrahtung. Bei analogen Schaltungen liegt das Augenmerk weiter auf einer Vermeidung bzw. Minimierung der kapazitiven Einkopplungen für Signalleitungen. Die Abschätzung der Leitungseigenschaften sollte sehr einfach sein, da aufwendige Verfahren den Verdrahtungsvorgang stark verlangsamen würden.

Die lokale Verdrahtung wird häufig in allgemeine Verdrahter und eingeschränkte Verdrahter unterteilt. Die allgemeinen Verdrahter können Verbindungen zwischen beliebig angeordneten Punkten erstellen. Demgegenüber sind eingeschränkte Verdrahter lediglich in der Lage spezielle Anordnungen zu behandeln.

4.2.2.1 Rasterbasierte Verdrahter

Der rasterbasierte Punkt-zu-Punkt-Verdrahter ist ein Wellenfrontverdrahter, der einen Weg zwischen Start-und Zielpunkt eines Netzes unter Verwendung eines gerasterten Verdrahtungsmodells sucht. Das Verfahren basiert in den meisten Fällen auf dem Algorithmus von Moore [Moor59] und wird auch Labyrinthverdrahtung genannt. Eine anschauliche Vorstellung liefert eine Welle, die sich ausgehend vom Startpunkt der Verdrahtung in alle Richtungen ausbreitet. Die Welle läuft an Hindernissen im Layout vorbei und erreicht den Zielpunkt, insofern eine mögliche Verbindung existiert. Auf diese Weise kann nicht nur geprüft werden, ob eine Lösung vorliegt, sondern auch die Distanz zwischen dem Start- und dem Zielpunkt berechnet werden. Der eigentliche Weg ist bis zu diesem Zeitpunkt noch unbekannt. Das Auffinden des Pfades erfolgt mit Hilfe eines Markierungsverfahrens.

Algorithmus 4-4: Rasterbasierte Verdrahtung (Lee Algorithmus)

```
jeden Startpunkt einer Verbindung mit 0 markieren;
für jedes bereits besuchte Rasterelement
   markiere jedes direkte Nachbarelement, sofern es keinen gesperrten Bereich
   repräsentiert, mit i+1, wenn das betrachtete Element den Wert i besitzt;
für_ende
beginnend beim Zielpunkt markiere einen zusammenhängenden Weg von markierten
Rasterelementen mit absteigenden aufeinanderfolgenden Markierungswerten;
markierter Weg führt zurück zum Startpunkt und stellt eine zulässige Verbindung
dar;
```

Für die Verdrahtung wird die Verdrahtungsfläche auf ein Raster abgebildet und in zulässige und gesperrte Bereiche unterteilt. Die Breite eines Rasterelements ist so zu wählen, dass zwei benachbarte Rasterpunkte verdrahtet werden können, ohne die Entwurfsregeln zu verletzen.

Grundsätzlich kann dafür die Summe aus der Leitungsbreite und dem Mindestabstand zwischen zwei Leiterbahnen verwendet werden [Adle01]. Ausgehend vom Startpunkt, der mit dem Wert 0 initialisiert wird, erfolgt ein Besuch der direkt benachbarten Rasterelemente. Befindet sich der Zielpunkt unter den besuchten Elementen, ist ein Weg gefunden und die Suche kann beendet werden. Andernfalls werden alle Nachbarelemente mit einem um 1 erhöhten Wert markiert und in eine Warteschlange eingereiht. Anschließend werden die Nachbarn dieser Punkte solange rekursiv untersucht, bis der Zielpunkt erreicht wird oder die Warteschlange leer ist. In diesem Fall existiert keine Verbindung zwischen dem Start- und dem Zielpunkt. Wenn der Zielpunkt erreicht worden ist, kann der Weg vom Ziel- zum Startpunkt mit Hilfe der markierten Rasterelemente zurückverfolgt werden. Es ergibt sich damit der in Algorithmus 4-4 aufgezeigte Ablauf für die rasterbasierte Verdrahtung. Das Verfahren ist auch als Lee-Algorithmus bekannt [Lee61].

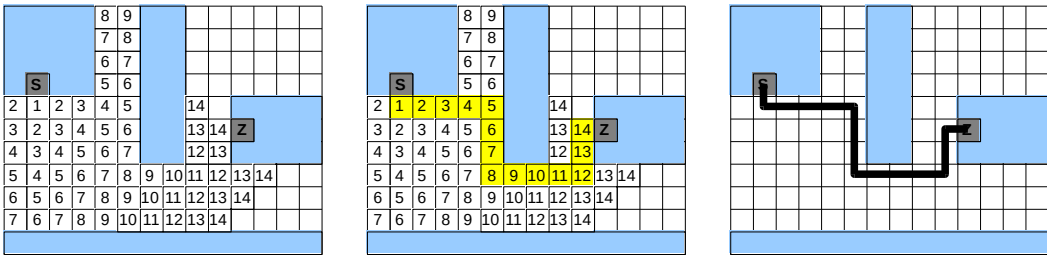


Abbildung 4-1: Beispiel für eine rasterbasierte Verdrahtung

Die Abbildung 4-1 zeigt ein Beispiel für die rasterbasierte Verdrahtung eines Zweipunktnetzes mit dem Lee-Algorithmus.

Verschiedene Verfahren können zur Bestimmung des nächsten Nachbarelements verwendet werden. Das bekannteste rasterbasierte Verdrahtungsverfahren, der Lee-Algorithmus [Lee61], setzt auf eine symmetrische Breitensuche. Im Gegensatz dazu wird beim Hadlock-Algorithmus [Hadl77] zur Suche nach dem nächsten benachbarten Rasterelement die A*-Heuristik [Hart68] eingesetzt. Eine Kombination von Lee-Algorithmus und einem Liniensuchverfahren ist durch den Soukup-Algorithmus [Souk92] gegeben. In diesem Fall kommt eine zielgerichtete Tiefensuche zum Einsatz. Das Verfahren erstellt zunächst eine Linie von Startpunkt in Richtung des Zielpunkts. Wenn das Ziel nicht erreicht werden kann, wird um die erstellte Line mit Hilfe des Lee-Algorithmus ein Rasterelement gesucht, welches den Abstand zum Ziel verringert. Ausgehend vom ermittelten Punkt wird eine neue Linie in Richtung Zielpunkt generiert. Das Verfahren wird solange wiederholt, bis der Zielpunkt erreicht ist.

Der entscheidende Vorteil der rasterbasierten Verdrahter liegt in der Eigenschaft, dass diese immer eine Verbindung ermitteln können, falls eine existiert. Die berechnete Verdrahtung ist dabei in jedem Fall die kürzeste mögliche Verbindung. Der Aufwand für eine Implementation ist verhältnismäßig gering. Den genannten Vorteilen steht dennoch eine Reihe an Nachteilen gegenüber. In jedem Iterationsschritt wird jeweils nur eine Verbindung betrachtet. Das führt zu einer relativ hohen Laufzeit des Algorithmus, infolge des wiederholt auszuführenden Markierungsverfahrens. Die Netzverbindungen werden unabhängig voneinander berechnet. Dadurch neigt das Verfahren dazu, mit früh angelegten Verbindungen Wege für spätere Verbindungen zu versperren. Neben der Abhängigkeit von der Reihenfolge der zu verdrahtenden Netze führt die Abbildung auf ein Raster, bei dem die Freiflächen ebenfalls gerastert werden, zu einem nicht unerheblichen Speicherplatzbedarf. So steigt die Anzahl der Rasterelemente quadratisch mit der Kantenlänge der zu untersuchenden Schaltung an. Infolgedessen können äußerst umfangreiche Schaltungen nicht mehr mit den Wellenfront-Verdrahtern verdrahtet werden.

4.2.2.2 Linienverdrahter

Die linienbasierten Verdrahter suchen ähnlich wie die rasterbasierten Verdrahter einen Weg vom Start zum Ziel. Der Unterschied zwischen den beiden Verfahren liegt in der Vorgehensweise. Die Linienverdrahter verfolgen eine globale Verdrahtungsstrategie und verzichten auf eine Rasterung der Layoutfläche. Die Ausgangsituation ist die eingeteilte Layoutfläche mit Freiflächen und zu umgehenden Hindernissen. Die Teillösungen werden durch das Hinzufügen von Liniensegmenten erweitert. Diese Liniensegmente werden als Versuchslinien bezeichnet. Das Verfahren startet mit der Erzeugung von horizontalen und vertikalen Versuchslinien im Start- und Zielpunkt. Die Linien werden solange ausgedehnt, bis die Grenze der Verdrahtungsfläche oder ein Hindernis erreicht wird. Falls sich die Versuchslinien schneiden, so ist ein Pfad gefunden und die Suche kann beendet werden. Andernfalls werden orthogonale Linien zu den vorhergehenden Versuchslinien erzeugt. Der Vorgang wird iterativ solange wiederholt, bis sich der vom Startpunkt ausgehende Linienzug mit dem von Zielpunkt schneidet. Der Ablauf der linienbasierten Verdrahtung ist in Algorithmus 4-5 zusammengefasst.

Algorithmus 4-5: Linienbasierte Verdrahtung

```
erstelle vom Start- und Zielpunkt der zu verdrahtenden Verbindung ausgehend
jeweils eine Linie, die am Rand des Layouts oder an einem Hindernis endet;
solange keine Verbindung gefunden ist
   wenn die beiden Linien sich schneiden dann
      eine Verbindung ist gefunden und die Schleife kann beendet werden;
   sonst
      erstelle auf jeder der beiden Linien Fluchtlinien senkrecht zu den
      zuvor erzeugten Linien;
   wenn_ende
solange_ende
Folge von Liniensegmenten und Fluchtpunkten repräsentiert eine Verbindung vom
Start- zum Zielpunkt;
```

Generell unterscheidet man zwei verschiedene Ansätze zur linienbasierten Verdrahtung. Der Algorithmus von Hightower [High69] erstellt die Versuchslinien anhand von charakteristischen Punkten der zu umgehenden Hindernisse. Es wird damit in jedem Iterationsschritt nur eine horizontale oder vertikale Line heuristisch erzeugt. Diese Vorgehensweise ist günstig in Bezug auf den Speicherbedarf und Laufzeit. Es ist jedoch möglich, dass nicht immer eine existierende Lösung ermittelt werden kann. Demgegenüber werden beim Verfahren nach Mikami [Mika68] die Fluchtpunkte der Versuchslinien nach einem festen Raster, das sich nicht nach der Umgebung richtet, generiert. Zudem besteht durch die größere Anzahl an Versuchslinien eine größere Wahrscheinlichkeit eine Lösung zu finden. Die Verwendung eines Rasters bedingt einen erhöhten Speicherplatz- und Laufzeitbedarf.

Die allgemeine Vorgehensweise eines Linienverdrahters ist der Abbildung 4-2 (a) zu entnehmen. Die Auswahl des nächsten Fluchtpunktes der beiden genannten Algorithmen ist beispielhaft in Abbildung 4-2 (b) gezeigt.

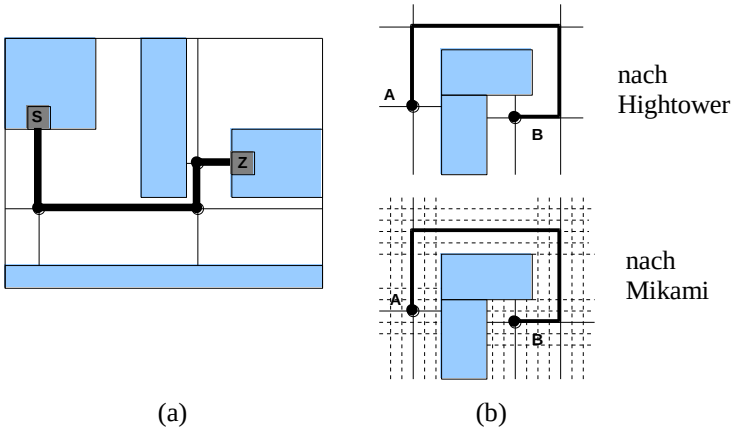


Abbildung 4-2: (a) Beispiel für ein linienbasiertes Verfahren, (b) Wahl des Fluchtpunktes

Die Linienverdrahter zeichnen sich im Vergleich mit den Labyrinthverdrahtern durch einen geringeren Speicherbedarf und eine kürzere Laufzeit aus. Das linienbasierte Verfahren ist besonders gut geeignet für Verdrahtungen mit relativ wenigen aber großen Hindernissen auf der Verdrahtungsfläche. In diesem Fall ist der Geschwindigkeitsgewinn gegenüber den rasterbasierten Verfahren besonders groß [Malo04]. Der Implementierungsaufwand für Algorithmen ist vergleichbar mit dem der rasterbasierten Verfahren. Es wird ferner lediglich eine Verbindung zurzeit untersucht. Dieser Sachverhalt und die prinzipbedingte geringe Anzahl an Richtungswechseln können sich nachteilig auf die Verdrahtbarkeit auswirken. So neigen die Verfahren dazu, mit früh angelegten Verbindungen, Wege für spätere Verdrahtungen zu versperren. Insbesondere das Hightower-Verfahren kann nicht immer garantieren, einen existierenden Pfad zu finden. Ebenso ist ein ermittelter Pfad ist nicht zwingender Weise die kürzeste Verbindung. Demgegenüber findet das Verfahren von Mikami eine Lösung, wenn diese vorhanden ist. Es handelt sich dabei auch um eine recht kurze Verbindung, jedoch erfolgt die Suche zu Lasten vieler Fluchtpunkte.

4.2.2.3 Kanalverdrahter

Die Kanalverdrahter werden üblicherweise beim automatisierten Entwurf mit Standardzellen bzw. Gate-Arrays eingesetzt. Das Verfahren basiert auf einer Einschränkung der Geometrie. Die Verdrahtungsflächen bestehen aus Kanälen. Dabei handelt es sich um ein Rechteck, welches an zwei sich gegenüberliegenden Seiten die Anschlüsse besitzt. Der Kanal verfügt obendrein über feste oder bedingt variable Bereiche, den Spuren, in denen die Verbindungsleitungen verlaufen sollen. Dabei werden die vertikalen Liniensegmente als Branch und die horizontalen Verbindungen als Trunk bezeichnet. Die Verdrahtung erfolgt jeweils auf eigenen Metallebenen. Die Kanalkapazität ist zurückzuführen auf die Anzahl der Spuren und der Anzahl der Metalllagen, die Trunks aufnehmen können.

4.2.2.3.1 Left-Edge-Verdrahter

Die Durchführung der Verdrahtung erfolgt unter Verwendung des Left-Edge-Verfahrens [Hash71]. Die Voraussetzung für die Anwendung des Algorithmus besteht darin, dass jedes zu verdrahtende Netz lediglich ein horizontales Liniensegment aufweist. Die Trunks beginnen und enden an den Positionen der korrespondierenden Anschlüsse. Sie werden nach ihren linken Anfangspositionen (left edge) aufsteigend in eine Liste einsortiert. Zu Beginn des Algorithmus wird das erste Liniensegment aus der Liste entfernt und der ersten Spur zugewiesen. Die Liste mit den verbleibenden Trunks wird sequentiell mit dem Endpunkt des aktuellen Segments aus der Spur verglichen. Wenn sich ein Anfangspunkt rechtsseits vom Endpunkt befindet, wir dieser Trunk aus der Liste entfernt und in der

Spur hinter dem vorherigen angeordnet. Der Vorgang wird solange wiederholt, bis die Spur gefüllt oder die Liste leer ist. Falls in der aktuellen Spur keine weiteren Trunks eingefügt werden können, wird mit dem ersten verbliebenen Trunk eine neue Spur angelegt und das Verfahren beginnt von vorn.

4.2.2.3.2 Greedy-Kanalverdrahter

Der Greedy-Kanalverdrahter stellt ein alternatives Verfahren zum Left-Edge-Algorithmus dar. Es erlaubt eine beliebige Anzahl horizontaler Segmente. Die Greedy-Kanalverdrahtung nach Rivest und Fiduccia [Rive82] geht schrittweise im Kanal vor. Es legt die jeweils möglichen Verbindungen bzw. führt die notwendigen Vorbereitungen durch. Dabei werden vom Algorithmus bei Bedarf dynamisch Spuren oder Doglegs angelegt. Bei den Doglegs handelt es sich um einen Knick, der bei einem genügend großen Spaltenabstand erstellt werden kann. Auf diese Weise kann eine Konfliktsituation mit einer anderen Leitung gelöst werden. Der Vorteil des Verfahrens liegt in einer besseren Kanalausnutzung im Vergleich mit dem Left-Edge-Verfahren. Nachteilig ist, dass eine einmal getroffene Entscheidung nicht mehr zurückgenommen werden kann. Dadurch können Konflikte auftreten, die durch eine Schrittrücknahme gelöst werden könnten.

4.2.2.3.3 Switchbox-Verdrahter

Rechtecke, die mehrere Kanäle miteinander verbinden, werden als Switchboxen bezeichnet. Innerhalb der Switchbox soll mit Hilfe der Switchbox-Verdrahter eine gültige Verdrahtung gefunden werden, wobei die Anzahl der Spuren und Spalten unveränderlich ist. Maximal vier Kanäle können auf diese Weise miteinander verbunden werden. Die Problemlösung der Switchbox-Verdrahtung ist aufwendiger als die der allgemeinen Kanalverdrahtung. Es werden oft verschiedene Heuristiken angewandt, um entsprechende Verdrahtungsvarianten auszuführen. In [Luk85] wird einer der am häufigsten verwendeten Algorithmen für Switchbox-Verdrahter vorgestellt. Das Verfahren stellt eine Weiterentwicklung der Greedy-Kanalverdrahtung dar [Rive82]. Die wesentliche Änderung besteht darin, dass der Algorithmus nicht nur in vertikaler, sondern auch in horizontaler Richtung ausgeführt wird.

4.2.2.3.4 Fluss-Verdrahter

Eine Flussverdrahtung kann nur unter Einhaltung bestimmter Randbedingungen ausgeführt werden. Es ist damit ein spezielles Verdrahtungsverfahren. Alle zu verbindenden Netze müssen 2-Punkt-Verbindungen sein. Die jeweiligen Anschlüsse müssen sich auf den gegenüberliegenden Seiten des Kanals befinden. Die Reihenfolge der Anschlüsse muss auf beiden Seiten identisch sein, d.h. es darf zu keiner Kreuzung von Verbindungen kommen. Die Aufgabe des Verfahrens besteht lediglich darin, die Kanalhöhe zu minimieren. Die Verbindungen werden mittels einer einzigen Maske der Leitungsebenen erstellt.

Obwohl es ein sehr restriktives Verfahren ist, tritt es bei digitalen Schaltungen sehr oft bei der Verdrahtung von standardisierten Zellstrukturen auf. Bei analogen Schaltungen findet der Fluss-Verdrahter beispielsweise Anwendung im Layoutgenerator DBE [Conw92].

4.2.3 Sonstige Ansätze zur Verdrahtung integrierter Schaltungen

Die prozedurale und vorlagenbasierte Verdrahtung stellt den einfachsten Weg, eine Verdrahtung zu erreichen, dar. Der Verlauf der Verbindungen wird mittels einer vollständigen Beschreibung durch den Designer vorgegeben. Diese Vorgehensweise ist eine geeignete Methode für die interne Verdrahtung von bekannten Strukturen, wie beispielsweise den Stromspiegeln, die mit Hilfe von

Modulgeneratoren erzeugt werden. Eine Anwendung dieser Verfahren ist in [Conw92, Onod90, Zhan01] zu finden.

Zur Verdrahtung kritischer Netze und der Platzierung der entsprechenden Zellen in einem Schritt wird bei dem Werkzeug ANAGRAM II [Cohn91] das Verfahren Simulated Annealing verwendet. Infolge einiger Einschränkungen und einer hohen Laufzeit ist dieser Ansatz lediglich für sehr kleine Schaltungen geeignet.

Eine rasterfreie Labyrinthverdrahtung ist durch den analogen Verdrahter GARA des Layoutpakets ALSYN [Meye93] implementiert worden. Es wird eine "Corner Stitching"-Datenstruktur benutzt, die erstmals durch Ousterhout [Oust84] eingeführt wurde. Bei der Suche nach den Wegpunkten der jeweiligen Pfade werden die Fluchtpunkte, die sich vor und hinter einer benachbarten Geometrie befinden, untersucht. Die Entwurfsregeln werden dynamisch bei der Wegsuche berücksichtigt. Die Repräsentation der tatsächlichen Layoutfläche garantiert, dass mögliche Wege gefunden werden.

Eine Erweiterung des Verdrahtungswerkzeugs GARA ist durch das Programm PARSY gegeben [Schr05]. Es wird hier eine neue Methode zur Verdrahtung von Leitungsbündeln unter Berücksichtigung entsprechender Symmetrien gegeben. Das Haupteinsatzgebiet ist die Erstellung von analogen Signalleitungen, kritischen digitalen Busleitungen und Taktleitungen.

4.3 Vergleich der vorgestellten Konzepte

Insgesamt zeigt sich, dass in den letzten Jahren die zweistufige Layouterzeugung eines Designs Standard ist. Die Layouterzeugung integrierter Schaltung verläuft überwiegend in zwei Schritten. Zuerst erfolgt eine geeignete Anordnung der Zellen mit Hilfe eines Platzierers. Die Verbindungen zwischen den Zellen werden anschließend durch einen Verdrahter erzeugt. Bei dieser Vorgehensweise werden unterschiedlichen Typen von Platzierern und Verdrahtern kombiniert. Es sind deutliche Fortschritte bei der Entwicklung der Verdrahtungsmodelle zu verzeichnen. Dennoch erfordert die Hintereinanderausführung stets eine Abschätzung der Folgeschritte. Die Abweichungen zwischen der Abschätzung der Netze während der Platzierung und der tatsächlichen Verdrahtung können derart signifikant werden, dass einige Schaltungen nicht unter Einhaltung aller Nebenbedingungen verbunden werden können [Malo04].

Zwei wichtige Aspekte, die nicht direkt mit den jeweiligen Algorithmen verbunden sind, werden bei der Platzierung benötigt. Es sind Methoden zum Abschätzen der Netzlänge, die in den Kostenfunktionen der Platzierungsverfahren mit einbezogen werden und die Verdrahtungsflächen, die sich um die Zellen befinden können. Diese Flächen können die Abstände zwischen den Zellen signifikant beeinflussen.

Für die Netzlängenabschätzung existieren diverse Modelle, die sich im Hinblick auf den Aufwand und der Genauigkeit der Vorhersage unterscheiden. In Abbildung 4-3 ist eine Auswahl der gängigen Verfahren zur vereinfachten Näherung der Netzlänge veranschaulicht. Die Anschlüsse werden als Rechtecke dargestellt, entgegen der üblicherweise verwendeten Darstellung in Form eines Punktes [Ship88]. Insbesondere bei analogen Schaltungen kann ein einzelnes Terminal die ganze Seite einer Zelle einnehmen bzw. die Zelle vollständig bedecken (z.B. bei Kapazitäten). Es sind auch häufiger Terminals zu finden, die sich aus einer Folge von Rechtecken zusammensetzen. Derartige Anschlüsse werden in der Regel durch ein minimal umschließendes Rechteck beschrieben.

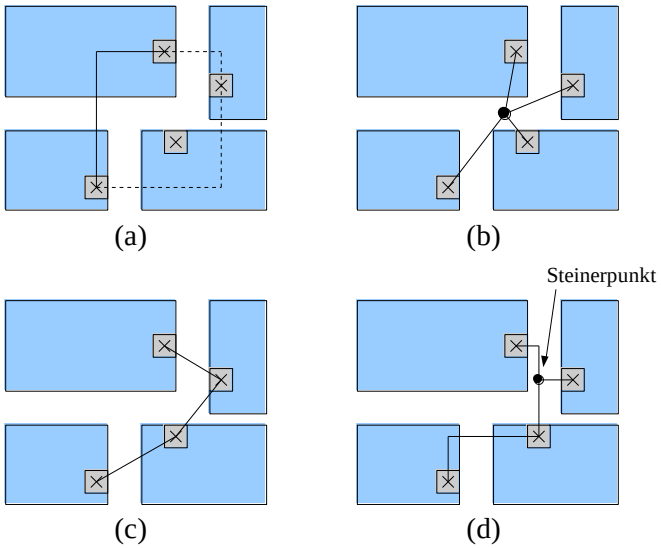


Abbildung 4-3: Netzlängenabschätzung mit Hilfe des halben Umfangs des umschließenden Rechtecks (Half Perimeter) (a), einer Schwerpunktberechnung (b), einem minimalen Spannbaum (c) und eines Steinerbaums (d).

Bei den Verfahren zur Netzlängenabschätzung handelt es sich um:

- Half Perimeter:

  Es stellt die einfachste Möglichkeit dar, benötigt wenig Rechenzeit und wird daher am häufigsten verwendet. Die Netzlänge zwischen den Anschlüssen wird durch den halben Umfang des minimal umschließenden Rechtecks, welches alle erforderlichen Terminals umfasst, näherungsweise berechnet. Diese Methode wird beispielsweise in [Koza84, Lovr91] verwendet.

- Schwerpunkt:

  Basierend auf der Summe der Abstände der Anschlüsse zum gewichteten Zentrum des Netzes erfolgt bei diesem Verfahren die Abschätzung der Netzlänge [Wong86].

- Minimaler Spannbaum:

  Zur Berechnung des minimalen Spannbaums ist es zunächst notwendig, den vollständigen Graphen der Verbindungen zwischen den Terminals zu erstellen. Jedes Terminal wird dabei durch einen Knoten des Graphen repräsentiert. Anschließend ist der kürzeste Teilgraph zu ermitteln, der alle Knoten beinhaltet. In [Char94, Cohn91] findet dieses Verfahren beispielsweise Anwendung.

- Steinerbaum

  Bedingt durch einen erheblichen Rechenaufwand handelt es sich beim Steinerbaum um das aufwändigste Verfahren zur Netzlängenabschätzung. Die Vorgehensweise ist vergleichbar mit der Erstellung eines minimalen Spannbaums. Der Unterschied besteht darin, dass neben den vorhandenen Knoten spezielle Verzweigungsknoten, sogenannte Steinerpunkte, eingefügt werden können. Damit beschreibt der Steinerbaum den kürzesten möglichen Pfad für ein Netz. Die Berechnung der Steinerpunkte hingegen ist ein NP-vollständiges Problem [Sarr92].

Die Realisierung der Verdrahtung analoger Schaltungen erfolgt überwiegend mit Hilfe verallgemeinernde Flächenverdrahter, basierend auf der Labyrinth- oder Linienverdrahtung. Die Qualität der resultierenden Leitungen ist stark abhängig von der vorausgehenden Platzierung. Im

Gegensatz zur Detailverdrahtung digitaler Schaltungen, erfolgt die Erstellung der Leitungen für analoge Schaltungen für ein Netz nach dem anderen. Dadurch tritt häufiger das Problem auf, dass bereits verlegte Leitungen aufgelöst und neu zu erstellenden sind.

5 Platzierung

Unter der Modellierung eines praktischen Problems versteht man die Extraktion der wesentlichen Eigenschaften und Zielsetzungen der Problemstellung und deren Abbildung auf eine mathematische Aufgabe. Eine solche Aufgabe lässt sich häufig durch Einführen von Zustandsvariablen und deren Verkettung ableiten. Bei der Modellierung ist es insbesondere wichtig, dass das praktische Problem zunächst präzise nachgebildet wird. Beim Versuch der Anwendung dieser Leitlinie auf das Platzierungsproblem ergeben sich einige Schwierigkeiten.

Eine Platzierung muss in allen Entwurfsarbeiten die Verdrahtbarkeit gewährleisten. Das zugehörige Entscheidungsproblem ist NP-vollständig. Aus diesem Grund wird bei den Platzierungsverfahren die Verdrahtbarkeitsbedingung dadurch ersetzt, dass in der Zielfunktion die Verdrahtungslänge berücksichtigt wird. Die Verdrahtungslänge wird abgeschätzt, da diese im Allgemeinen nicht in polynomialer Laufzeit berechenbar ist. Die Aufgabe, eine genaue Modellierung des allgemeinen Platzierungsproblems zu finden, wird zusätzlich dadurch erschwert, dass verschiedene Entwurfsverfahren mit unterschiedlichen Optimalitätskriterien existieren. Es ist eine Vielzahl an technischen Randbedingungen vorhanden, welche von Anwendung zu Anwendung stark differieren. So dürfen beispielsweise bei bestimmten Anwendungen Netze eine vorgegebene Länge nicht überschreiten. Umgekehrt dürfen andere Netze eine Mindestlänge nicht unterschreiten. Bedingungen dieser Art sind in einem angemessenen Rahmen bereits beim Platzieren der Zellen zu berücksichtigen.

Die Menge an Problemen bei der Modellierung lassen es zunächst schwer möglich erscheinen, eine geeignete Realisierung für das Platzierungsproblem zu finden. Nicht zuletzt aus diesem Grund existiert eine große Anzahl an Verfahren, welche zur Lösung des Platzierungsproblems entwickelt wurden. Diese sind jedoch zum Teil nur auf einen Entwurfsstil oder für eine spezielle Zielfunktion effizient anwendbar. Insbesondere können zusätzliche Nebenbedingungen bewirken, dass eine vollkommen neue Implementierung notwendig wird.

Im Rahmen dieser Arbeit ist ein Ansatz entwickelt worden, der eine annähernd simultane Platzierung und Verdrahtung integrierter, analoger Schaltungen erlaubt. Im nachfolgenden Abschnitt wird zunächst die grundlegende Vorgehensweise der sukzessiven Platzierung und Verdrahtung beschrieben. Der wesentliche Inhalt dieses Kapitels ist die Platzierung. Der Ablauf und die verwendeten Modelle für die Verdrahtung werden im nächsten Kapitel 6 beschrieben. Das der Platzierung und Verdrahtung zugrunde liegende Layoutmodell wird in Kap. 5.2 erläutert. Ein zentrales Element für das in dieser Arbeit präsentierte Verfahren stellt die Kompaktierung dar. Aus diesem Grund werden in Kap. 5.4 die grundlegenden Ansätze vorgestellt. Mit Hilfe der zu platzierenden Module und den Netzliste wird ein Verbindunsgraph erstellt (siehe Kap. 5.3). Auf der Basis des Verbindungsgraphen wird für die Module ein binärer Platzierungsbaum erzeugt (siehe Kap. 5.5), mit dem die Reihenfolge der Kompaktierungen festgelegt wird. Die Auswertung des Platzierungsbaums und der damit verbundene Aufbau der endgültigen Platzierung ist Bestandteil von Kap. 5.6. Den Abschluss des Kapitels bildet eine kurze Beschreibung der objektorientierten Implementierung des Platzierers.

5.1 Idee der sukzessiven Platzierung und Verdrahtung

Im Schaltplan-Editor von Cadence werden mit Hilfe des ALADIN-Pakets die Bauelemente der Schaltung gruppiert. Mittels der Modulgeneratoren werden die Layouts der Elementgruppen, die

Module, erstellt. Dabei können verschiedene Topologien für ein Modul erzeugt werden, die bei der Platzierung mit berücksichtigt werden sollen. Die Layoutdaten werden für jede Modultopologie in einer separaten Datei abgelegt. Die Beschreibung der geometrischen Elemente des jeweiligen Layouts erfolgt im SKILL-Format, der Interpretersprache von Cadence. Ebenso wird automatisch für jedes Layout der Module eine Liste der möglichen Anschlusspunkte erzeugt. Es können somit zu jedem Modul mehrere Layouttopologien vorgegeben werden. Während der Platzierung wird die voraussichtlich am besten geeignete Darstellung ausgewählt.

Neben den Daten der Layouts und Anschlüsse benötigt das Platzierungs- und Verdrahtungswerkzeug eine Konfigurationsdatei. Diese enthält einen Verweis auf die Layoutdaten, die zugehörigen Terminals und die Sensitivitätsmatrix. Mit dieser Matrix können die parasitären Kapazitäten während der Verdrahtung beeinflusst werden.

Jedes Netz ist durch den Schaltungsdesigner zu charakterisieren, d.h. neben der Angabe einer Priorität ist die Art der Leitung anzugeben. Es kann zwischen einer Eingabe-, Takt-, Signal- und Versorgungsleitung unterschieden werden. Auf diese Weise wird die Verdrahtung der Module entscheidend beeinflusst. Neben der Priorität und dem Leitungstyp ist die zu erwartende Stromdichte des Netzes zu spezifizieren, wodurch die Leitungsbreite definiert wird. Diese kann mit Hilfe einer DC-Simulation der Schaltung bestimmt werden.

In der Initialisierungsphase werden die Layoutdaten sequentiell eingelesen. Aus diesen Daten wird jeweils ein Kantenmodell erstellt, welches für die weiteren Prozess-Schritte verwendet wird. Diese Vorgehensweise führt zu einer Einsparung bezüglich der Laufzeitkomplexität und des Speicherbedarfs der einzelnen Module.

Unter Verwendung der Netzliste wird ein Graph erstellt, der die Beziehung zwischen den Modulen beschreibt. Die Knoten entsprechen den Modulen und die Kanten repräsentieren die Verbindungen zwischen den Modulen. Mit Hilfe des Verbindungsgraphen wird die Reihenfolge der zu platzierenden Module festgelegt. Dazu wird der Graph solange einer Vereinigung der Knoten unterzogen, bis lediglich ein Knoten zurückbleibt. Es werden jeweils zwei Knoten mit dem höchsten Verbindungsgewicht zu einem neuen Knoten vereinigt. Anschließend werden die Gewichte der Kanten des Verbindungsgraphen aktualisiert. Der Vorgang ist vergleichbar mit dem Zusammenschluss zweier Module zu einem neuen Modul. Damit ist die Verwendung des in [Wolf96, Wolf98a, Wolf98b, Wolf99b] vorgestellten hierarchischen Kompaktierers möglich. Es werden stets zwei Objekte mittels Kompaktierung zu einem neuen Objekt vereinigt. Die Einhaltung der technologiebedingten Restriktionen erfolgt während des Kompaktierungsvorgangs automatisch. Für die Module bleibt vor der Kompaktierung noch zu bestimmen, welche Topologie jeweils zu verwenden ist, falls mehrere zur Auswahl stehen. Es ist ebenso die Möglichkeit einer Transformation der Module zu berücksichtigen sowie die Ausrichtung zueinander, welche orthogonal zur Kompaktierungsrichtung erfolgt.

Die Auswahl der aufgeführten Parameter erfolgt mittels eines Entscheidungsbaums. Mit dessen Hilfe werden die Folgen der Parameterwahl auf die nachfolgend zu platzierenden Module untersucht. Im Entscheidungsbaum werden die möglichen Topologien, Kompaktierungsrichtungen und Transformationen der Module berücksichtigt. Die Wurzel repräsentiert das Zielobjekt, die direkten Kindknoten die verschiedenen Variationen für das zu kompaktierende Objekt. Die Vorgehensweise wird für die weiteren Module wiederholt, wobei das zu kompaktierende Objekt den Elternknoten bildet und die verschiedenen Möglichkeiten des nächsten Moduls die entsprechenden Nachfolger-Knoten. Auf diese Weise entstehen diverse Kompaktierungssequenzen. Jede dieser Folgen wird auf

einen Schnittbaum abgebildet. Die Schnittlinien werden durch die dazu orthogonal verlaufenden Kompaktierungsrichtungen dargestellt. Um eine dichtere Anordnung der Module zu ermöglichen wird der Schnittbaum mit Hilfe von Rotationen optimiert. Die Rotationen können Änderungen der Kompaktierungsrichtung oder Kompaktierungsreihenfolge beinhalten.

Die resultierenden Platzierungen des Entscheidungsbaums werden mittels einer Kostenfunktion bewertet. Aufbauend auf den Bewertungsresultaten wird die zu verwendende Kompaktierungsrichtung, Topologie und Orientierung für die eingangs zu kompaktierenden Module ausgewählt. Anschließend werden die Kompaktierung und die Verdrahtung für die beiden Module ausgeführt. Das Ergebnis wird in einem neuen Modul gespeichert.

5.2 Layoutmodell

Jedes Modul, genauer dessen Topologie, wird durch eine SKILL-Datei beschrieben. Das folgende Codefragment zeigt einen Auszug einer derartigen Topologiebeschreibung. Das Modul ist, unter Verwendung von MOGLAN, in einer 0.25µm CMOS-Technologie der IHP GmbH erzeugt worden.

```
…
dbCreateRect(CV "Metal1" list(-0.84:-21.09 2.84:-20.39))~>NODE="net11"
dbCreateRect(CV "GatPoly" list(0:-20.51 2:0.51))~>NODE="vdd!"
dbCreateRect(CV "Activ" list(0-51:-20 0:0))~>NODE="net11"
…
```

Die geometrischen Elemente, wie Rechtecke oder Polygone, werden jeweils durch einen SKILL-Befehl repräsentiert. Das Schlüsselwort dbCreateRect beispielsweise zeigt an, dass es sich um ein Rechteck handelt. Der folgende Parameter CV ist für die Darstellung innerhalb des Cadence Design Framework von Bedeutung und hat keinen weiteren Einfluss auf die Platzierung und Verdrahtung. Der nachfolgende Layername wird durch das Mapping des Technologie-Interfaces von ALADIN (siehe Kap. A.1) in das Layernamenssystem von MOGLAN überführt. Die im dem Beispiel aufgeführten Maskennamen "Metal1", "GatPoly" und "Active" werden entsprechend zu "METAL1", "POLY" und "DIFFUSION". Auf diese Weise werden eine Technologie-Unabhängigkeit erreicht und die weiteren Schritte während der Platzierung und Verdrahtung, in Bezug auf die Abfrage von technologiebedingten Werten, wie Mindestabstand bzw. Mindestweite, vereinfacht. Die anschließende Liste der Koordinaten wird mittels eines Skalars in eine Ganzzahl-Darstellung überführt. Mit Hilfe des Netznamens, der in dem Feld NODE abgelegt ist, wird jedem Element ein Knotenpotential zugeordnet.

Zur Reduzierung der Datenmengen und Optimierung der Laufzeit, wird die SKILL-Beschreibung einer Topologie kurz eingelesen, anhand der resultierenden Elemente ein Kantenbild erstellt und wieder aus dem Speicher entfernt. Während der Platzierung werden zu jeder verwendeten Topologie die notwendigen Translationen und Transformationen gespeichert. Nach Abschluss des Platzierungs- und Verdrahtungsvorgangs werden die geometrischen Daten erneut eingelesen, die gespeicherten Operationen auf diese Elemente angewandt und in der Zieldatei abgelegt. Während der Kompaktierung wird für jede Kante des Zielobjekts geprüft, ob zu einer Kante im kompaktierten Objekt eine Minimaldistanz eingehalten werden muss.

Durch die Verwendung von rechtwinkligen, geometrischen Objekten treten lediglich vertikale und horizontale Strecken auf. Da bei einer vertikalen Kante die x-Koordinate der beiden Punkte und bei einer horizontalen Kante entsprechend die y-Koordinaten identisch sind, kann jede Kante mit Hilfe eines 3-Tupel eindeutig identifiziert werden. Eine vertikale Kante mit den Punkten P1 und P2 wird damit durch das Tripel eV = $\{x_{P1}, y_{P1}, y_{P2}\}$ beschrieben, wobei y_{P1} die y-Koordinate des linken und y_{P2} die y-Koordinate des rechten Punkts liefert, während x_{P1} die gemeinsame x-Koordinate der beiden Punkte P1 und P2 darstellt. Um bei der Kompaktierung eine Überprüfung der vertikalen und horizontalen Kanten mit derselben Funktion durchführen zu können, ist in [Wolf99b] vorgeschlagen worden, die horizontalen Kanten um 90° im Uhrzeigersinn zu drehen und anschließend an der x-Achse zu spiegeln. Infolge dieser Transformationsschritte werden die horizontalen Kanten auf vertikale Kanten abgebildet. Eine Unterscheidung zwischen horizontalen und vertikalen Kanten ist bei der Kompaktierung nicht mehr notwendig. Das Tripel $e_H = \{y_{P1},x_{P1},x_{P2}\}$ beschreibt damit eindeutig eine horizontale Kante mit den Eckpunkten P1 und P2. In den Tripeln für vertikale und horizontale Kanten wird also die gemeinsame Koordinate der Eckpunkte an der ersten Stelle ablegt. Auf die einzelnen Elemente des Tripels kann mit Hilfe der Funktionen X(e), Y_L(e) und Y_H(e) zugegriffen werden, wobei X(e) den ersten Eintrag, Y_L(e) die zweite Koordinate und demgemäß Y_H(e) den dritten Wert des Tripels einer Kante e zurückgibt. Den Abstand von zwei Kanten e_1 und e_2 wird im Kompaktierungsverfahren durch die Differenz $X(e_2)– X(e_1)$ berechnet. Infolge der Transformation der horizontalen Kanten gilt diese Abstandsbestimmung sowohl für vertikale als auch horizontale Kanten.

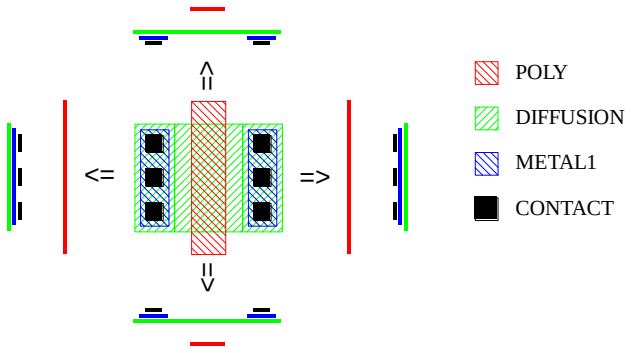


Abbildung 5-1: Generierung der Kantenbilder am Beispiel eines einfachen MOS-Transistors

Bei der Kompaktierung sind für die Berechnung des minimalen Verschiebungsvektors lediglich die äußeren Kanten jeder Ebene eines Objekts zu berücksichtigen. Aus dem Kantenbild können alle verdeckten Kanten desselben Layers entfernt werden. Auf diese Weise wird nicht nur der Speicherbedarf reduziert, sondern auch die Komplexität bei der Bestimmung des minimalen Translationsvektors. Dadurch wird eine schnellere Berechnung der Kompaktierung ermöglicht. In Abbildung 5-1 sind für einen einfachen MOS-Transistor die entsprechenden Kantenbilder zu jeder Seite dargestellt. Es wird zu Beginn des Platzierungs- und Verdrahtungsprozesses für jede Topologie der zu bearbeitenden Module ein solches Kantenbild erstellt. Es sind die Kantenbilder für jede der vier Seiten einer Topologie zu erzeugen. Dazu wird beim sequentiellen Einlesen der Layoutdaten das aktuelle Rechteck bzw. Polygon durch eine Menge an Kanten, basierend auf den Seiten des jeweiligen Elements, dargestellt. Die Kanten werden, gemäß ihrer Orientierung, den vier Listen zugeordnet. Zur Speicherung der Kanten wird jeweils eine sortierbare Liste verwendet. Die Einträge werden alphanumerisch nach den Maskennamen gruppiert. Innerhalb einer Gruppe erfolgt die Sortierung der Kanten in Abhängigkeit von der Lage der Kanten. Horizontale Kanten werden von links nach rechts und vertikale Kanten von unten nach oben sortiert. Nachdem die Kanten in die Listen

eingefügt worden sind, wird ein Verschmelzungsprozess gestartet, mit dessen Hilfe redundante und verdeckte Kanten aus den Listen entfernt werden. Der Ablauf für diesen Vorgang ist in Algorithmus 5-1 skizziert.

Algorithmus 5-1: Verschmelzen von Kantenbildern

```
für i:=1 bis n
   für j:=1 bis n
      wenn ((i≠j) und (L(ei)=L(ej)) dann
         wenn ((X(ej)>X(ei)) und (Orientation(ei)=NORTH)
               oder (Orientation(ei)=EAST)))
               oder ((X(ej)<X(ei) und ((Orientation(ei)=SOUTH)
               oder (Orientation(ei)=WEST)))) dann
            wenn ((YL(ej)≤YL(ei)) und (YH(ej)≥YH(ei)) dann
               // ej verdeckt ei vollständig
               lösche ej;
            sonst wenn ((YL(ej)>YL(ei)) und (YH(ej)<YH(ei)) dann
               // ej verdeckt einen Teil von ei
               erstelle Kopie eN von ei und füge eN an das Ende der Liste an;
               YH(ei):=YL(ej);
               YL(eN):=YH(ej);
            sonst wenn ((YL(ej)<YL(ei)) und (YH(ej)≤YH(ei)) und (YH(ej)>YL(ei))) dann
               YL(ei):=YH(ej);
            sonst wenn ((YL(ej)≥YL(ei)) und (YH(ej)>YH(ei)) und (YH(ej)<YL(ei))) dann
               YH(ei):=YL(ej);
            wenn_ende
         sonst wenn (X(ei)=X(ej)) dann
            wenn ((YL(ej)>YL(ei)) und (YH(ej)<YH(ei))) dann
               // ej ist vollständig in ei enthalten
               lösche ej;
            sonst wenn ((YL(ej)≤YL(ei)) und (YH(ej)≥YH(ei))) dann
               // ei ist vollständig in ej enthalten
               lösche ei;
            sonst wenn ((YL(ej)≤YL(ei)) und (YH(ej)≤YH(ei)) und (YH(ej)≥YL(ei))) dann
               YL(ei):=YL(ej);
               lösche ej;
            sonst wenn ((YL(ej)≥YL(ei)) und (YH(ej)≥YH(ei)) und (YL(ej)≤YH(ei))) dann
               YH(ei):=YH(ej);
               lösche ej;
            wenn_ende
         wenn_ende
      wenn_ende
   für_ende
für_ende
Kantenliste sortieren
```

Es wird über alle n Kanten eines Kantenbildes iteriert. Dabei wird jede Kante mit allen anderen Kanten verglichen. Wenn zwei Kanten den gleichen Layer aufweisen und es sich nicht um dieselbe Kante handelt, wird deren Lage zueinander untersucht. Wie bereits beschrieben, wird jede Kante durch ein Tripel dargestellt. Die Funktion X(e) liefert den ersten Eintrag des Tripels. Zusätzlich ist es möglich, für jede Kante deren Orientierung abzufragen. Basierend auf diesen Informationen ist es möglich, einen Vergleich dahingehend durchzuführen, ob sich die Kante e_j näher am entsprechenden Rand des Moduls befindet als die Kante e_i. Wenn dieser Fall vorliegt, bleibt zu prüfen, inwieweit eine Überlappung der beiden Kanten vorliegt. In Abhängigkeit von den Resultaten dieses Vergleichs ist die Kante e_i aus der Kantenliste zu entfernen, eine neue Kante auf der Basis von e_i zu erstellen oder die Abmessung von e_i anzupassen. Wenn der erste Eintrag der Tripel von e_i und e_j übereinstimmt, ist ebenfalls eine Anpassung der Kanten notwendig. Falls e_j durch e_i vollständig verdeckt wird, erfolgt das Löschen von e_j aus dem Kantenbild und umgekehrt. Für den Fall, dass lediglich eine Teilüberlappung vorliegt, wird e_i entsprechend verändert und e_j entfernt.

Ebenso erfolgt nach einem kombinierten Schritt, bestehend aus der Kompaktierung und Verdrahtung zweier Module, die Erstellung der Kantenbilder für die resultierende neue Zelle. Diese bestehen aus den verschmolzenen Kantenbildern der beiden Zellen sowie den erzeugten Leitungen zwischen den beiden Modulen.

5.3 Verbindungsgraph

Die Synthese einer analogen Schaltung beginnt in der Regel mit einer geeigneten Schaltungstopologie bzw. einer existierenden Schaltung, die aus einer vergleichbaren Anwendung hervorgeht. Anschließend wird iterativ die Schaltung mit Hilfe geeigneter Simulationen optimiert. Infolge der zunehmenden Strukturminiaturisierungen kommt es zu einer Verschlechterung der elektrischen Parameter für analoge Schaltungen. Dadurch verringert sich das Designfenster, innerhalb dessen die Parameter der Elemente der Schaltung gewählt werden müssen. Als Folge davon vergrößert sich der Optimierungsaufwand für analoge Schaltungen.

In [Wolf98a] ist der Designassistent von ALADIN (Abkürzung für Automatic Layout Design Aid for Analog Integrated Circuits, siehe Anhang A) für analoge Experten entwickelt worden, um deren Kenntnisse in den Syntheseprozess mit einzubeziehen. Das im Rahmen dieser Arbeit erstellte Platzierungs- und Verdrahtungswerkzeug bezieht seine Eingabedaten aus den Modulgeneratoren, die ebenfalls einen Teil der ALADIN-Umgebung bilden und den Eingaben des Schaltungsdesigners entsprechen, die über den Designassistenten erfolgen.

Nach der Eingabe des Schaltbildes in den Schematic-Editor des DesignFramework II von Cadence, wird die Schaltung in verschiedene Module gruppiert. Für jede Schaltungsgruppe wird ein geeigneter Generator aufgerufen, um das Layout für das entsprechende Modul zu erzeugen. Der Generator wird aus einer Liste existierender Generatoren ausgewählt, die mit Hilfe der Modulgeneratorumgebung erstellt worden sind. Zu jedem Generator ist ein entsprechendes Schaltbild vorhanden, mit dessen Hilfe die Informationen über externe Verbindungen und Eigenschaften, wie Weite und Länge der Transistoren, von der Schaltung an den Generator übergeben werden können. Die Layoutgenerierung kann mittels der Kapazitäts-Sensitivitätsmatrix beeinflusst werden, indem die parasitären Kapazitäten kontrolliert werden [Wolf97]. Nach der Erstellung des Layout des Moduls werden automatisch die Anschlüsse der Zelle ermittelt und in einer separaten Datei gespeichert. Es ist möglich, für ein Modul mehrere Layoutversionen zu erzeugen, beispielsweise indem andere Faltungsfaktoren gewählt werden. In Fall einer Differenzstufe ist es auch möglich, verschiedene Layoutstrukturen, wie z.B. eine meanderförmige Anordnung bzw. eine kreuzgekoppelte Version der Differenzenpäarchen, über die entsprechenden Generatoren erzeugen zu lassen. Während des Platzierungsvorgangs wird aus den verschiedenen Versionen die am besten geeignete Struktur ausgewählt und in das Platzierungsergebnis aufgenommen.

Zu Beginn des Platzierungsverfahrens wird, basierend auf den Modulen und Netzlisten, ein Graph erstellt, der die Beziehungen zwischen den Modulen dokumentiert. Zur Untersuchung dieser primären Beziehungen wird die Netzliste ausgewertet. Es ergibt sich eine Verbindungsgraph G_{conn}= (V_{conn}, E_{conn}), dessen Knoten auf die jeweiligen Module verweisen. Die Kanten enthalten das Verbindungsgewicht zwischen den Modulen, welches sich aus den Prioritäten der einzelnen Netze ableiten lässt. In Algorithmus 5-2 ist der Ablauf der Schritte für das Erstellen des Verbindungsgraphen dargestellt.

Es wird über die Menge M aller Module der Schaltung, für die die Platzierung und Verdrahtung durchgeführt werden soll, iteriert. Zu jedem Modul wird ein neuer Knoten v_i erstellt und der

Knotenmenge V des Verbindungsgraphen hinzugefügt. Innerhalb des Knotens wird das Modul in einem Binärbaum ablegt. Dieser beinhalt zunächst nur einen Wurzelknoten.

Algorithmus 5-2: Erstellen des Verbindungsgraphen

```
für i:=1 bis |M|
   erstelle für das Module mi∈M einen neuen Knoten vi;
   vi der Knotenmenge Vconn des Verbindungsgraphen hinzufügen;
   für j:=1 bis |Vconn|
      wenn ((vi≠vj) und (HasConnection(vi,vj)=TRUE)) dann
         wij:=CalcConnectionWeight(vi,vj);
         erstelle neue Kante ei mit dem Gewicht wij, um die Knoten vi und vj zu
         verbinden;
         ei der Kantenmenge Econn des Verbindungsgraphen hinzufügen;
      wenn_ende
   für_ende
für_ende
```

In einer weiteren Iteration über alle bereits erstellten Knoten wird geprüft, inwieweit eine Verbindung zwischen dem aktuellen Modul und bereits erfasst Modulen vorliegt. Dazu wird nach gemeinsamen Netzen zwischen dem Modul des Knotens v_i und der Module der Knoten v_j gesucht. Existieren gemeinsame Netze, so wird das Verbindungsgewicht w_{ij} berechnet und dem Graphen eine neue Kante, der die Knoten v_i und v_j verbindet, hinzugefügt. Das Gewicht der Kante entspricht w_{ij}.

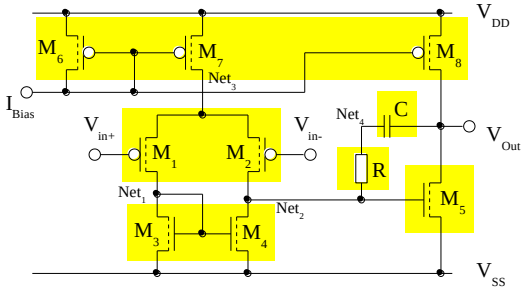
Das Verbindungsgewicht berechnet sich gemäß der folgenden Gleichung

$$w_{ij} = \sum_{k=1}^{\#Nets} \delta_{ij}^{(k)} \cdot \alpha_{net(k)}, \tag{5-1}$$

wobei $\alpha_{net(k)}$ die Priorität des k-ten Netzes und $\delta_{ij}^{(k)}$ wie folgt definiert ist:

$$\delta_{ij}^{(k)} = \begin{cases} 1 & \text{falls net(k) in Modul i und j} \\ 0 & \text{sonst} \end{cases}. \tag{5-2}$$

Nach der Konstruktion des Verbindungsgraphen wird jedem Knoten ein eindeutiger Schlüssel in Form einer Nummer zugewiesen. Auf diese Weise können die Knoten im weiteren Verlauf eindeutig identifiziert werden.



Netz	Priorität	Typ
V_{DD}	10	Supply
V_{SS}	10	Supply
Net_{Bias}	10	Supply
Net_1	40	Signal
Net_2	50	Signal
Net_3	20	Supply
Net_4	40	Signal
Net_{Out}	40	Signal

Abbildung 5-2: Zweistufiger CMOS-Operationsverstärker mit zugehöriger Netzliste

Der Ablauf für die Erstellung eines Verbindungsgraphen soll anhand des Beispiels in Abb. 5-2 nochmal kurz verdeutlicht werden. Es ist das Schaltbild eines zweistufigen CMOS-Operationsverstärkers dargestellt. Die erste Stufe besteht aus einer p-Kanal Differenzstufe (M_1, M_2)

sowie der zugehörigen aktiven Last (M_3, M_4). Die Stromversorgung des Verstärkers erfolgt mit Hilfe einer Stromspiegelschaltung bestehend aus den p-Kanal Transistoren M_6 bis M_8 sowie die Bias-Stromquelle I_{Bias}. Die zweite Stufe wird durch die Stromquelle M_8 und den n-Kanal Treibertransistor M_5 realisiert. Zur Kompensation ist ein RC-Glied eingefügt worden. Die p-Kanal-Transistoren M_1 und M_2 bilden das Modul der Differenzstufe. Die n-Kanal-Transistoren M_3 und M_4 sowie die p-Kanal-Transistoren M_6-M_8 werden jeweils in einem Stromspiegelmodul zusammengefasst. Für den Widerstand, die Kapazität und die Ausgangsstufe, in der Form des n-Kanal-Transistors M_5, wird je ein Modul erstellt. Im Schaltbild sind die Bulk-Anschlüsse nicht explizit aufgeführt. Es wird davon ausgegangen, dass das Bulk der p-Kanal-Transistoren mit V_{DD} verbunden ist und entsprechend bei den n-Kanal-Transistoren mit V_{SS}.

Neben der Schaltung ist die Liste der Netze aufgeführt. Für jedes Netz sind eine Priorität und eine Charakterisierung des Netzes gegeben. In der Tabelle sind lediglich diejenigen Netze aufgeführt, die zu einer Verbindung der Module beitragen. Die Anwendung des Algorithmus 5-2 liefert den in Abb. 5-3 gezeigten Verbindungsgraphen. Dieser Verbindungsgraph dient als Startlösung für das Platzierungsverfahren.

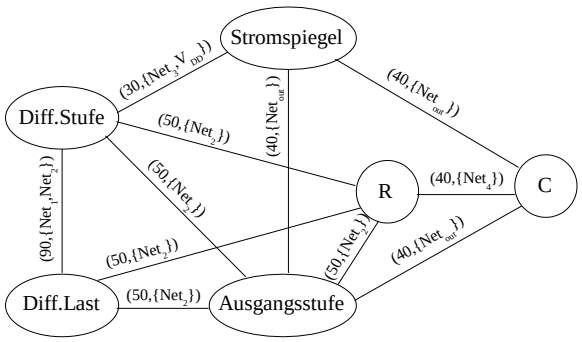


Abbildung 5-3: Verbindungsgraph des Beispiels aus Abbildung 5-2

Die Knoten beinhalten die jeweiligen Module, die im Schaltbild ausgewählt worden sind. An jeder Kante des Graphen ist neben dem Kantengewicht eine Liste der Netze aufgeführt, die zur Berechnung des Gewichts beigetragen haben.

5.4 Kompaktierungsverfahren

Die Zielstellung einer Kompaktierung ist ein flächenminimales entwurfsregelgerechtes Layout. Es ist ein aufwändiger Prozess zu dem verschiedene Vorgehensweisen und Algorithmen entwickelt wurden. Eine Übersicht dazu ist in [Boye88, Cho85] gegeben. Obwohl die Ideen zur horizontalen und vertikalen Kompaktierung orthogonaler Layoutelemente ähnlich sind, wurden verschiedene Ansätze zur Lösung des Kompaktierungsproblems vorgeschlagen. Jedes Verfahren weist seine Vor- und Nachteile auf. Neben der Optimierung der Layoutfläche sollten Kompaktierungsalgorithmen in der Lage sein, gezielt einzelne Parameter einer Schaltung zu berücksichtigen und zu verbessern.

Der überwiegende Teil der Algorithmen basiert auf einem 1-dimensionalen Ansatz, wenn auch einige Verfahren dazu übergehen, eine Kompaktierung in beide Richtungen (horizontal und vertikal) simultan durchzuführen. In [Sast82, Schl83] ist jedoch gezeigt worden, dass das Problem der 2-dimensionalen Kompaktierung NP-vollständig ist. Eine Verschiebung der Zellen in nur einer Richtung ermöglicht eine bessere Vorhersagbarkeit der Ergebnisse. Die Ausgaben eines Kompaktierers müssen den geometrischen und elektrischen Entwurfsregeln der zu verwendenden Technologie entsprechen.

Weitestgehend alle implementierten Kompaktierer lassen sich auf den Ansatz der Scherlinien (Shear-line) [Aker70, Dunl78, Dunl80], des virtuellen Gitters [Ackl83, Boye83, Boye88, West81a, West81b] oder des Restriktionsgraphen (Constraint-graph) [Anzi93, Anzi03, Cho77, Fang91, Gao89, Hsue79, King84, Liao83, Schi83, Wolf83, Wolf96, Wolf99a, Wolf99b] zurückführen. Die Kompaktierungsalgorithmen können basierend auf der Verschiebung der Elemente weiter klassifiziert werden, nämlich hinsichtlich einer 1-dimensionalen, 1,5-dimensionalen und einer 2-dimensionalen Kompaktierung. Bei einer 1-dimensionalen Kompaktierung wird die Layoutfläche reduziert, indem die Abstände zwischen den Objekten in jeder Richtung nacheinander minimiert werden. Die grundlegende Idee der 1,5-dimensionalen Kompaktierung besteht darin, neben der 1-dimensionalen Kompaktierung genügend laterale Verschiebungen zu erlauben, so dass mögliche Störungen beseitigt werden können. Bei der 2-dimensionalen Kompaktierung wird versucht, die Layoutfläche direkt zu minimieren.

Im Nachfolgenden wird ein kurzer Überblick zu den Ansätzen der Scherlinien (Kap. 5.4.1), des virtuellen Gitters (Kap. 5.4.2) und des Restriktionsgraphen (Kap. 5.4.3) gegeben. Ein großer Teil der entworfenen Kompaktierer sowie der in dieser Arbeit verwendete Algorithmus machen vom Restriktionsgraphen Gebrauch. Dieses Modell offeriert eine bessere Flexibilität im Vergleich zu den anderen Verfahren und erlaubt eine effiziente Implementierung. In [Wolf85] wurden verschiedene 1-dimensionale Kompaktierungsstrategien untersucht und miteinander verglichen. Dabei wurden die Vorteile des Restriktionsgraphen bestätigt.

In Kap. 5.4.4 wird das in dieser Arbeit verwendete Kompaktierungsmodell näher erläutert. Es basiert auf dem Restriktionsgraphen und ist Bestandteil der Modulgeneratorumgebung MOGLAN [Wolf96, Wolf97, Wolf98b, Wolf99a, Wolf99b].

5.4.1 Modell der Scherlinie

Eines der ersten Beispiele für einen Kompaktierungsalgorithmus basiert auf der Verwendung von Scherlinien. Die Methode wurde erstmals in [Aker70] vorgestellt, um das Layout einer Schaltung zu verdichten, indem überschüssige Leerräume entfernt werden. Dazu werden überbreite Bänder, die Verdichtungsstreifen (compression ridges), sukzessive über die Gesamtbreite des Layouts hinzugefügt und wieder entfernt. Ein einzelnes Band kann dabei durchgehend sein oder an einigen Punkten versetzt verlaufen. Die Stelle eines Versatzes des Verdichtungsstreifens ist eine Scherlinie. Auf diese Weise kann eine höhere Layoutdichte erzielt werden.

Im ursprünglichen Ansatz wurde ein grobmaschiges System verwendet. Das Layout wird durch Einträge in einer rechteckförmigen Matrix repräsentiert. Die Matrix wird auf der Suche nach den Verdichtungsstreifen durchlaufen. Die Kompaktierung erfolgt durch wiederholtes Abtasten in zwei Richtungen und dem Entfernen von überschüssigen Leerräumen solange, bis keine weiteren Bänder mehr gefunden werden können. Ein Beispiel für durchgehende und versetzte Verdichtungsstreifen mit Scherlinien ist in Abb. 5-4 dargestellt.

In [Dunl80] ist das Verfahren dahingehend erweitert worden, das die Layoutdaten in einer Datenbank hierarchisch abgelegt und das Layout in einer symbolischen Form dargestellt werden kann. Die Kompaktierung erfolgt nun durch Anwendung der Verdichtungsstreifen auf einem virtuellen Gitter des symbolischen Layouts.

Der hauptsächliche Vorteil der Kompaktierung mit Hilfe der Scherlinien ist die konstruktionsbedingte Einfachheit. Es kann mit einem geringen Aufwand auf ein hierarchisches Schema abgebildet werden,

bei dem die einzelnen Zellen als verbotene Zonen repräsentiert werden, in die keine Verdichtungsstreifen gelegt werden dürfen. Ein Nachteil der beschriebenen Vorgehensweise liegt in der Zeitkomplexität der Berechnung der Bänder. Eine rekursive Methode zur Bestimmung der Verdichtungsstreifen ist sehr zeitaufwändig. Zusätzlich ist es nicht möglich, benutzerdefinierte Randbedingungen zu berücksichtigen.

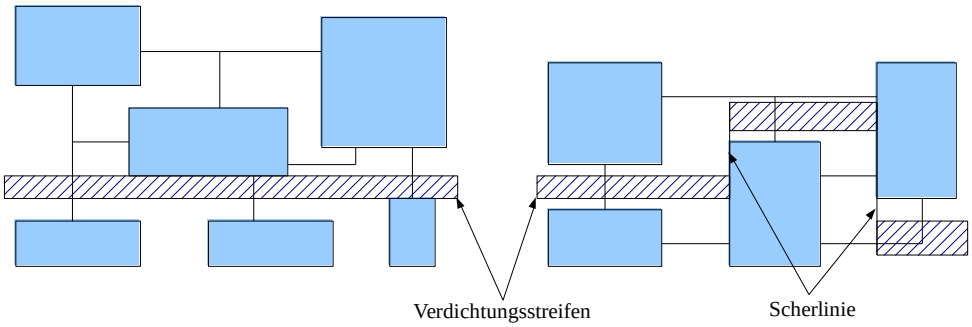


Abbildung 5-4: Beispiele für Verdichtungsstreifen und Scherlinien

5.4.2 Virtuelles Gitter

Bei dem Modell des virtuellen Gitters wird angenommen, dass sich alle Elemente auf einem beweglichen Gitter befinden. Der Abstand zwischen benachbarten Gitterlinien wird festgelegt durch die Dichte und Wechselbeziehung der Elemente, die sich entlang der Linien befinden. Eine Kompaktierung reduziert die Distanz zwischen den Gitterlinien und damit zwischen den zugehörigen Objekten. Der Ansatz für diese Vorgehensweise zur Kompaktierung ist erstmals im Designsystem MULGA [Ackl83, West81] verwendet worden. Die Kompaktierung wird zunächst in einer Richtung ausgeführt, indem den virtuellen Gitterlinien die Elemente zugeordnet werden. Während dieser Phase werden die Entwurfsregelverletzungen nur lokal überprüft. Anschließend erfolgt die Kompaktierung in die andere Richtung. Dabei werden noch nicht erfasste Regelverletzungen mit in Betracht gezogen und bei der Wahl der virtuellen Gitterlinien mit berücksichtigt. Ein Beispiel für eine Kompaktierung mit Hilfe des virtuellen Gitters ist in Abb. 5-5 gegeben.

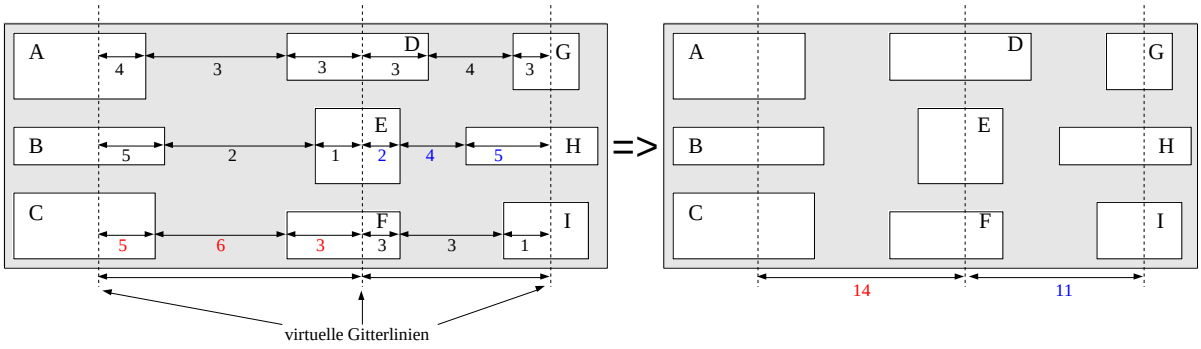


Abbildung 5-5: Beispiel für die Kompaktierung basierend auf dem virtuellen Gitter

Die Vorteile für diesen Ansatz liegen darin, dass die Zeitkomplexität und der Implementierungsaufwand verhältnismäßig gering sind. Nachteile des Ablaufs sind dadurch gegeben, dass alle Elemente auf einer Gitterlinie verschoben werden. Auf diese Weise könnte es zu einer Verschwendung von Flächen kommen, wenn die Elemente auf einer Linie deutliche Größenunterschiede aufweisen. Ebenso ist es schwierig, wie bei der Kompaktierung mittels der Scherlinien, benutzerdefinierte Nebenbedingungen zu berücksichtigen. Untersuchungen in [Wolf85]

haben gezeigt, dass die Ergebnisse von Kompaktierern basierend auf dem virtuellen Gitter im Vergleich zur Verwendung des Restriktionsgraphen nicht durch den Schaltungsdesigner vorhersagbar sind. Die Voraussagbarkeit bildet jedoch einen großen Vorteil der 1-dimensionalen Kompaktierer.

5.4.3 Restriktionsgraph

Der überwiegende Teil der Arbeiten, die sich mit der Layoutkompaktierung befassen, basiert auf dem Modell des Restriktionsgraphen. Dieses Modell wurde erstmals in [Cho77] präsentiert. Der Ansatz des Restriktionsgraphen löst das Kompaktierungsproblem grundlegend in zwei Schritten. Zunächst wird der Graph aus den Layoutdaten heraus aufgebaut, um die relativen Positionen und die erforderlichen, technologiebedingten Mindestabstände zwischen den Elementen zu erfassen. Anschließend erfolgt die Auswertung des Graphen, um die Layoutfläche mit Hilfe der Methode des längsten Pfades zu minimieren. Das einfache Modell des Restriktionsgraphen trennt das Kompaktierungsproblem in zwei unabhängige Kompaktierungen, die eine erfolgt in X-Richtung, die andere in Y-Richtung. Während der X-Kompaktierung werden die Elemente ausschließlich in horizontaler Richtung und entsprechend bei der Y-Kompaktierung nur in vertikaler Richtung verschoben. Grundsätzlich wird bei dieser Vorgehensweise angenommen, dass der Flächenbedarf infolge der getrennten Verschiebung hinreichend minimiert werden kann. Eine umfassende Erörterung der Methode des Restriktionsgraphen ist in [Cho85, Kede83, Liao83, Schi83, Ullm84] zu finden.

Die Knoten des Restriktionsgraphen repräsentieren die Elemente der zu bearbeitenden Schaltung. Die gewichteten, gerichteten Kanten modellieren neben den technologie-spezifischen Abstandsregeln benutzerdefinierte Nebenbedingungen. Der Aufbau des Restriktionsgraphen ist in der Regel ein zeitaufwendiger Prozess. Im schlechtesten Fall existiert zwischen jedem Knotenpaar eine Verbindung. Für die Kompaktierung hingegen ist jedoch meist nur eine kleine Teilmenge der Kanten des Graphen erforderlich. Es sind für ein Element meist nur die Abstände zu den benachbarten Objekten einzuhalten.

Für die Erstellung des Graphen sind verschiedene Techniken vorgeschlagen worden. Nachfolgend werden einige davon kurz beschrieben. Im CABBAGE Kompaktierer [Hsue79] beispielsweise wird die am weitesten bekannte Methode des Schattenwurfs (shadow propagation) zur Konstruktion des Restriktionsgraphen verwendet. Die grundlegende Idee zeichnet sich dadurch aus, dass entlang der Kompaktierungsrichtung ein Schatten vom denjenigen Element ausgebreitet wird, für das die benachbarten Objekte zu bestimmen sind. Der Schatten wird durch eine imaginäre Lichtquelle auf der anderen Seite des Elements verursacht. Der Schatten wird auf beiden Seiten vergrößert, um diagonale Nebenbedingungen mit zu berücksichtigen. Immer wenn der Schatten auf ein anderes Element trifft, wird eine Kante zwischen den korrespondierenden Knoten dem Graphen hinzugefügt. Der betroffene Teil des Schattens wird entfernt. Der Vorgang wird solange fortgesetzt, bis der Schatten vollständig aufgelöst oder keine weiteren Elemente, die im Schatten liegen, vorhanden sind. Anschließend werden die Schritte für die anderen Elemente des Layouts wiederholt. In Abb. 5-6 ist die Methode des Schattenwurfs zum Aufbau des Restriktionsgraphen anhand eines Beispiels dargestellt. Vor das Element A wird eine imaginäre Lichtquelle aufgestellt. Anschließend werden alle Elemente erfasst, die ganz bzw. teilweise im resultierenden Schatten des Elements A liegen.

Anstelle nach noch schnelleren Algorithmen zu suchen, liegt ein weiterer Ansatz für die Erstellung des Graphen darin, eine geeignete Datenstruktur als Ausgangspunkt zu nutzen. In [Cho85] wird die "Corner-Stitching"-Struktur [Oust84] vorgeschlagen, um benachbarte Elemente schnell zu finden.

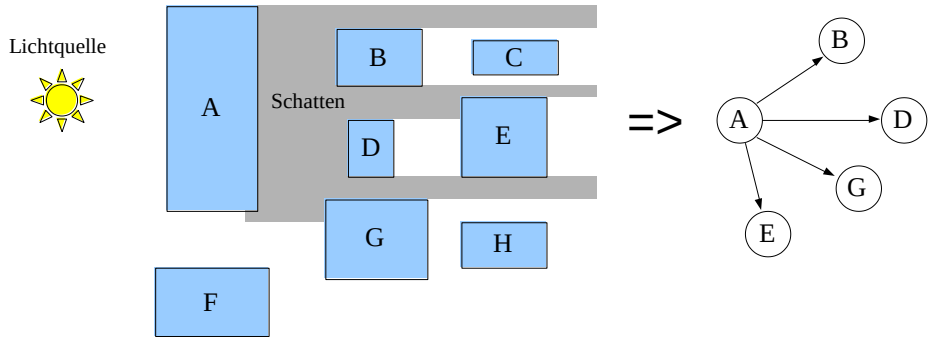


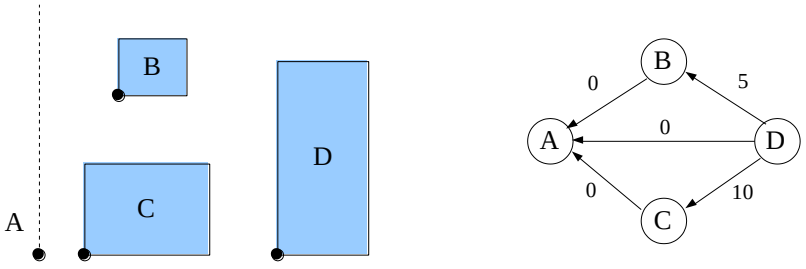
Abbildung 5-6: Beispiel für Erstellung eines Restriktionsgraphen mittels der Schattenwurf-Methode

Ein weiterer Ansatz zur Erzeugung der Nachbarschaftsbeziehungen ist durch das Scan-Line-Verfahren [Bair78, Szym83] gegeben. Der Ablauf für die Suche nach Abstandsbeziehungen zwischen den Elementen ist in diesem Fall vergleichbar mit der Entwurfsregelprüfung [Bair75, Bent80].

Die Abarbeitung für die vertikale und horizontale Richtung verlaufen analog, so dass an dieser Stelle lediglich die weiteren Schritte der Konstruktion des Restriktionsgraphen für die X-Kompaktierung betrachtet werden. Das Ergebnis der Suche der Nachbarschaftsbeziehungen der Elemente ist eine Menge an Ungleichungen der Form $x_i - x_j \geq a$. Mit Hilfe der Ungleichung wird beschrieben, dass sich die x-Koordinate des Elements i mindestens a Einheiten rechtsseits der x-Koordinate des Elements j befinden soll. Für jede Ungleichung in der gezeigten Form existiert im Restriktionsgraphen eine Verbindungskante vom Knoten i zu Knoten j mit dem Gewicht a. Auf diese Weise werden die Beschränkungen für die X-Kompaktierung dargestellt. In Abb. 57 ist ein Beispiel für einen Verbindungsgraphen gegeben. Das Auswerten der Objekte in Abb. 5-7(a) liefert die in Gl. (5-3) gezeigten Ungleichungen aus denen sich der Restriktionsgraph in Abb. 5-7(b) ergibt.

$$\begin{array}{ll} x_B - x_A \geq 0 & x_D - x_B \geq 5 \\ x_C - x_A \geq 0 & x_D - x_C \geq 10. \\ x_D - x_A \geq 0 & \end{array} \qquad (5\text{-}3)$$

Nach dem Aufbau des Restriktionsgraphen ist im nächsten Schritt der kritische Pfad im Graphen zu bestimmen. Dieser Pfad muss alle Abstandsforderungen erfüllen. Dazu wird der längste Pfad ermittelt. Die Länge des Pfades ergibt sich aus den Gewichten der Kanten vom Start- bis zum Zielknoten. Alle Elemente entlang dieses Pfades bestimmen damit die erzielbare minimale Breite des Layouts. Es werden außerdem alle möglichen Entwurfsregelverletzungen, die auch in der Eingabe enthalten sein können, automatisch korrigiert. Die eigenständige Korrektur der Entwurfsregeln ist eine wichtige Eigenschaft des Kompaktierers, die eine Zeitersparnis für den Schaltungsdesigner mit sich bringt.



(a) (b)

Abbildung 5-7: (a) Beispiel für eine Kompaktierung in westlicher Richtung, (b) zugehöriger Restriktionsgraph

Die Suche nach den kürzesten Wegen in einem Graphen ist ein bekanntes und viel untersuchtes Problem. Der Wechsel zu einer Suche nach dem längsten Pfad erfordert lediglich eine geringe Modifikation der bekannten Algorithmen [Szym83]. Für den in Abb. 5-7(b) gezeigten Restriktionsgraphen ergibt sich der in Abb. 5-8(b) dargestellten längsten Pfads. Das zugehörige Kompaktierungsergebnis ist der Abb. 5-8(a) zu entnehmen.

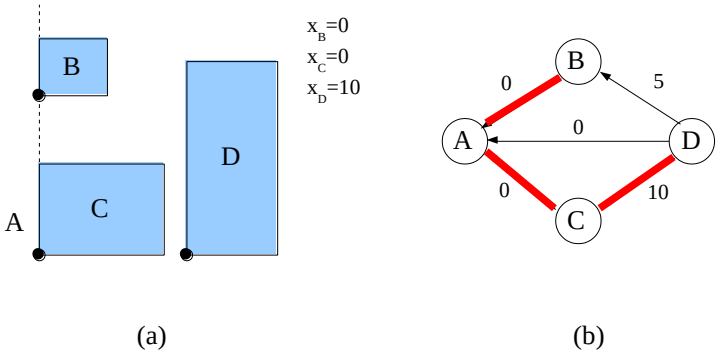


Abbildung 5-8: (a) Kompaktierungsergebnis des Beispiels aus Abb. 5-7(a), (b) Längster Pfad im Restriktionsgraphen

Eine Vielzahl an Kompaktierern, die von Restriktionsgraphen Gebrauch machen, erlauben bei den Beschränkungen sowohl obere als auch untere Schranken [Croc87, Kede84, Liao83]. Eine untere Schranke liegt vor, wenn zwei Elemente i und j eine gegebene Distanz b zueinander nicht überschreiten dürfen. Die Verbindung wird im Graphen als ein Paar Kanten zwischen den Knoten i und j dargestellt, wobei jede der beiden Kanten das Gewicht -b zugewiesen bekommt. Im Gegensatz dazu beschreibt eine obere Schranke eine mindestens einzuhaltende Distanz a zwischen zwei Elemente i und j. Die Darstellung erfolgt durch eine Kante vom Knoten i zum Knoten j mit dem Kantengewicht a. Die Verwendung beider Beschränkungen im Restriktionsgraphen erschwert den Kompaktierungsablauf und kann zu Zyklen innerhalb des Graphen führen. Positive Zyklen im Graphen führen dazu, dass das Kompaktierungsproblem unlösbar wird, da in diesem Fall keine realisierbare Topologie existiert. Ein Beispiel für einen solchen Zyklus ist in Abb. 5-9 gegeben. Das Element B, welches sich innerhalb des U-förmigen Elements A befindet, ist überbestimmt. Dies wird durch die in Gl. (5-4) dargestellten Ungleichungen verdeutlicht. Es ist offensichtlich, dass nicht beide Ungleichungen gleichzeitig erfüllt sein können.

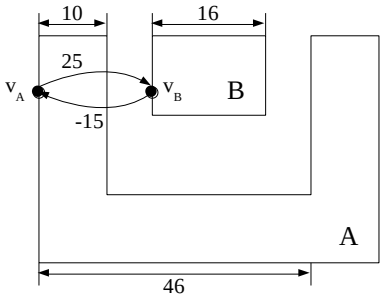


Abbildung 5-9: Beispiel für einen positiven Zyklus in einem Restriktionsgraphen

$$\left.\begin{array}{rcl} v_A + 10 + 15 & \leq & v_B \\ v_B + 16 + 15 & \leq & v_A + 46 \end{array}\right\} \Rightarrow v_A + 25 \leq v_B \leq v_A + 15. \qquad (5\text{-}4)$$

Ein entscheidender Vorteil des Restriktionsgraphen gegenüber den bisher präsentierten Modellen zur Kompaktierung besteht darin, dass durch den Schaltungsdesigner definierte Nebenbedingungen berücksichtigt werden können. Die Minimierung der Layoutfläche wird durch eine Verschiebeoperation in nur einer Richtung zurzeit erreicht. Eine getrennte Betrachtung und Ausführung der horizontalen und vertikalen Kompaktierungen führt nicht immer zu einem optimalen Ergebnis bzgl. der Flächenreduzierung. Aus diesem Grund sind in [Kede84, Sast82, Schl83] Modelle für eine 2-dimensionale Kompaktierung vorgestellt worden. In [Sast82, Schl83] wurde gezeigt, dass diese Form der Kompaktierung jedoch NP-vollständig ist. Alle präsentierten Modelle einer 2-dimensionalen Kompaktierung weisen eine vergleichbare Vorgehensweise auf. Es wird stets eine Menge zulässiger Nebenbedingungen für eine Anzahl regelkonformer Layouts betrachtet. Die Suche bzw. Auswahl einer optimalen Anordnung des Layouts aus den gegebenen Mengen erfolgt heuristisch mittels eines Branch-And-Bound-Algorithmus. Ein großer Nachteil der 2-dimensionalen Kompaktierung ist darin zu finden, dass neben dem Problem der NP-Vollständigkeit die Ergebnisse für eine gute Anordnung der Elemente nur durch eine umfangreiche Suche ermittelt werden können. Als Folge davon ist es dem Schaltungsdesigner nicht mehr möglich, die Topologie der fertigen Layouts vorherzusagen.

Ein einfacherer Ansatz ist in [Wolf84] gezeigt worden. Hier wird die Kompaktierung in einer Richtung dadurch verbessert, indem ein blockierendes Element in der orthogonalen Richtung verschoben wird. Obwohl dieses Modell strenggenommen ebenfalls eine 1-dimensionale Kompaktierung beschreibt, wird eine Restrukturierung der Eingabe derart durchgeführt, dass eine gute 2-dimensionale Optimierung erreicht wird. Aus diesem Grund wird das Verfahren 1,5-dimensionale Kompaktierung genannt, da die Geometrie nicht über die Freiheitsgrade wie bei einer echten 2-dimensionalen verfügt. Der Algorithmus verwaltet einen Nachbarschaftsgraphen. Die Knoten des Graphen repräsentieren die Elemente, die Kanten stehen für horizontale und vertikale Nachbarschaften. Zwei Elemente haben eine horizontale Verschiebungskante, wenn sie eine gemeinsame vertikale Grenze aufweisen. Entsprechend werden die Knoten von Elementen mit einer vertikalen Kante verbunden, wenn die Elemente horizontal aneinander grenzen. Die Kanten enthalten Bezeichner mit der minimal erlaubten Distanz zwischen den Elementen. Vier zusätzliche Knoten werden dem Graphen hinzugefügt, um alle Elemente innerhalb eines vorgegebenen Rechtecks zu halten. Die freien Flächen werden bei der Berechnung der Kanten für die Nachbarschaftsbeziehungen ignoriert. Das Verfahren geht davon aus, dass die Eingabe ein teilweise fertiges Layout ist, welches aus der zweimaligen Anwendung eines 1-dimensionalen Kompaktierers hervorgeht. Es werden zwei Listen "Floor" und "Ceiling" unterhalten. Die Liste "Floor" enthält alle Elemente, die vom oberen Rand aus sichtbar sind und möglicherweise Nachbar eines zukünftigen Blocks werden. Die Elemente, die sofort verschoben werden können, d.h. alle vom unteren Rand aus sichtbaren Elemente, werden in der Liste "Ceiling" abgelegt. Die Auswahl der untersten Elemente in der "Ceiling"-Liste und die Verschiebung an eine Stelle in der "Floor"-Liste erfolgt derart, dass die Lücke zwischen "Floor" und "Ceiling" maximiert wird. Der Vorgang wird solange wiederholt, bis alle Elemente aus "Ceiling" nach "Floor" verschoben sind.

5.4.4 Kompaktierung von zwei Zellen

Für die Platzierung der Zellen wird der in [Wolf96, Wolf97, Wolf99a, Wolf99b] entworfene eindimensionale Kompaktierer verwendet. In jedem Kompaktierungsschritt sind genau zwei Zellen beteiligt. Jede der Zellen wird durch ein Kantenbild dargestellt. Der minimale Abstand zwischen den jeweiligen beiden Zellen wird mittels eines Restriktionsgraphen berechnet. Daraus ergibt sich eine Verschiebung, die auf die kompaktierte Zelle angewendet werden muss, damit sie in demselben

Koordinatensystem wie die Zielzelle dargestellt werden kann. Die Kompaktierung erfolgt nur in einer Richtung. Daher vereinfacht sich der Restriktionsgraph, was eine schnelle Berechnung des Verschiebungsvektors möglich macht. Der Vorteil liegt darin, dass das Ergebnis der Kompaktierung für den Schaltungsdesigner vorhersagbar bleibt.

Es ist in jeder Zelle ein Bezugspunkt definiert, hinsichtlich dessen die Lage der Layoutkanten definiert ist. Für das Zielobjekt sei das der Punkt Z und für die zu kompaktierende Zelle der Punkt K. Es wird ein Restriktionsgraph erstellt, der zum einen je einen Knoten für die Punkte Z und K enthält, zum anderen für jede Kante des Zielobjekts und des zu kompaktierenden Objekts je einen Knoten zugewiesen bekommt. Ausgehend vom Knoten Z wird zu jedem Knoten, der eine Layoutkante der Zielzelle darstellt, eine Kante erzeugt. Entsprechend wird dem Graphen von jedem Knoten der Layoutkanten der zu kompaktierenden Zelle zum Knoten K eine Kante hinzugefügt. Die Gewichte dieser Kanten entsprechen den Abständen zwischen dem Punkt S und den jeweiligen Layoutkanten der Zielzelle bzw. der Distanz zwischen dem Punkt K und den Layoutkanten des kompaktierten Objekts.

Es bleibt die Verbindungskanten zwischen den Knoten der Layoutkanten der beiden Zellen zu erstellen. Dazu wird genau dann eine Verbindung generiert, wenn ein Minimalabstand zwischen zwei Kanten des Zielobjekts und des zu kompaktierenden Objekts existiert. Das Gewicht der Kante ist der technologiebedingte, definierte Minimalabstand.

Zur Berechnung des minimalen Verschiebungsvektors ist im resultierenden Restriktionsgraphen der längste Pfad von Z nach K zu bestimmen. Die Pfadlänge setzt sich aus der Summe der Kantengewichte entlang des Pfades von Z nach K zusammen. Konstruktionsbedingt beinhaltet jeder Pfad genau drei Kanten. Der längste Pfad berechnet sich gemäß Gl. (5-5), wobei lediglich diejenigen Kanten i und j berücksichtigt werden, für die eine Mindestdistanz definiert ist.

$$L_{\max,\mathrm{Path}} = \max\{\delta_{Z,i} + \delta_{ij} + \delta_{K,j}\} \text{ mit } 1 \leq i \leq |E_Z| \text{ und } 1 \leq j \leq |E_K|, \tag{5-5}$$

wobei E_Z die Menge der Layoutkanten des Zielobjekts, E_K die Menge der Layoutkanten des kompaktierten Objekts, $\delta_{Z,i}$ das Gewicht der Kanten vom Knoten Z zum i-ten Knoten der entsprechenden Layoutkante der Zielzelle, $\delta_{K,j}$ das Gewicht der Kante vom Knoten K zum j-ten Knoten der entsprechenden Layoutkante der kompaktierten Zelle und δ_{ij} das Gewicht der Kante zwischen den Knoten i und j ist, falls diese existiert.

Aus dem längsten Pfad kann der Verschiebungsvektor, mit dem das zu kompaktierende Objekt in das Koordinatensystem des Zielobjekts abgebildet wird, mit der Gl. (5-6) berechnet werden.

$$\Delta = L_{\max,\mathrm{Path}} - (K - Z), \tag{5-6}$$

dabei sind K und Z die Bezugspunkte der zu kompaktierenden Zelle bzw. der Zielzelle. Für eine Kompaktierung in Richtung Westen bzw. Süden müssen die geometrischen Elemente des kompaktierten Objekts um Δ entlang der x-Achse bzw. y-Achse verschoben werden. Entsprechend wird bei einer Kompaktierung in Richtung Osten bzw. Norden eine Translation des kompaktierten Objekts um $-\Delta$ ausgeführt.

Im vorhergehenden Kap. 5.4.3 ist das Auftreten von Zyklen und die damit verbundenen Problem in einem Restriktionsgraphen kurz beschrieben worden. Bei dem hier vorgestellten konstruktiven Ansatz können keine Zyklen auftreten, da eine Zelle immer an eine existierende Struktur mit

minimalem Abstand platziert wird und eine Definition von den Maximalabständen bislang nicht erforderlich ist [Wolf99b]. Nach der Kompaktierung werden die Layoutkanten der beiden Zellen vereinigt, wobei nur die äußeren Kanten jedes Layers gespeichert werden. Die resultierenden Kanten und die beiden Zellen werden in einer neuen Zelle zusammengefasst.

5.5 Aufbau eines binären Platzierungsbaums

In Kap. 5.3 ist der Ablauf der Erstellung eines Verbindungsgraphen $G_{conn}= (V_{conn}, E_{conn})$ auf der Basis einer Menge von Modulen und einer Netzliste beschrieben worden. Nachfolgend werden die Schritte erläutert, um aus diesem Graphen ein binären Platzierungsbaum zu erhalten.

Der in dieser Arbeit verwendete Kompaktierer zum Aufbau des Layouts verarbeitet stets zwei Objekte zurzeit. Aus diesem Grund werden die Verbindungen zwischen den Modulen untersucht, zwei Module ausgewählt und zu einem neuen Modul vereinigt. Der Vorgang lässt sich in Form eines Binärbaumes darstellen. Während der Auswertung bleibt für je zwei Module die Kompaktierungsrichtung, Orientierung, Ausrichtung zueinander sowie die Topologie, falls mehrere für ein Modul vorhanden sind, zu bestimmen. Mit Hilfe des binären Platzierungsbaumes wird festgelegt, wann welches Modul platziert werden soll bzw. welche beiden Module ein neues Modul bilden. Es entsteht auf diese Weise eine Platzierungshierarchie. Die Vorgehensweise ermöglicht es ebenfalls, den Platzierungs- und Verdrahtungsvorgang nur für Teile der Schaltung durchführen und die Teillösungen für einen späteren Ablauf zu verwenden. Ebenso können Layouts durch den Schaltungsdesigner angefertigt bzw. verändert werden und anschließend der Platzierungs- und Verdrahtungsablauf mit diesen Zellen erneut gestartet werden.

Unter Verwendung des Verbindungsgraphen ist ein Binärbaum zu erstellen, der die Reihenfolge der zu kompaktierenden Module beschreibt. Der Verbindungsgraphen muss die Eigenschaft aufweisen, dass es sich um einen zusammenhängenden Graphen handelt. Mit anderen Worten jeder Knoten des Graphen ist von jedem anderen Knoten über einen Weg aus erreichbar. Der nachfolgende Algorithmus 5-3 beschreibt die Schritte, die zur Erzeugung des Baumes notwendig sind.

Es wird über alle Knoten des Verbindungsgraphen solange iteriert, bis die Knotenmenge lediglich aus einem Knoten besteht. Zu Beginn der Schleife wird die Kante des Verbindungsgraphen mit dem höchsten Gewicht bestimmt. Es können mehrere Verbindungen mit dem gleichen Verbindungsgewicht auftreten. Diese werden in einer temporären Liste gespeichert. Falls eine Kante mit einem höheren Gewicht auftritt, werden alle Elemente aus der Liste durch diese Kante ersetzt und die Suche weiter fortgesetzt. Für den Fall, dass mehrere Kanten gefunden wurden, erfolgt im nächsten Schritt der Versuch die Anzahl weiter zu reduzieren. Dazu wird jede der ermittelten Kanten mit den anderen Kanten der temporären Liste verglichen. Zunächst wird nach einem gemeinsamen Knoten zwischen je zwei Kanten e_i und e_j gesucht. Existiert ein derartiger Knoten $v \in V_{conn}$ wird bei jeder der beiden Kanten der zu v_c adjazente Knoten v_x mit $x \in \{1, 2\}$ ermittelt. Für jeden dieser beiden Knoten werden die Gewichte der anliegenden Kanten aufsummiert. Anschließend wird diejenige Kante zwischen v_c und dem entsprechenden Knoten v_x entfernt, die die geringere Gewichtssumme aufweist. Bei einer Übereinstimmung der Gewichtssummen wird eine der beiden Kanten per Zufall ausgewählt und aus der Liste gelöscht. Es verbleibt eine Liste mit Kanten, die keine gemeinsamen Knoten aufweisen.

Algorithmus 5-3: Erstellen des Platzierungsbaums

```
solange |Vconn|>1
   // suche nach Kanten, die das höchste Verbindungsgewicht aufweisen
   e_Best:=∅; L_Best:=∅;
   für i:=1 bis |E_conn|
      wenn ((e_Best=∅) oder (Weight(e_i)>Weight(e_Best))) dann
         L_Best:={e_i};
         e_Best:=e_i;
      sonst wenn (Weight(e_i)=Weight(e_Best)) dann
         L_Best:=L_Best∪{e_i};
      wenn_ende
   für_ende
   // prüfen, ob mehrere Kanten gefunden wurden und die resultierende Menge
   // ggf. weiter reduzieren
   wenn (|L_Best|>1) dann
      für i:=1 bis |L_Best|-1
         für j:=1 bis |L_Best|
            v_c:=CommonNode(e_i,e_j);
            wenn (v_c≠∅) dann
               w_i:=CalcWeightSum(v_c,e_i);
               w_j:=CalcWeightSum(w_c,e_j);
               r:=Random(0,1);
               wenn ((w_i>w_j) oder ((w_i=w_j) und (r≥0.5))) dann
                  entferne e_j aus L_Best;
               sonst
                  entferne e_i aus L_Best;
               wenn_ende
            wenn_ende
         für_ende
      für_ende
   wenn_ende
   // Verschmelzung der Knoten der ermittelten Kanten durchführen
   für i:=1 bis |L_Best|
      Knoten v_neu erstellen, der die Inhalte der beiden Knoten der Kante e_Best,i
      übergeben bekommt;
      v_neu der Knotenmenge V_conn hinzufügen;
      übergebe alle Kanten der Knoten der Kante e_Best,i an den Knoten v_neu;
      entferne die Knoten der Kante e_Best,i aus der Knotenmenge V_conn;
   für_ende
solange_ende
```

In einem linearen Durchlauf über diese resultierende Liste werden für jede Kante die beiden Knoten, die mit einander verbunden sind, zu einem neuen Knoten verschmolzen und in der Knotenmenge des Verbindungsgraphen ersetzt. Der Verschmelzungsvorgang besteht darin, einen neuen Knoten v'_{ij} zu erstellen, der die Dateninhalte und Verbindungen der beiden Knoten v_i und v_j zugewiesen bekommt. Anschließend wird v'_{ij} der Knotenmenge des Verbindungsgraphen hinzugefügt und die beiden Knoten v_i und v_j gelöscht.

Die Knoten des Verbindungsgraphen verwenden zur Speicherung der Module jeweils einen binären Platzierungsbaum. Zu Beginn des gesamten Verfahrens besteht jeder der Binärbäume lediglich aus der Wurzel, die auf das entsprechende Modul verweist. Während der Vereinigung zweier Knoten v_i und v_j des Verbindungsgraphen enthält der neue Knoten zunächst einen leeren Platzierungsbaum, bestehend aus der Wurzel, die auf die leere Menge zeigt. Die Wurzel wird als linker Kindknoten des Platzierungsbaums des Knotens v_i übergeben. Entsprechend wird für den rechten Nachfolger des Platzierungsbaums des Knotens v_j verwendet. In Abb. 5-10 ist ein Beispiel für den Aufbau des Platzierungsbaums während des Verschmelzungsvorgangs gegeben.

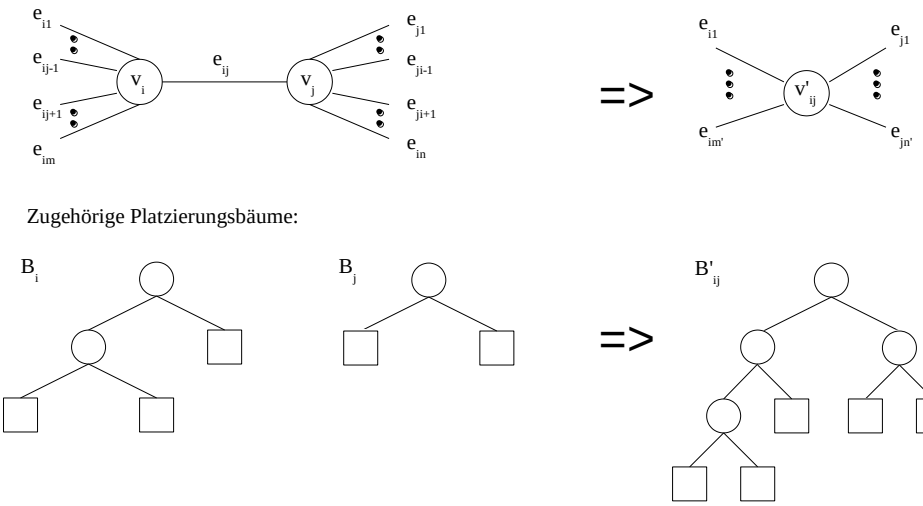


Abbildung 5-10: Verschmelzung zweier Knoten des Verbindungsgraphen und die Kombination der korrespondierenden Platzierungsbäume

Es sind die Knoten v_i und v_j dargestellt, wobei v_i über m Kanten verfügt und v_j über n Verbindungen. Die e_{ij} Kanten repräsentieren die Verbindung zwischen v_i und v_j. Neben den Knoten und Verbindungskanten sind die zu v_i und v_j zugehörigen Platzierungsbäume B_i und B_j gezeigt. Die Blätter der beiden Bäume verweisen auf die jeweiligen Module. Nach der Vereinigung der beiden Knoten v_i und v_j geht daraus der Knoten v'_{ij} hervor. Entsprechend wird für diesen Knoten ein neuer Platzierungsbaum mit einer leeren Wurzel erstellt. Diesem Baum werden die Platzierungsbäume B_i und B_j als linker bzw. rechter Kindknoten übergeben, so dass der in der Abbildung gezeigte Platzierungsbaum B'_{ij} entsteht. Für den neuen Knoten sind die Kanten des Verbindungsgraphen zu prüfen. Es ist möglich, dass v_i und v_j gemeinsame adjazente Knoten besitzen. Aus diesem Grund sind alle Kanten von v_i und v_j zu untersuchen und demgemäß anzupassen. Der Ablauf der Aktualisierung der Verbindungskanten ist in Algorithmus 5-4 dargestellt.

Algorithmus 5-4: Aktualisieren der Kanten des Verbindungsgraphen

```
für k:=1 bis m_i
   v_o:=opposite(v_i,e_ik);
   erstellen eine neue Kante e‘, die die Knoten v_o und v’_ij verbindet;
   füge e‘ der Kantenmenge E_conn hinzu;
   entferne e_ik aus E_conn;
für_ende
für k:=1 bis m_j
   v_o:=opposite(v_j,e_ik);
   wenn (v_o≠v_i) dann
      wenn (AreAdjacent(v’_ij,v_o)=TRUE) dann
         suche diejenige Kante e‘, die v_o und v’_ij verbindet;
         weight(e‘):=weight(e‘)+weight(e_jk);
      sonst
         erstelle eine neue Kante e‘, die die Knoten v_o und v’_ij verbindet;
         füge e‘ der Kantenmenge E_conn hinzu;
      wenn_ende
   wenn_ende
für_ende
lösche alle Kanten aus E_conn, die auf den Knoten v_j verweisen;
```

Es wird in der ersten Schleife über alle m_i Kanten des Knotens v_i iteriert. In jedem Schritt wird der zu v_i benachbarte Knoten v_o mit Hilfe der aktuellen Kante e_{ik} bestimmt. Anschließend erfolgt die Erstellung einer neuen Kante e', die die beiden Knoten v'_{ij} und v_o miteinander verbindet. Der neuen

Kante e' wird das Gewicht von e_{ik} zugewiesen. Im nächsten Schritt werden die Kante e' der Kantenmenge E_{conn} hinzugefügt, während die Kante e_{ik} gelöscht wird. Es bleiben die Kanten, die auf den Knoten v_j verweisen, zu aktualisieren. Dazu erfolgt ein Durchlauf über alle m_j Kanten des Knotens v_j. Zu Beginn der Iteration ist, wie in der vorhergehenden Schleife, der zu v_j adjazente Knoten v_o zu ermitteln. Eine Kantenanpassung ist nur dann notwendig, wenn der benachbarte Knoten nicht dem Knoten v_i entspricht, da alle Verbindungen zwischen v_i und v_j infolge der Verschmelzung aufgelöst werden. Es bleibt zu prüfen, ob bereits eine Verbindung zwischen dem neuen Knoten v'_{ij} und v_o existiert. In diesem Fall wird das Gewicht der Verbindungskante von v_j und v_o auf das Gewicht der Kante zwischen v'_{ij} und v_o aufsummiert. Andernfalls wird eine neue Kante erstellt, die die Knoten v'_{ij} und v_o verbindet und der Kantenmenge E_{conn} hinzugefügt. Nach dem Schleifendurchlauf werden alle Kanten, die auf den Knoten v_j verweisen, aus der Kantenmenge des Verbindungsgraphen entfernt.

5.6 Auswertung des binären Platzierungsbaums

Im vorhergehenden Abschnitt wurde gezeigt, wie mit Hilfe des Verbindungsgraphen ein binärer Platzierungsbaum erstellt werden kann, der die Reihenfolge und den Zeitpunkt der zu platzierenden Module festlegt. In einer Post-Order-Traversierung über den Platzierungsbaum erfolgt die Konstruktion der Platzierung, d.h. es wird zuerst der linke Teilbaum, dann der rechte durchlaufen und anschließend die Wurzel betrachtet. Während des Ablaufs werden stets zwei Module mit Hilfe eines Kompaktierers zu einem neuen Modul vereinigt. Das Ergebnis liegt nach Abschluss des Verfahrens in Form eines Moduls vor, welches neben den zu platzierenden Zellen die vollständige Verdrahtung enthält. Die Blätter des Platzierungsbaums enthalten die tatsächlichen Module, während in den inneren Knoten die jeweiligen Platzierungsergebnisse zwischengespeichert werden. Für den Baumdurchlauf wird eine rekursive Funktion verwendet, deren Aufbau dem Algorithmus 5-5 zu entnehmen ist.

Algorithmus 5-5: Rekursive Funktion zur Auswertung des Platzierungsbaums

```
Funktion EvaluatePlacementTree(Knoten v): Modul
Beginn
   wenn (IsLeaf(v)=TRUE) dann
      gib Modul zurück, welches in v gespeichert ist;
   sonst
      ModA:=EvaluatePlacementTree(Left(v));
      ModB:=EvaluatePlacementTree(Right(v));
      berechne mittels eines Entscheidungsbaums die voraussichtlich optimale
      Ausrichtung und Kompaktierungsrichtung zwischen ModA und ModB;
      erstelle ein neues Modul ModN;
      füge ModA und ModB dem Module ModN hinzu;
      verwende ModA als Zielobjekt, ModB als zu kompaktierendes Objekt;
      führe Kompaktierung zwischen ModA und ModB durch;
      berechne Verdrahtung zwischen ModA und ModB;
      füge resultierende Leitungen dem Modul ModN hinzu;
      gib ModB zurück;
   wenn_ende
Ende
```

Zu Beginn des Verfahrens wird der Methode die Wurzel des binären Platzierungsbaums übergeben. Handelt es sich beim aktuellen Knoten um ein Blatt des Baums, d.h. besitzt der Knoten keine weiteren Nachfolger, so wird das im Knoten gespeicherte Modul zurückgegeben. Andernfalls erfolgen die rekursiven Aufrufe der Funktion für die linken bzw. rechten Nachfolger des aktiven Knoten. Aus diesen gehen jeweils zwei Module hervor, für die die Kompaktierungsrichtung, die Ausrichtung und die Topologie zu bestimmen sind. Mit Hilfe eines Entscheidungsbaums werden diese benötigten Werte ermittelt. Anschließend wird die Kompaktierung der beiden Module ausgeführt

und das Ergebnis in einem neuen Modul gespeichert. Es wird ebenfalls die Verdrahtung zwischen den beiden kompaktierten Modulen berechnet und in dem neuen Modul abgelegt. Die Funktion schließt mit der Rückgabe des neuen Moduls.

Die nachfolgenden Abschnitte behandeln die Grundlagen zum Entscheidungsbaum, dessen Implementation und Auswertung (siehe Kap. 5.6.1 und Kap. 5.6.2). Die Darstellung einer Platzierungslösung des Entscheidungsbaums erfolgt mittels eines Schnittbaums. Die Konstruktion dieser Datenstruktur sowie die Schritte um daraus eine zu bewertende Platzierung zu erhalten sind ebenfalls Bestandteil der nachstehenden Unterkapitel (siehe Kap. 5.6.3). Für die Auswahl einer Lösung aus den möglichen Ergebnissen des Entscheidungsbaums werden unterschiedliche Entscheidungsregeln bereitgestellt. Diese sind Gegenstand des Kap. 5.6.4.

5.6.1 Grundlagen Entscheidungsbaum

Der Entscheidungsbaum stellt eine Analyse- und Entscheidungsmethode dar, mit der die beste Strategie aus einer Anzahl möglicher relevanter Strategien festgestellt werden kann. Sie dienen zur automatischen Klassifikation von Datenobjekten und damit zur Lösung von Entscheidungsproblemen [Brei84, Frie77, Quin86]. Ein Entscheidungsbaum besteht immer aus einer Wurzel, beliebig vielen inneren Knoten, den sogenannten Entscheidungsknoten, sowie mindestens zwei Blättern. Jeder Knoten repräsentiert eine logische Regel, jedes Blatt eine Antwort auf das Entscheidungsproblem. Entscheidungsabfragen beginnen bei der Wurzel und verzweigen sich durch die darauf folgenden Entscheidungsmöglichkeiten permanent weiter. Auf diese Weise entsteht ein Wurzelnetzwerk, welches dabei hilft, vorgegebene Ziele mit optimalem Erwartungswert bzw. Wahrscheinlichkeit bezüglich des tatsächlichen Eintreffens zu erreichen. Es ist eine Methode, die sich verhältnismäßig einfach implementieren lässt. Die Entscheidungswege sind in der Regel leicht verständlich und ihr gesamter Berechnungsvorgang ist leicht nachvollziehbar. Jeder Pfad von der Wurzel zu einem Blattknoten entspricht einer logischen Formel in der Form einer "wenn-dann"-Regel.

Ein Entscheidungsbaum, basierend auf einer Menge von symbolischen bzw. diskreten Merkmalen, ist ein Klassifikationsbaum, während ein Entscheidungsbaum mit numerischen bzw. kontinuierlichen Merkmalen als Regressionsbaum bezeichnet wird. Diese Form der Bäume wird in einer Vielzahl an Applikationen eingesetzt, wie beispielsweise in der Mustererkennung, Sprachverarbeitung oder der Approximation numerischer Funktionen. Ein großes Einsatzgebiet von Entscheidungsbäumen betrifft das maschinelle Lernen und der damit verbundenen Klassifikation von Daten. Zur Berechnung der Entscheidungsbäume werden verschiedene Algorithmen verwendet. Der Aufbau des Entscheidungsbaumes erfolgt bei Einsatz innerhalb des maschinellen Lernens häufig mit dem C4.5-Algorithmus [Quin93] bzw. mit dem CART-Verfahren [Brei84].

Beim CART-Algorithmus können lediglich Binärbäume erzeugt werden. Die zentrale Aufgabe dieses Verfahrens ist damit das Finden einer optimalen binären Trennung der Daten aus denen der Baum aufgebaut wird. Jeder Datensatz wird durch ein Attribut beschrieben. Zu jedem Attribut wird ein Schwellwert ermittelt. Je höher der Informationsgehalt des Attributes und der daraus resultierende Schwellwert sind, desto weiter oben befindet sich der zum Attribut gehörige Knoten innerhalb des Entscheidungsbaumes. Der Informationsgehalt eines Attributes ist hoch, wenn durch den Schwellwert ein Attribut ausgewählt wird, mit welchem eine hohe Trefferquote bei der Klassifikation erzielt werden kann.

Das C4.5-Verfahren weist eine vergleichbare Vorgehensweise wir der CART-Algorithmus auf. Es ist jedoch nicht eine binäre Aufteilung notwendig. Damit ergibt sich ein beliebiger Baum mit einer

größeren Breite und einer geringeren Tiefe im Vergleich zum resultierenden Baum des CART-Verfahrens. Alle Entscheidungsbäume haben gemeinsam, dass diese "top-down" erstellt werden [Fire77, Quin86].

Die mögliche Größe von Entscheidungsbäumen kann sich negativ auswirken. Aus diesem Grund sind Pruning-Methoden, d.h. das bewusste Ignorieren von Informationen, entwickelt worden, um die Ausmaße der Entscheidungsbäume zu beschränken. Wenn dem Algorithmus infolge bereits zur Verfügung stehender Daten bekannt ist, dass das gesuchte Objekt bzw. die gewünschte Strategie nicht in einem Teilbaum enthalten ist, so kann dieser im weiteren Verlauf verworfen werden. Es kann beispielsweise auch die maximale Tiefe bzw. die Mindestanzahl der Daten je Knoten eingegrenzt werden, um die Größe des Entscheidungsbaumes zu reduzieren. Der große Vorteil der Entscheidungsbäume liegt darin, dass diese gut erklär- und nachvollziehbar sind.

5.6.2 Implementierung und Auswertung des Entscheidungsbaums

Das Prinzip des Entscheidungsbaums wird genutzt, um die optimale Anordnung zweier zu kompaktierender Module zu ermitteln. Es werden die Auswirkungen der möglichen Kompaktierungsrichtungen, Orientierungen der Module sowie unterschiedlichen Topologien auf die nachfolgenden Module untersucht. Für jede der Kombinationen wird eine Platzierung errechnet, deren Kosten bestimmt und mit Hilfe einer geeigneten Entscheidungsregel die kostengünstigste ausgewählt. Diese Auswahl beinhaltet neben der Kompaktierungsrichtung die jeweilige Topologie der beiden Module sowie eine mögliche Transformation. Der Ablauf für die Berechnung einer optimalen Anordnung ist in Algorithmus 5-6 gezeigt. Ohne Beschränkung der Allgemeinheit bezeichnet A das Zielmodul und B das zu kompaktierende Modul.

Algorithmus 5-6: Berechnung einer optimalen Anordnung von zwei Modulen A und B

```
List_cost:=∅;
List_Modules:=FindNextModules(A,B);
für i:=1 bis #Topologies(A)
   List_dir:=AllowedCompactionDirs(B);
   für j:=1 bis #Topologies(B)
      für k:=1 bis |List_dir|
         m_A:=Topology(i,A);
         m_B:=Topology(j,B);
         List_tmp:=CreatePlacementCost(m_A,m_B,List_dir[k],List_Modules);
         List_cost:=List_cost∪{{m_A,m_B,List_dir[k],List_tmp}};
         wenn (IsRotationAllowed(B)=TRUE) dann
            m_B:=Transform(Topology(j,B));
            List_tmp:=CreatePlacementCost(m_A,m_B,List_dir[k],List_Modules);
            List_cost:=List_cost∪{{m_A,m_B,List_dir[k],List_tmp}};
         wenn_ende
      für_ende
   für_ende
für_ende
mittels einer Entscheidungsregel aus List_cost die günstigste Anordnung für A und
B auswählen;
```

Vor der Iteration über die verschiedenen Topologien des Moduls A sind zuerst die noch zu platzierenden Module zu ermitteln. Dazu wird der binäre Platzierungsbaum einer Post-Order-Traversierung unterzogen. Ausgehend von der Wurzel wird der linke, dann der rechte Teilbaum durchlaufen und anschließend die Wurzel betrachtet. Während des Durchlaufs werden die Module, die von den Blättern referenziert werden, in einer linearen Liste $List_{Modules}$ nacheinander abgelegt. Aus dieser Liste werden die Einträge, beginnend am Kopf der Liste, solange entfernt, bis die Liste leer

ist oder die gelöschten Einträge die Module A und B beinhalten. Die resultierende Menge beinhaltet alle Module, die noch zu platzieren und verdrahten sind.

Für das Modul B werden alle erlaubten Kompaktierungsrichtungen ermittelt. Wenn es sich bei dem vorliegenden Modul um einen Zellenblock, bestehend aus NMOS- bzw. npn-Transistoren handelt, und durch den Schaltungsdesigner die Lage des V_{SS}- bzw. Ground-Zeile vorgegeben wurde, wird die entsprechende Kompaktierungsrichtung für dieses Modul ausgeschlossen. Befindet sich die V_{SS}-Schiene am südlichen Rand der Schaltung, dann wird für die n-Typ Module die Kompaktierungsrichtung Süd nicht zugelassen. Die Vorgehensweise erfolgt analog für die Lagebeziehung zwischen PMOS- und pnp-Modulen und der V_{DD}-Schiene. Auf diese Weise werden die Zellen entsprechend ihres Typs in die Nähe zu den V_{DD}- bzw. V_{SS}-Anschlüssen platziert.

In einer Iteration über alle erlaubten Kompaktierungsrichtungen und Topologien des Moduls B werden für jede der beiden Layoutgeometrien von A und B die Platzierungskosten berechnet. Dabei werden die Kompaktierungsrichtungen und die nachfolgenden Module mit berücksichtigt. Es wird ein Entscheidungsbaum, basierend auf den gegebenen Daten, erstellt. Für jede mögliche Platzierungsanordnung werden die Kosten berechnet und in einer Liste gespeichert. Ebenso werden die Zwischenergebnisse in die Ergebnisauswertung mit einbezogen. Es wird ein Datentupel, bestehend aus den jeweiligen Topologien von A und B, der aktuellen Kompaktierungsrichtung, sowie der Liste der möglichen Kosten erstellt. Falls für das Modul B eine Transformation erlaubt ist, werden die genannten Schritte für eine transformatierte Topologie der Zelle B erneut durchgeführt.

Nach Abschluss der Iteration wird mit Hilfe einer durch den Schaltungsdesigner ausgewählten Regel die kostengünstigste Anordnung für die Module A und B ausgewählt. In der Wahl der Lösung sind die zu verwendende Topologie von A und B, die Kompaktierungsrichtung sowie die Transformation von B enthalten.

Algorithmus 5-7: Beschreibung der Methode CreatePlacementCost

```
List_CS:={{m_A,m_B},compDir}};
wenn (|List_Modules|>SearchDepth) dann
   n:=SearchDepth;
sonst
   n:=|List_Modules|;
wenn_ende
für i:=1 bis n
   List_tmp:=∅;
   solange (List_CS≠∅)
      cs:=RemoveFirst(List_CS);
      List_tmp:=List_tmp∪CreateCompactionSequence(First(cs),Last(cs),List_Modules[i]);
   solange_ende
   List_CS:=List_tmp;
für_ende
List_cost:=∅;
für i:=1 bis |List_CS|
   B_S:=CreateSlicingTree(List_CS[i]);
   EvalSlicingTree(Root(B_S));
   List_cost:=List_cost∪CalcPlacementCost(Module(Root(B_S)));
für_ende
gib List_cost zurück;
```

Im Algorithmus 5-7 sind die Schritte zur Berechnung der Platzierungskosten gezeigt. Die Eingabeparameter der Methode setzen sich aus den beiden Layouttopologien der zu kompaktierenden Zellen A und B zusammen. Zusätzlich wird eine Liste der noch nachfolgenden

Module übergeben. Es erfolgt der Aufbau eines Entscheidungsbaumes. Die Vorgehensweise ist vergleichbar mit einem Breitendurchlauf in einer Baumstruktur. Den Startknoten bildet ein Datentupel, bestehend aus den Geometrien der beiden Module A und B sowie der zu verwendenden Kompaktierungsrichtung. Die Höhe des nachfolgend aufzubauenden Entscheidungsbaumes ist hauptsächlich abhängig von der Anzahl der nachfolgenden Module, die bei der Auswahl einer geeigneten Anordnung der beiden zu kompaktierenden Zellen berücksichtigt werden. Bei einer großen Menge zu platzierender Module kann die Anzahl der möglichen Kombinationen, die untersucht werden sollen, enorme Ausmaße annehmen, so dass hier die Möglichkeit einer Suchtiefenbegrenzung vorgesehen ist. Auf diese Weise wird zwar der Lösungsraum eingegrenzt, jedoch kann das Verfahren damit auch auf Systemen mit begrenzten Ressourcen im Hinblick auf Speicher und Rechenleistung eingesetzt werden. Es folgt eine Iteration über die noch zu platzierenden Module. In jedem Schleifendurchlauf wird eine temporäre, leere Liste erstellt. Die Umsetzung der Liste $List_{CS}$ erfolgt mit Hilfe einer Warteschlange. Es wird nach dem "First In"-"First Out"-Prinzip gearbeitet, d.h. es wird immer das Element aus der Warteschlange entnommen, welches in die Menge der vorhandenen Einträge als erstes abgelegt wurde. Solange $List_{CS}$ Objekte enthält, wird das erste Element der Liste entfernt. Mit Hilfe dieses Datenelement und des aktuellen Moduls werden die mögliche neuen Kompaktierungssequenzen errechnet und in der temporären Liste abgespeichert. Der Ablauf für die Bestimmung der möglichen Platzierungskombinationen ist dem Algorithmus 5-8 zu entnehmen. Jedes resultierende Datentupel enthält zwei Listen. Die erste Liste besteht aus einer Folge von Layouttopologien der Module. Es sind mindestens zwei Einträge in der Liste vorhanden. Die zweite Sequenz hingegen beschreibt eine Folge von Kompaktierungsrichtungen. Der erste Eintrag der Modulliste repräsentiert das erste Zielobjekt. Alle nachfolgenden Modullayouts stellen die zu kompaktierenden Objekte dar. Zu jedem dieser Layouts ist in der zweiten Liste die zu verwendende Kompaktierungsrichtung definiert. Damit enthält die zweite Liste stets genau einen Eintrag weniger als die Liste der Topologien.

Als Grundlage zur Berechnung der Platzierungskosten dienen der Flächenbedarf und das Seitenverhältnis des umschließenden Rechtecks einer Platzierung. Die Kosten für die Fläche und das Seitenverhältnis verweisen jedoch in unterschiedlich große Wertebereiche, so dass eine direkte Auswertung der Kosten nicht möglich ist. Aus diesem Grund erfolgt eine Abbildung der jeweiligen Kosten auf das Intervall [0,1]. Die Abbildung erfolgt mit Hilfe der Gauß Funktion um einen gegebenen Zielwert. Bei der Berechnung der Kosten für die Fläche wird als Erwartungswert die minimal mögliche Fläche als Erwartungswert verwendet. Dieser Wert ist gegeben durch die folgende Summe

$$A_{\text{Modules}} = \sum_{i=1}^{\#\text{Modules}} A_{\text{Module(i)}}, \tag{5-7}$$

d.h. es werden die Flächen der jeweiligen Module aufsummiert. Damit ergibt sich für die Kosten der Fläche einer Platzierung

$$f_A(x) = 1 - \left| e^{-\frac{1}{2}\cdot\left(\frac{x-A}{A}\right)^2} \right|, \tag{5-8}$$

wobei A den Erwartungswert in Form der Gl. (5-7) für die minimale Fläche beschreibt. Auf vergleichbare Weise werden die Kosten für das Seitenverhältnis auf das Intervall [0, 1] abgebildet. Es ist

$$f_\varphi(x) = 1 - \left| e^{-\frac{1}{2}\cdot\left(\frac{x-\varphi}{\varphi}\right)^2} \right|, \tag{5-9}$$

wobei φ einen vom Schaltungsdesigner vorgegebenen Wert größer Null darstellt. Es ist das zu erzielende Seitenverhältnis. Damit ergibt sich die Kostenfunktion zu

$$c(x_1, x_2) = \alpha_A \cdot f_A(x_1) + \alpha_\varphi \cdot f_\varphi(x_2), \tag{5-10}$$

bestehend aus den gewichteten Kosten der Flächen und des Seitenverhältnisses der Platzierung.

In Algorithmus 5-8 sind die Schritte zur Erweiterung der Listen für ein weiteres Modul dargestellt. Für das aktuell zu betrachtende Modul m werden alle zulässigen Kompaktierungsrichtungen bestimmt. In jedem Durchlauf über die verschiedenen Topologien des Moduls werden die zuvor ermittelten Kompaktierungsrichtungen ausgewertet. Der Methode sind als Parameter die Liste der bisherigen Module ($List_M$) und Kompaktierungsrichtungen ($List_{cd}$) übergeben worden. Von diesen Listen werden in jedem Schritt Kopien angefertigt. An das Ende der neuen Liste wird jeweils die aktuelle Modulgeometrie bzw. die Verschiebungsrichtung angefügt. Beide Listen werden in einem Datentupel zusammengefasst und in einer Ergebnisliste abgespeichert. Anschließend erfolgt eine Überprüfung, inwieweit eine Rotation für das Modul m erlaubt ist, die eine orthogonale Ausrichtung zur Folge hätte. In diesem Fall wird ein weiteres Datentupel mit der geänderten Topologie der Ergebnisliste hinzugefügt.

Algorithmus 5-8: Beschreibung der Methode CreateCompactionSequence

```
List_dir:=AllowedCompactionDirs(m);
List_result:=∅;
für i:=1 bis #Topologies(m)
   für j:=1 bis |List_dir|
      erstelle Kopie List'_M von List_M;
      erstelle Kopie List'_cd von List_cd;
      List'_M:=List'_M∪Topology(m);
      List'_cd:=List'_cd∪List_dir[j];
      List_result:=List_result∪{List'_M,List'_cd};
      wenn (IsRotationAllowed(m)=TRUE) dann
         erstelle Kopie List''_M von List_M;
         List''_M:=List''_M∪Rotate(Topology(m));
         List_result:=List_result∪{List''_M,List'_cd};
      wenn_ende
   für_ende
für_ende
```

5.6.3 Aufbau und Auswertung des Schnittbaums

Ein Schnittbaum ist eine Möglichkeit der Modellierung der Layoutfläche. Es handelt sich dabei um einen vereinfachten Polargraphen und ist die am häufigsten verwendete Datenstruktur für das Floorplanning. Zu einem Floorplan existiert jeweils ein horizontaler und vertikaler Polargraph. Ein Knoten eines horizontalen polaren Graphen repräsentiert einen vertikalen Kanal, entsprechend wird ein horizontaler Kanal mittels eines Knoten des vertikalen Polargraphen dargestellt. Die Kanten der Graphen entsprechen den Modulen, wobei die Gewichte der Breite bzw. Höhe des jeweiligen Moduls entsprechen. Aus den längsten Pfaden der beiden Graphen können die Abmessungen des Layouts ermittelt werden. Bei einem Schnittbaum repräsentieren die inneren Knoten sowohl die vertikalen als auch die horizontalen Schnittlinien. Die Blätter entsprechen den einzelnen Topologien der Module. Die Baumstruktur schränkt zwar den Problemraum ein, da nur Platzierungen darstellbar sind, die durch rekursive vollständige Schnitte durch die Layoutfläche entstehen. Andererseits liefert die Verwendung der hierarchischen Kompaktierung genau solch eine Darstellung des Layouts.

Aufbauend auf den Abfolgen der zu platzierenden Zelltopologien und den korrespondierenden Kompaktierungsrichtungen lässt sich daraus die Platzierung der Module berechnen. Dafür wird ein Schnittbaum verwendet, der zusätzliche Optimierungen der Modulanordnung ermöglicht. Die Schnittlinien verlaufen orthogonal zur Kompaktierungsrichtung. In Algorithmus 5-9 sind die Schritte zum Aufbau eines Schnittbaums aus den Kompaktierungssequenzen zusammenfassend dargestellt. Das Datentupel cs bildet den Eingabeparameter für die Methode. Es setzt sich aus der Menge der zu platzierenden Modulgeometrien und den zu verwendenden Kompaktierungsrichtungen zusammen.

Zunächst wird ein binärer Schnittbaum, bestehend aus einem Wurzelknoten, erstellt. Diesem Knoten wird die erste Modultopologie zugewiesen. In einem Schleifendurchlauf über die verbleibenden Module wird der Schnittbaum aufgebaut. Für jedes Modul ist die Erstellung von zwei neuen Knoten notwendig. Einer der Knoten bildet die neue Wurzel des Schnittbaums. Aufgrund der Tatsache dass die Wurzel ab jetzt ein innerer Knoten ist, wird dieser die aktuelle Kompaktierungsrichtung zugewiesen. Den linken Nachfolger bildet der alte Wurzelknoten, während der rechte Nachfolger der zweite neu erzeugte Knoten ist. Dieser Knoten verweist auf die aktuelle Modulgeometrie. Falls der Schnittbaum mindestens drei Module enthält, wird die Möglichkeit einer Optimierung überprüft und ggf. ausgeführt. Nach Ablauf des Verfahrens resultiert ein Schnittbaum, mit dessen Hilfe in einer Post-Order-Traversierung die Platzierung generiert werden kann.

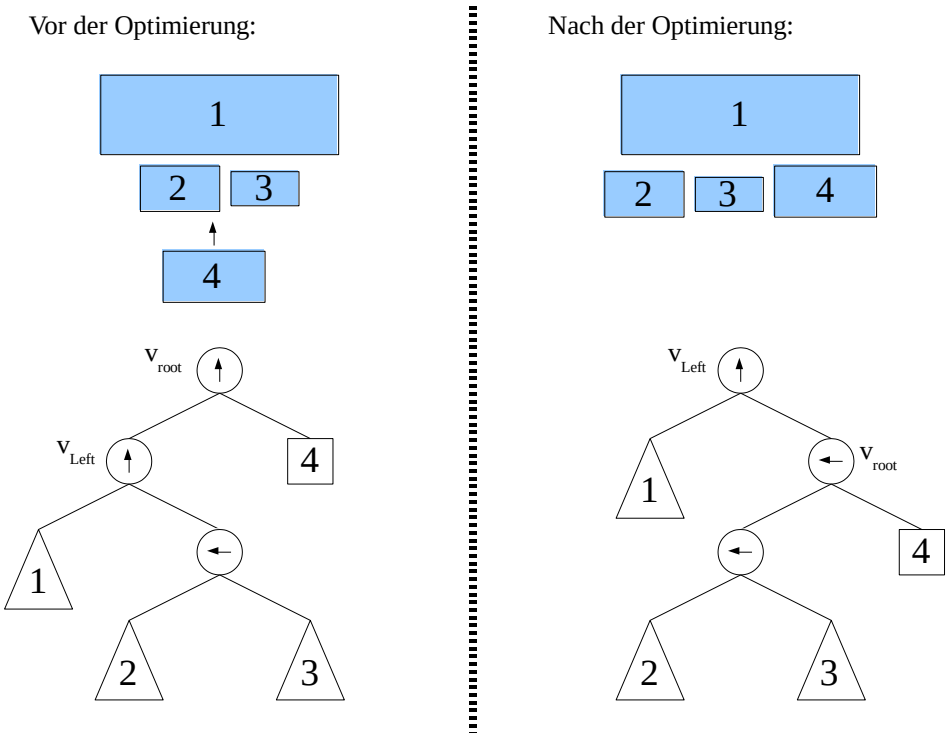


Abbildung 5-11: Optimierung des Schnittbaums bei einer Übereinstimmung der Kompaktierungsrichtungen zwischen der Wurzel und dem linken Nachfolger

Hinsichtlich der Optimierung ergeben sich drei zu untersuchende Fälle, die im Folgenden anhand von einfachen Beispielen kurz erläutert werden. Der Schnittbaum wird derart aufgebaut, dass stets der linke Kindknoten der neuen Wurzel dem vorherigen Baum entspricht. Aus diesem Grund werden die Kompaktierungsrichtungen, die in der alten und der neuen Wurzel abgelegt sind, miteinander verglichen. Dabei können drei mögliche Beziehungen der Kompaktierungsrichtungen unterschieden werden. Es kann eine Übereinstimmung vorliegen. Dieser Fall ist für ein einfaches Beispiel in

Abbildung 5-11 dargestellt. An den Zellenblock, der aus dem Teilbaum 1 hervorgeht, werden mit der Kompaktierungsrichtung die Module, bestehend aus den Teilbäumen 2 und 3, herankompaktiert. Die Zellen 2 und 3 gehen selber aus einer westlichen Kompaktierung hervor. In dem mit v_{Left} bezeichneten Knoten des Schnittbaums, der die Anordnung der Teilbäume 1 bis 3 beschreibt, ist somit die Kompaktierungsrichtung Nord eingetragen.

Algorithmus 5-9: Aufbau eines Schnittbaumes

```
List_M:=First(cs);
List_cd:=Last(cs);
erstelle einen leeren Schnittbaum B_S;
v_Root:=neuer Knoten; Root(B_S):=v_Root;
Left(v_Root):=Right(v_Root):=∅;
Module(v_Root):=List_M[1];
für i:=2 bis |List_M|
   v_1:=v_Root;
   v_Root:=neuer Knoten; Root(B_S):=v_Root;
   v_2:=neuer Knoten;
   Left(v_Root):=v1; Right(v_Root):=v_2;
   Module(v_2):=List_M[i];
   Dir(v_Root):=List_cd[i-1];
   wenn (i≥3) dann
      OptimizeSlicingTree(B_S);
   wenn_ende
für_ende
gib B_S zurück;
```

Erfolgt das Hinzufügen des nächsten Moduls 4 ebenfalls mit einer nördlichen Kompaktierung, ergibt sich der in der linken Seite von Abbildung 5-11 gezeigte Schnittbaum. Falls die nachfolgende Ungleichung erfüllt ist, wird eine Umordnung des Schnittbaums durchgeführt. Es ist

$$w\big(\mathrm{Right}(v_{\mathrm{Root}})\big) + w\big(\mathrm{Right}(v_{\mathrm{Left}})\big) \leq (1+\delta)\cdot w\big(\mathrm{Left}(v_{\mathrm{Left}})\big), \tag{5-11}$$

wobei $\delta \in [0, 1]$ ein vom Benutzer definierter Wert ist. Es werden die Weite des neuen Moduls mit der Weite des Moduls, welches sich aus dem rechten Nachfolger des Knoten v_{Left} ergibt, aufsummiert. Wenn die Summe kleiner oder gleich der Weite, eventuell vergrößert durch den Skalar δ desjenigen Moduls ist, welches aus dem linken Nachfolger von v_{Left} hervorgeht, wird der Schnittbaum verändert. Dabei wird die neue Wurzel v_{Root} zum rechten Nachfolger von v_{Left}. Der rechte Kindknoten von v_{Left} wird zum linken Nachfolger von v_{Root} und v_{Left} bildet die Wurzel des Schnittbaums. Abschließend ist die Kompaktierungsrichtung, die in v_{Root} abgelegt ist, anzupassen. Die Änderung erfolgt in Abhängigkeit zwischen der Kompaktierungsrichtung von v_{Root} und der, die im ursprünglichen, rechten Nachfolger von v_{Left} abgelegt ist. Sofern die beiden Richtungen orthogonal zueinander verlaufen, erhält v_{Root} die im rechten Kindknoten von v_{Left} abgespeicherte Richtung. Andernfalls wird eine Richtung, die senkrecht zu der in v_{Root} verläuft, zufallsbasiert ausgewählt. Bezogen auf das Beispiel resultiert der in Abbildung 5-11 auf der rechten Seite dargestellte Schnittbaum und die zugehörige Anordnung der Module.

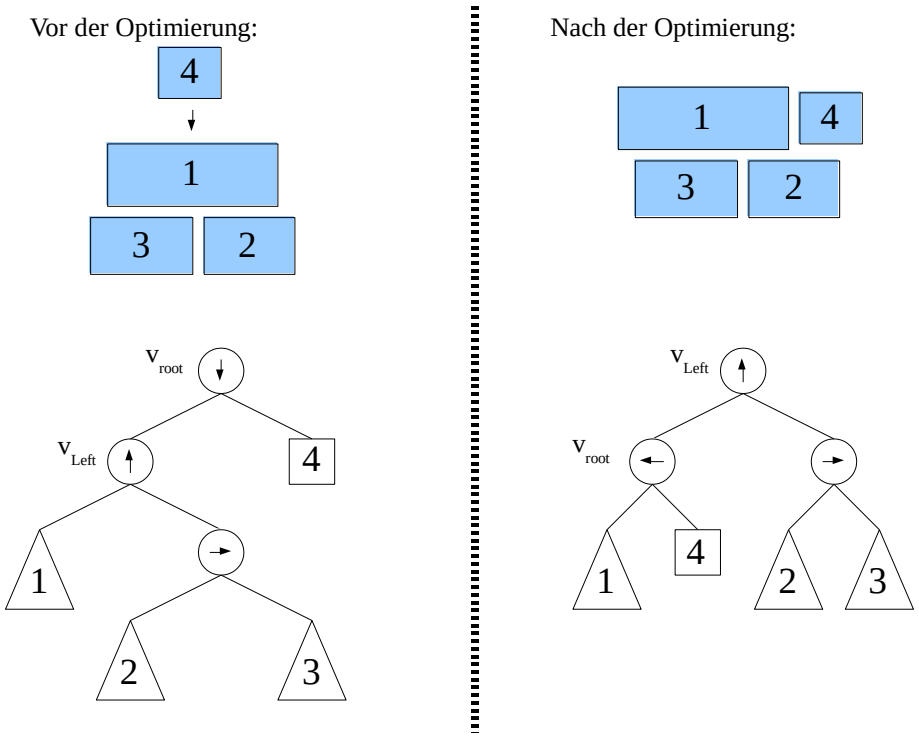


Abbildung 5-12: Optimierung des Schnittbaums bei gegenläufigen Kompaktierungsrichtungen zwischen der Wurzel und dem linken Nachfolger

Die nächste Möglichkeit der Lage der Kompaktierungsrichtungen der Wurzel und dessen linkem Nachfolger besteht darin, dass diese gegenläufig sind. Ein Beispiel für diesen Fall ist in Abbildung 5-12 gegeben. Die Anordnung der Module, ausgehend vom Knoten v_{Left}, stimmt mit der aus dem vorhergehenden Beispiel überein. Lediglich die Kompaktierung des neuen Moduls 4 erfolgt in südlicher Richtung an die bestehende Struktur. Die Prüfung auf eine Änderung des Schnittbaums verläuft analog zu den vorhergehenden, beschriebenen Schritten. Wenn die nachfolgende Ungleichung erfüllt ist, wird die Rekonstruktion durchgeführt

$$w\big(\mathrm{Right}(v_{\mathrm{Root}})\big) + w\big(\mathrm{Left}(v_{\mathrm{Left}})\big) \leq (1+\delta)\cdot w\big(\mathrm{Right}(v_{\mathrm{Left}})\big), \tag{5-12}$$

Im Gegensatz zur Ungleichung in Gl. (5-11) werden die Weite des Moduls des rechten Kindknoten von v_{Root} und die Weite des Moduls, welches aus dem linken Nachfolger von v_{Left} entsteht, aufsummiert und mit der Weite des Moduls aus dem rechten Nachfolger von v_{Left} verglichen. Wenn die Ungleichung erfüllt ist, erfolgt eine Umgestaltung des Schnittbaums. Der linke Nachfolger von v_{Left} wird zum linken Kindknoten von v_{Root}. Der Knoten v_{Root} wird im Austausch zum linken Nachfolger von v_{Left}. Der Knoten v_{Left} bildet damit die neue Wurzel des Schnittbaums.

Es bleibt der Fall zu betrachten, wenn die Kompaktierungsrichtungen der Wurzel und deren linken Nachfolger orthogonal zueinander verlaufen. Ein Beispiel für eine derartige Anordnung ist in Abbildung 5-13 gegeben. An die Module 1 und 2, die aus einer Kompaktierung in nördlicher Richtung hervorgehen, wird das Modul 3 mit der Kompaktierungsrichtung West herankompaktiert. Der resultierende Schnittbaum sowie die mögliche Modulanordnung ist der linken Seite der Abbildung 5-13 zu entnehmen.

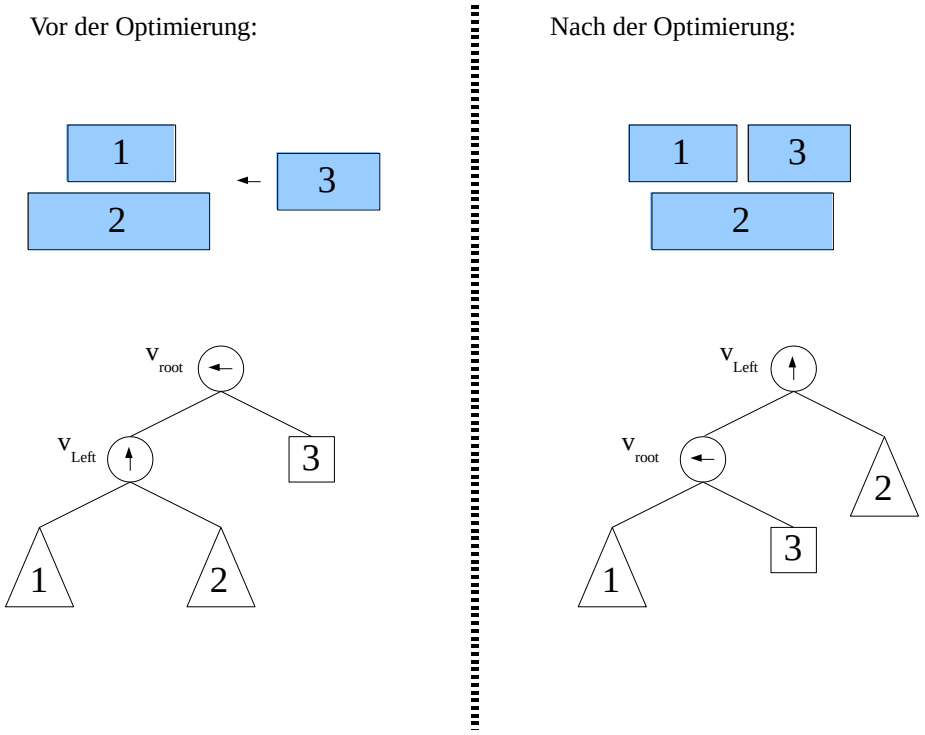


Abbildung 5-13: Optimierung des Schnittbaums bei einer orthogonalen Ausrichtung der Kompaktierungsrichtungen zwischen der Wurzel und dem linken Nachfolger

Es können an dieser Stelle zwei mögliche Veränderungen unterschieden werden. Zunächst wird die nachfolgende Ungleichung betrachtet. Diese ist für das vorliegende Beispiel gegeben durch

$$w\big(\mathrm{Right}(v_{\mathrm{Left}})\big) + w\big(\mathrm{Right}(v_{\mathrm{Root}})\big) \leq (1+\delta)\cdot w\big(\mathrm{Left}(v_{\mathrm{Left}})\big), \tag{5-13}$$

d.h. es wird die Summe aus der Weite des neu hinzugefügten Moduls und der Weite des rechten Nachfolgers von v_{Left} gebildet und mit der skalierten Weite des linken Nachfolgers von v_{Left} verglichen. Falls die Ungleichung erfüllt ist, erfolgt eine Restrukturierung des Schnittbaums. Dabei wird der rechte Nachfolger von v_{Left} zum linken Kindknoten von v_{Root}. Entsprechend verweist der rechte Kindknoten von v_{Left} auf v_{Root}. Schließlich bildet v_{Left} die Wurzel des Schnittbaums. Eine Änderung der Kompaktierungsrichtung von v_{Root} ist nicht notwendig.

Wenn die Ungleichung nicht zutrifft, wie im gezeigten Beispiel, bleibt eine weitere Ungleichung zu prüfen. Diese ist wie folgt definiert

$$w\big(\mathrm{Left}(v_{\mathrm{Left}})\big) + w\big(\mathrm{Right}(v_{\mathrm{Root}})\big) \leq (1+\delta)\cdot w\big(\mathrm{Right}(v_{\mathrm{Left}})\big), \tag{5-14}$$

Es wird, im Gegensatz zur vorhergehenden Ungleichung, die Summe aus der Weite des neuen Moduls und der Weite des Moduls, welches aus dem linken Nachfolger von v_{Left} hervorgeht, gebildet. Entsprechend wird das Ergebnis in Relation mit der skalierten Weite des Moduls aus dem rechten Nachfolger von v_{Left} gesetzt. Sofern die Ungleichung erfüllt ist, wird der linke Kindknoten von v_{Left} zum linken Nachfolger von v_{Root}. Der Knoten v_{Root} wird demgemäß zum linken Nachfolger von v_{Left}. Die Repräsentation der Wurzel des Schnittbaums erfolgt schließlich durch den Knoten v_{Left}. Dieser Fall ist im Beispiel in Abbildung 5-13 dokumentiert. Der rekonstruierte Schnittbaum ist in der rechten Seite der Abbildung zusammen mit der korrespondierenden Modulanordnung abgebildet.

Algorithmus 5-10: Ablauf der Optimierung eines Schnittbaumes

```
v_Left:=Left(v_Root);
wenn (Dir(v_Root)=Dir(Left(v_Root))) dann
   wenn ((((Dir(v_Root)=WEST) oder (Dir(v_Root)=EAST))
            und (h(Right(v_Root))+h(Right(v_Left))≤(1+δ)·h(Left(v_Left))))
         oder (((Dir(v_Root)=NORTH) oder (Dir(v_Root)=SOUTH))
            und (w(Right(v_Root))+w(Right(v_Left))≤(1+δ)·w(Left(v_Left))))) dann
      wenn (Dir(Right(v_Left))⊥Dir(v_Root)) dann
         Dir(v_Root):=Dir(Right(v_Left));
      sonst
         Dir(v_Root):=zufällige Kompaktierungsrichtung orthogonal zu Dir(v_Root);
      wenn_ende
      Left(v_Root):=Right(v_Left);
      Right(v_Left):=v_Root;
      Root(B_S):=v_Left;
   wenn_ende
sonst wenn (Dir(v_Root)=OppositeDir(Dir(Left(v_Root))) dann
   wenn ((((Dir(v_Root)=WEST) oder (Dir(v_Root)=EAST))
            und (h(Right(v_Root)+h(Left(v_Left))≤(1+δ)·h(Right(v_Left))))
         oder (((Dir(v_Root)=NORTH) oder (Dir(v_Root)=SOUTH))
            und (w(Right(v_Root)+w(Left(v_Left))≤(1+δ)·w(Right(v_Left))))) dann
      wenn (Dir(Left(v_Left))⊥Dir(v_Root)) dann
         Dir(v_Root):=Dir(Left(v_Left));
      sonst
         Dir(v_Root):=zufällige Kompaktierungsrichtung orthogonal zu Dir(v_Root);
      wenn_ende
      Left(v_Root):=Left(v_Left);
      Left(v_Left):=v_Root;
      Root(B_S):=v_Left;
   wenn_ende
sonst wenn (Dir(v_Root)⊥Dir(Left(v_Root))) dann
   wenn ((((Dir(v_Root)=WEST) oder (Dir(v_Root)=EAST))
            und (w(Left(v_Left)+w(Right(v_Root))≤(1+δ)·w(Right(v_Left))))
         oder (((Dir(v_Root)=NORTH) oder (Dir(v_Root)=SOUTH))
            und (h(Left(v_Left)+h(Right(v_Root))≤(1+δ)·h(Right(v_Left))))) dann
      Left(v_Root):=Left(v_Left);
      Left(v_Left):=v_Root;
      Root(B_S):=v_Left;
   sonst wenn ((((Dir(v_Root)=WEST) oder (Dir(v_Root)=EAST))
            und (w(Right(v_Left)+w(Right(v_Root))≤(1+δ)·w(Left(v_Left))))
         oder (((Dir(v_Root)=NORTH) oder (Dir(v_Root)=SOUTH))
            und (h(Right(v_Left)+h(Right(v_Root))≤(1+δ)·h(Left(v_Left))))) dann
      Left(v_Root):=Right(v_Left);
      Right(v_Left):=v_Root;
      Root(B_S):=v_Left;
   wenn_ende
wenn_ende
```

Im Algorithmus 5-10 sind die Schritte zur Optimierung für die drei möglichen Fälle zusammenfassend beschrieben. Ebenso ist hier die Verallgemeinerung der Ungleichungen dargestellt. Die genannten Veränderungen des Schnittbaums werden nur dann durchgeführt, wenn die jeweilige Ungleichung erfüllt ist.

Die Auswertung des Schnittbaums, aus der die tatsächliche Platzierung abgeleitet werden kann, erfolgt mit Hilfe einer Post-Order-Traversierung. Die dazu notwendige rekursive Methode ist in Algorithmus 5-11 dargestellt. Für die Ausführung ist zu Beginn die Wurzel des Schnittbaums als Startknoten anzugeben. Das Platzierungsergebnis bzw. die Zwischenergebnisse werden in den Knoten jeweils gespeichert, so dass nach Ablauf der Funktion die endgültige Platzierung der Wurzel des Schnittbaums entnommen werden kann.

Algorithmus 5-11:Auswertung des Schnittbaums zum Aufbau einer Platzierung

```
Funktion EvalSlicinigTree(node)
Beginn
   wenn ((Left(node)≠∅) und (IsLeaf(Left(node))=FALSE)) dann
      EvalSlicingTree(Left(node));
   wenn_ende
   wenn ((Right(node)≠∅) und (IsLeaf(Right(node))=FALSE)) dann
       EvalSlicingTree(Right(node));
   wenn_ende
   CalcAlignment(node);
   m_Dest:=Module(Left(node));
   m_Comp:=Module(Right(node));
   dir_Comp:=Dir(node);
   kompaktiere das Modul m_Comp an das Modul m_Dest mit der Kompaktierungsrichtung
   dir_Comp;
   speichere das Kompaktierungsergebnis im Knoten node ab;
Ende
```

Bei der Traversierung des Schnittbaumes ist darauf zu achten, dass der rekursive Aufruf der Funktion lediglich für innere Knoten erfolgt. Der linke Nachfolger eines Knotens verweist auf das Zielobjekt, das zu kompaktierende Objekt wird durch den rechten Nachfolger referenziert. Basierend auf den Kompaktierungsinformationen wird die Ausrichtung zwischen den zu kompaktierenden Modulen bestimmt. Anschließend wird die Kompaktierung durchgeführt und das daraus resultierende neue Modul im aktuellen Knoten abgelegt. Der Vorgang wird solange rekursiv ausgeführt, bis die Wurzel des Schnittbaumes erreicht ist. Nach Abschluss des Verfahrens ist das finale Layout der Wurzel zu entnehmen.

Der mögliche Verlauf der Ausrichtung zweier Module zueinander, in Abhängigkeit von der Kompaktierungsrichtung, ist dem Algorithmus 5-12 zu entnehmen. Falls es sich beim zu untersuchenden Knoten um die Wurzel handelt, werden die beiden zu kompaktierenden Module bei einer nördlichen bzw. südlichen Kompaktierungsrichtung horizontal zueinander zentriert. Andernfalls erfolgt eine vertikale Zentrierung der beiden Module. Wenn der aktuelle Knoten nicht der Wurzel des Schnittbaums entspricht, wird die Kompaktierungsrichtung des Elternknotens mit herangezogen für die Berechnung der Modulausrichtung. Erfolgt die Kompaktierung für den aktuellen Knoten und dessen Elternknoten in beiden Fällen in vertikaler bzw. horizontaler Richtung, d.h. stimmen die beiden Kompaktierungsrichtungen überein bzw. verlaufen parallel zueinander, so werden die beiden Module zentriert ausgerichtet.

Für den Fall, dass die beiden Kompaktierungsrichtungen orthogonal sind, bleibt zu prüfen, ob der Knoten in Bezug auf den Elternknoten ein linker bzw. rechter Nachfolger ist. Falls es sich um einen linken Kindknoten handelt, erfolgt die Ausrichtung der beiden Module entgegen der Kompaktierungsrichtung des Elternknotens. D.h. für eine im Elternknoten ablegte Kompaktierung in nördlicher Richtung werden die beiden Module entlang einer gemeinsamen südlichen Sichtlinie ausgerichtet. Auf analoge Weise erfolgt die Ausrichtung für die drei verbleibenden möglichen Himmelsrichtungen.

Handelt es sich hingegen bei dem aktuellen Knoten in Bezug auf den Elternknoten um einen rechten Nachfolger, so werden die Module gemäß der Kompaktierungsrichtung des übergeordneten Knotens positioniert. Bei einer nördlichen Richtung im Elternknoten werden die beiden Module entlang einer gemeinsamen nördlichen Linie ausgerichtet. Die Behandlung der verbleibenden drei Kompaktierungsrichtungen erfolgt auf vergleichbare Weise.

Algorithmus 5-12: Beschreibung der Methode CalcAlignment

```
wenn (IsRoot(node)=TRUE) dann
   wenn ((Dir(node)=NORTH) oder (Dir(node)=SOUTH)) dann
      CenterHor(Left(node),Right(node));
   sonst
      CenterVer(Left(node),Right(node));
   wenn_ende
sonst
   dirP:=ParentDir(node);
   wenn ((dirP=Dir(node)) oder (dirP=Opposite(Dir(node)))) dann
      wenn ((Dir(node)=NORTH) oder (Dir(node)=SOUTH)) dann
         CenterHor(Left(node),Right(node));
      sonst
         CenterVer(Left(node),Right(node));
      wenn_ende
   sonst
      wenn (IsLeftChild(node)=TRUE) dann
         wenn (dirP=NORTH) dann
            SameSouth(Left(node),Right(node));
         sonst wenn (dirP=EAST) dann
            SameWest(Left(node),Right(node));
         sonst wenn …
         wenn_ende
      sonst
         wenn (dirP=NORTH) dann
            SameNorth(Left(node),Right(node));
         sonst wenn (dirP=EAST) dann
            SameEast(Left(node),Right(node));
         sonst wenn …
         wenn_ende
      wenn_ende
   wenn_ende
wenn_ende
```

5.6.4 Entscheidungsregeln

Eine Entscheidung unter Ungewissheit bezeichnet Situationen bei denen zwar die Alternativen, die möglichen Zustände und die Ergebnisse bei Wahl einer bestimmten Alternative und Eintritt eines Zustands bekannt sind, in denen aber die Eintrittswahrscheinlichkeiten unbekannt sind. Die Entscheidung unter Ungewissheit besteht aus einer Auswahl zwischen verschiedenen Alternativen $a_i \in A = \{a_1,\ldots,a_m\}$, die abhängig von den möglichen Zuständen $s_j \in S = \{s_1,\ldots,s_n\}$ verschiedene Nutzwerte u_{ij} zur Folge haben. Die Nutzwerte können durch eine Nutzenmatrix $U = (u_{ij})$, mit $1 \leq i \leq m$ und $1 \leq j \leq n$, dargestellt werden. Eine Alternative dominiert eine andere, wenn diese unter allen Zuständen mindestens denselben Nutzen besitzt wie die übrigen Alternativen und in wenigstens einem Zustand einen höheren Nutzen erzielt. Genauer gesagt gilt:

$$a_k \text{ dominiert } a_i \Leftrightarrow \forall j: u_{kj} \geq u_{ij} \text{ und } \exists j: u_{kj} > u_{ij} \,. \tag{5-15}$$

Existiert eine dominante Alternative in einer Entscheidungssituation, so können alle verbleibenden Alternativen gestrichen werden.

Für zwei zu kompaktierende Module werden die zulässigen Kompaktierungsrichtungen sowie die verschiedenen Topologien untersucht. Es wird unter Einbeziehung der noch zu platzierenden Module ein Entscheidungsbaum erstellt. Dabei werden für jedes nachfolgende Modul ebenfalls die möglichen Darstellungen, Transformationen und Verschiebungsrichtungen betrachtet. Nach Auswertung des Entscheidungsbaums ergeben sich, unter Verwendung der nachfolgenden Module, diverse Platzierungsergebnisse. Für diese sind jeweils die Kosten berechnet worden. Um eine Abbildung der erhaltenen Daten auf das Modell zur Entscheidung unter Ungewissheit zu ermöglichen, bilden die

möglichen Kombinationen der ersten beiden Module die Alternativen. Die Kosten einer Platzierung entsprechen dem Nutzwert. Die verschiedenen Platzierungsergebnisse repräsentieren die möglichen Zustände. Es ergibt sich eine Darstellung, wie in Tabelle 5-1 gezeigt.

Tabelle 5-1: Ergebnismatrix für die Entscheidungsregeln

Alternativen	Zustände
$\{m_{A1}, m_{B1}, Dir_1\}$	$c_{11}, c_{12}, \ldots, c_{1l}$
$\{m_{A2}, m_{B2}, Dir_2\}$	$c_{21}, c_{22}, \ldots, c_{2l}$
…	…
$\{m_{Am}, m_{Bn}, Dir_4\}$	$c_{k1}, c_{k2}, \ldots, c_{kl}$

Eine Alternative besteht aus einem Datentupel, in dem die jeweiligen Topologien der beiden zu kompaktierenden Module und die Kompaktierungsrichtung enthalten sind. Die Topologie kann dabei bereits einer Transformation unterzogen worden sein. Aus der Menge der Alternativen ist unter Berücksichtigung der verschiedenen Zustände die kostengünstigste auszuwählen. Durch Angabe einer Wertfunktion w: A $\rightarrow$IR wird eine Entscheidungsregel festgelegt. Es genügt daher im Folgenden die Wertfunktionen der jeweiligen Entscheidungsregeln zu erklären. Die Auswahl einer der im Folgenden beschriebenen Entscheidungsregeln erfolgt durch den Schaltungsdesigner. Dazu ist in der Konfiguration des Platzierungs- und Verdrahtungswerkzeugs die entsprechende Regel anzugeben.

5.6.4.1 Maximin-Regel

Die von ihrem Begründer A.Wald ursprünglich als Verlustfunktion angewandte Regel wird auch Minimax-Regel genannt [Wald50]. Es wird jeweils nur geringste Nutzen betrachtet, welcher bei der Wahl einer Alternative in den möglichen Zuständen eintreten kann. Die Alternativen werden nur anhand der jeweils schlechtesten Ergebnisse verglichen, während andere mögliche Resultate nicht betrachtet werden. Die Wertfunktion berechnet somit zu einer Alternative den garantierten Nutzen

$$w_{\text{maximin}}(a_i) = \min\{u_{ij} | 1 \le j \le n\}. \tag{5-16}$$

5.6.4.2 Maximax-Regel

Die Maximax-Regel ist eine äußerst optimistische Entscheidungsregel. Es wird jede Alternative nur anhand des höchsten Nutzens beurteilt. Für die Entscheidung wird das Maximum der Zeilenmaxima der Nutzenmatrix verwendet. Die Wertfunktion berechnet zu einer Alternative den Maximalnutzen

$$w_{\text{maximax}}(a_i) = \max\{u_{ij} | 1 \le j \le n\}. \tag{5-17}$$

Die Maximin- und Maximax-Regeln berücksichtigen nicht alle Nutzwerte, sondern wählen stets nur den höchsten oder niedrigsten Nutzen einer Alternative aus.

5.6.4.3 Hurwicz-Regel

Die Entscheidungsregel nach Hurwicz erlaubt einen Kompromiss zwischen pessimistischen und optimistischen Entscheidungsregeln, indem mittels eines Optimismusparameters λ, mit $0 \le \lambda \le 1$ die subjektive Einstellung zum Ausdruck gebracht werden kann. Die jeweiligen Zeilenmaxima der Nutzenmatrix werden mit λ und die Zeilenminima entsprechend mit $(1-\lambda)$ multipliziert. Die Wertfunktion berechnet sich damit gemäß der nachfolgenden Gleichung

$$w_{\text{Hurwicz}}(a_i) = \lambda \cdot \max\{u_{ij} | 1 \le j \le n\} + (1-\lambda) \cdot \min\{u_{ij} | 1 \le j \le n\}. \tag{5-18}$$

Größere Werte für λ bedeuten eine optimistischere Grundeinstellung, wobei für λ=1 die Anwendung der Maximax-Regel vorliegt und für λ=0 die Maximin-Regel. Es werden ebenfalls nicht alle Nutzwerte betrachtet, sondern die Alternativen anhand eines gewichteten Mittelwertes des besten und des schlechtesten Nutzwertes bewertet.

5.6.4.4 Laplace-Regel

Bei der Laplace-Regel erhalten alle Zustände die gleiche Eintrittswahrscheinlichkeit. Es wird sich für die Alternative mit den größtmöglichen mittleren Nutzen entschieden. Dazu wird das arithmetische Mittel der Zeilenwerte der Nutzenmatrix berechnet. Die Wertfunktion berechnet sich damit wie folgt

$$w_{\text{Laplace}}(a_i) = \frac{1}{n}(u_{i1} + u_{i2} + \cdots + u_{in}). \tag{5-19}$$

Es kommt hier zu einem echten gegenseitigen Abwägen der Bedingungen, wobei allerdings die Annahme der Gleichwahrscheinlichkeit systematisch anfechtbar und in vielen Fällen aus Sicht der Praxis unrealistisch ist.

5.6.4.5 Niehans-Savage-Regel

Um die betreffende Entscheidungsfunktion definieren zu können, ist zunächst für jeden Zustand s_j der maximale Nutzen $u_j^* = \max\{u_{ij} | 1 \leq j \leq n\}$ zu ermitteln. Im Vergleich mit den u_j^* ergeben sich zu jeder Alternative a_i sogenannte Bedauernswerte $d_{ij} = u_{ij} - u_j^*$ mit $1 \leq i \leq m$ und $1 \leq j \leq n$. Die d_{ij} sind sämtlich kleiner oder gleich Null. Je größer ein Bedauernswert dem Absolutbetrag nach ausfällt, desto mehr kann der maximale Nutzen durch den tatsächlichen realisierten Nutzen u_{ij} als verfehlt angesehen werden. Die Wertfunktion zur Niehans-Savage-Regel berechnet den minimalen (negativen) Bedauernswert

$$w_{\text{Niehans}}(a_i) = \min\{d_{ij} | 1 \leq j \leq n\}. \tag{5-20}$$

Es wird sich für die Alternative entschieden, bei der der minimale Bedauernswert am größten ist. Das Vorgehen verrät eine in die Vergangenheit gerichtete pessimistische Einstellung.

5.7 Objektorientierte Implementierung des Platzierers

Im Folgenden werden die Grundzüge der Programmstruktur, ausgewählte Aspekte der Implementierung sowie konzeptionelle Besonderheiten in Bezug auf die Umsetzung in der verwendeten Programmiersprache umrissen. Die Klassen des Platzierungs-und Verdrahtungswerkzeugs sind in der Sprache C++ unter Verwendung der Entwicklungsumgebung SunStudio auf einer SunBlade 2500 erstellt worden. Für die Zugriffe auf Technologiedaten und für bestimmte geometrische Basisfunktionen wurde die MOGLAN-Bibliothek [Wolf96, Wolf98b, Wolf99a] benutzt. Der Abbildung 5-14 ist das vereinfachte Klassendiagramm des Platzierungs- und Verdrahtungswerkzeug zu entnehmen. Aus Gründen der Übersichtlichkeit wurde auf die Darstellung von weiteren Querverbindungen zwischen den einzelnen Klassen sowie auf die Klassendefinitionen der zusätzlich implementierten Bibliothek für Datenstrukturen verzichtet. Diese Bibliothek enthält Klassen für die Bereitstellung von beispielsweise verketteten Listen, Arrays, Graphen- und Baumstrukturen.

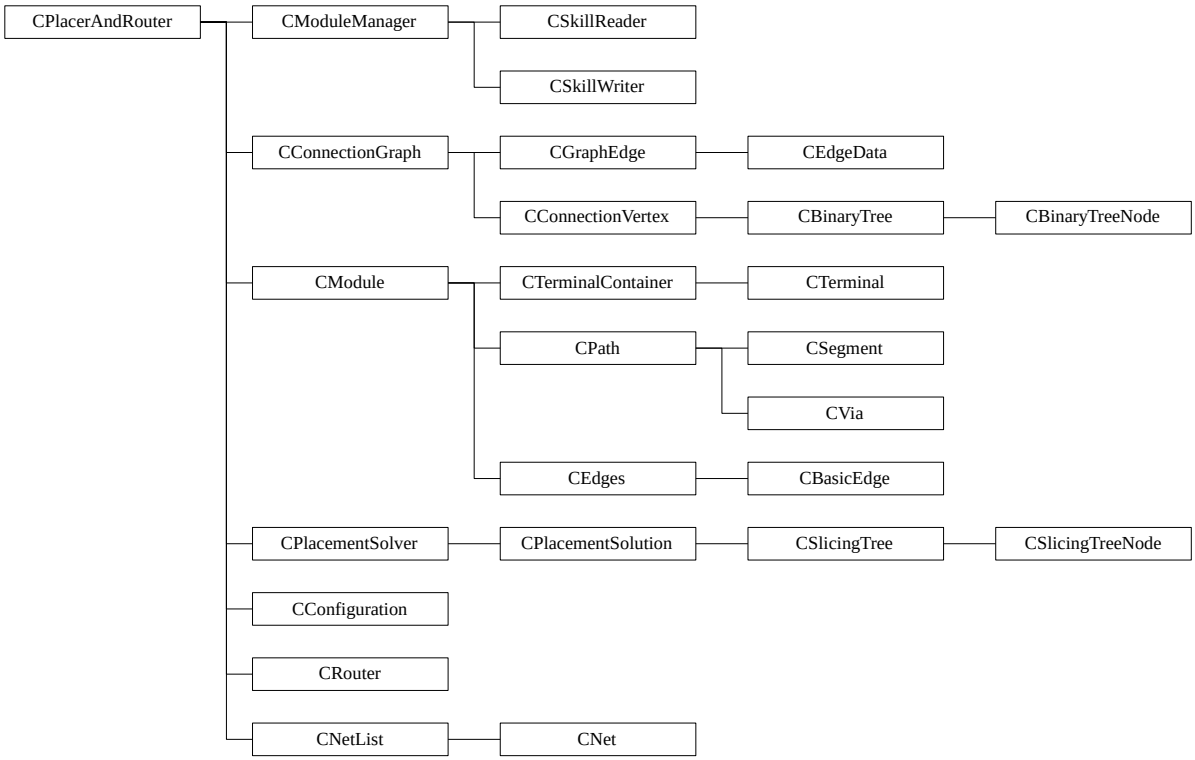


Abbildung 5-14: Vereinfachte Klassenhierarchie des Platzierungswerkzeugs

Insgesamt lassen sich die Klassen in sieben Gruppen unterteilen. Für die Verwaltung der Netzlisten existieren zwei Klassen (CNetList, CNet). Der Aufbau und die Verwaltung des Verbindungsgraphen erfordert sechs Klassen (CConnectionGraph, CGraphEdge, CConnectionVertex, CBinaryTree, CBinaryTreeNode). Acht Klassen werden für die Repräsentation der Module deren Kantenbilder, Anschlüsse und Pfade (CModule, CTerminalContainer, CTerminal, CPath, CSegment, CVia, CEdges, CBasicEdge) verwendet. Zur Überführung der Eingabedaten und zur Speicherung der Ergebnisse sind drei Klassen (CModuleManager, CSkillReader, CSkillWriter) implementiert worden. Der Aufbau und die Auswertung des Platzierungsbaums erfolgt mit Hilfe von vier Klassen (CPlacmentSolver, CPlacementSolution, CSlicingTree, CSlicingTreeNode). Die benutzerspezifischen Einstellungen für das Platzierungs- und Verdrahtungswerkzeug werden durch die Klasse CConfiguration gekapselt. Die Verfahren für die Verdrahtung sind in der Klasse CRouter modelliert. Die Klasse verfügt über eine eigene Hierarchie, deren Beschreibung dem nachfolgenden Kapitel 6 entnommen werden kann. Alle aufgeführten Klassendefinitionen werden durch die Klasse CPlacerAndRouter verwaltet. Sie bildet damit die zentrale Schnittstelle.

Die Klasse CModuleManager verwaltet neben den Klassen zum Lesen und Schreiben der Layoutdaten, die Methoden für die Überführung der geometrischen Elemente in ein Kantenbild. Für jedes Modul wird eine Instanz vom Typ CModule erzeugt, die einen Verweis auf die in einer Datei gespeicherten geometrischen Objekte enthält. In einem Modul werden alle Verschiebe- und Transformationsoperationen, die während der Platzierung an dem jeweiligen Modul ausgeführt werden, gespeichert. Nach Abschluss des Platzierungs-und Verdrahtungsverfahrens werden die Ergebnisse in einer Datei abgelegt. Dabei werden die Layoutdaten der jeweiligen Module aus den entsprechenden Dateien eingelesen, den in einem Modul abgelegten Translationen und Transformationen unterzogen und in die Zieldatei geschrieben. Für das Einlesen der geometrischen Objekte wird die Klasse CSkillReader benutzt. Sie basiert auf der abstrakten Klasse CFileReader. Diese

stellt Funktionen zur Verwaltung der Daten bereit und bildet eine einheitliche Schnittstelle für die Klasse CModuleManager. Die Klasse CSkillReader erweitert CFileReader dahingehend, dass SKILL-Befehle von Cadence ausgewertet werden können. Die Befehle beschreiben die Layoutdaten. Auf analoge Weise wird die Klasse CSkillWriter, die von der abstrakten Klasse CFileWriter abgeleitet ist, zum Speichern der Layoutdaten im SKILL-Format genutzt. Die abstrakten Basisklassen kapseln die I/O-Aufgaben, während deren Nachfolger für die Datenkonvertierung zuständig sind. Das verwendete Konzept ermöglicht eine einfache Erweiterung der I/O-Schnittstelle auf andere Eingabeformate.

Der Verbindungsgraph wird durch die Klasse CConnectionGraph bereitgestellt. Neben der Verwaltung der Knoten und Kanten verfügt die Klasse über Methoden zur Vereinigung der Knoten und der damit verbundenen Aktualisierung der Platzierungsbäume. In der Klasse CConnectionVertex, mit deren Hilfe ein Knoten des Verbindungsgraphen modelliert wird, ist ein Verweis auf eine Instanz eines Platzierungsbaums enthalten. Der Platzierungsbaum wird durch die Klassen CBinaryTree und CBinaryTreeNode repräsentiert. Die Gewichtungen der Kanten des Verbindungsgraphen werden im Datencontainer CEdgeData abgelegt. Die Kanten selbst werden mittels der Klasse CGraphEdge instanziiert.

Die Klasse CModule verwaltet eine Liste der verschiedenen Topologien, die für das jeweilige Modul bereitstehen. Bei einem Wechsel der Topologie wird das Kantenbild des Moduls automatisch aktualisiert. Falls das Modul aus der Kompaktierung anderer Module hervorgegangen ist, werden die Referenzen auf die entsprechenden Module in diesem abgelegt. Bezugnehmend auf die kompaktierten Module werden deren Kantenbilder in das übergeordnete Modul kopiert und einem Verschmelzungsvorgang der Kanten unterzogen. Ein Objekt der Klasse CModule ist damit entweder der Repräsentant einer einzelnen Zelle oder die Darstellung der hierarchischen Anordnung der Instanzen von CModule. Für jede Seite des Objekts (Nord, Ost, Süd, West) werden in je einer Instanz vom Typ CEdges die äußeren Kanten des Objekts abgelegt. In der Liste der Kanten sind alle Kanten des hierarchisch aufgebauten Moduls gespeichert. Dies erlaubt einen schnellen Zugriff auf die Daten bei der Kompaktierung. Aufgrund der Tatsache, dass für jede Zelle lediglich die äußeren Kanten gespeichert werden, erfolgt in jeder Instanz von CModule eine Speicherung der Informationen zu den geometrischen Operationen. Die Operatoren, in Form von Translationen und Transformationen, werden beim Erzeugen der Ergebnisdatei auf die tatsächlichen Layoutdaten angewandt, um die finale eigentliche Platzierung zu erhalten.

In der Klasse CModule ist weiter eine Liste der Anschlüsse enthalten, die durch Instanzen der Klasse CTerminalContainer bereitgestellt werden. Die Klasse basiert auf einer doppelt verketteten Liste. Sie enthält zusätzliche Methoden für geometrische Operationen. Für jedes Netz wird ein Objekt vom Typ CTerminalContainer instanziiert. Jeder Anschluss, der in diesem Container abgelegt ist, wird durch eine Instanz der Klasse CTerminal repräsentiert. Es handelt sich dabei um eine Ableitung der Klasse CBasicEdge. Sie verfügt dadurch über geometrische Operatoren und die Datenfelder einer Kante. Zusätzlich kann ein Anschluss einen Verweis auf einen Pfad aufnehmen, was für die Verdrahtung von Bedeutung ist.

Ein Modul verfügt weiterhin über eine Liste der Instanzen der Pfade vom Typ CPath. Der Klasse CPath stehen Datenfelder und Methoden zur Verwaltung der geometrischen Objekte einer Verbindungsroute zur Verfügung. Die einzelnen Abschnitte einer Leitung werden durch die Klasse CSegment modelliert. Für mögliche Durchkontaktierungen ist die Klasse CVia zu verwenden. Alle drei

Klassen zur Darstellung eines Pfades basieren, wie die Klasse CModule, auf der Klasse CGeometry. Diese Basisklasse kann einen Verweis auf die tatsächlichen Layoutdaten enthalten. Sie ist in der Lage neben den Kantenbildern Listen von Rechtecken und Polygonen zu verwalten. Ebenso erfolgt in der Klasse CGeometry die Umsetzung des Kompaktierungsverfahrens. Damit verfügt jede abgeleitete Klasse über die Fähigkeit den minimalen Abstand zu benachbarten Objekten, die ebenfalls auf CGeometry zurückzuführen sind, mittels der Kompaktieres zu berechnen. Neben den geometrischen Operatoren verfügt die Klasse über Methoden um eine Ausrichtung bzw. Zentrierung mit einer anderen Geometrie durchführen zu können.

Die Klasse CPlacementSolver beinhaltet die Schritte für den Aufbau und die Auswertung einer Platzierung unter Verwendung eines Entscheidungsbaums. In der Klasse werden die jeweiligen Platzierungsfolgen, die durch Instanzen vom Typ CPlacementSolution realisiert werden, verwaltet. Die Topologie und die Orientierung eines Moduls sind zu diesem Zeitpunkt bereits festgelegt. Neben der Folge der zu platzierenden Module ist eine Liste der zu verwendenden Kompaktierungsrichtungen in jeder Lösung enthalten. Der Aufbau der Platzierung und damit die Umsetzung der Kompaktierungsangaben erfolgt mit Hilfe einer Instanz der Klasse CSlicingTree über die jedes Objekt vom Typ CPlacementSolution verfügt. Die Klasse CSlicingTree ist eine abgeleitete Klasse der Binärbaumdarstellung CBinaryTree. Sie wurde um die Optimierungsmethoden und einer Post-Order-Traversierung zur Konstruktion der Platzierung erweitert. Jeder Knoten des Baums ist ein CSlicingTreeNode-Objekt. Es handelt sich dabei um einen Datencontainer, der neben einen Verweis auf die jeweilige Modultopologie Informationen über die Kompaktierungsrichtung aufnehmen kann. Die Berechnung der Kosten der resultierenden Platzierung wird in CPlacementSolution durchgeführt. Die anschließende Beurteilung der Kosten und die Auswahl einer Platzierung, unter Verwendung der Entscheidungsregeln, erfolgt in der Klasse CPlacementSolver.

Alle benutzerdefinierten Informationen zur Platzierung und Verdrahtung, wie beispielsweise die Lage der Versorgungsschienen, die zu benutzende Entscheidungsregel oder die Suchtiefe für den Entscheidungsbaum, werden durch die Klasse CConfiguration gekapselt.

6 Verdrahtung

Der überwiegende Teil der Verdrahtungsverfahren, die heute industriell eingesetzt werden, basieren auf einer zweistufigen Vorgehensweise, der Global- und der Detailverdrahtung. Während der Globalverdrahtung erfolgt eine Zuweisung von Verdrahtungsflächen zu jedem zu verdrahtenden Netz. Es wird auf diese Weise eine Abschätzung des Verlaufs des jeweiligen Netzes gegeben, ohne auf die genaue Geometrie einzugehen. Die physikalische Ausführungsform wird erst in der Detailverdrahtungsphase festgelegt. Dadurch ist es vor Ausführung der Detailverdrahtung schwierig, eine sichere Aussage zur Länge und den parasitären Effekten einer Leitung zu geben.

Im Nachfolgenden wird ein Verfahren zur Verdrahtung analoger integrierter Schaltungen beschrieben, welches stets nach einem Kompaktierungsschritt ausgeführt wird. Zunächst erfolgt eine Beschreibung der generellen Vorgehensweise, um die Verdrahtung zwischen zwei Zellen auszuführen. Anschließend werden die Erstellung einer orthogonalen Polygonhülle für eine Zellgeometrie und der darauf basierende Aufbau eines Graphen zur Repräsentation der möglichen Verbindungswege beschrieben. Der resultierende Wegenetz-Graph ist mit ein zentraler Bestandteil der Routensuche, die in Kap 6.3 ausführlich erläutert wird. Neben der Darstellung und Analyse von Routensuchverfahren zur Wegsuche in einem Graphen, wird in diesem Abschnitt die Verwendung von einfachen Leitungselementen zur Verbindung zweier Terminals untersucht. Die Suche nach einer Verbindung mit Hilfe des Wegenetz-Graphen erfolgt mit Hilfe des Algorithmus nach Floyd und Warshall (siehe Kap. 6.3.2). Bei der Layouterstellung eines Pfades (siehe Kap. 6.4) und dem Erzeugen von Durchkontaktierungen kann es zu Abstandverletzungen bzw. Kurzschlüssen kommen. Aus diesem Grund wird eine Entwurfsregelprüfung (siehe Kap. 6.5) durchgeführt, die das Auffinden und Korrigieren möglicher Fehler beinhaltet. Am Schluss des Kapitels wird kurz auf die objektorientierte Implementierung des Verdrahtungskonzepts eingegangen, durch die die Möglichkeit einer effektiven Wartung und Weiterentwicklung des Softwarepakets gegeben ist.

6.1 Grundlegende Vorgehensweise zur Verdrahtung

Der Aufbau des Layouts integrierter analoger Schaltungen erfolgt mit Hilfe eines hierarchischen Kompaktierers. Dabei werden stets zwei Objekte zurzeit betrachtet und zu einem neuen Objekt vereinigt. Nachdem die vorläufige Anordnung der beiden Zellblöcke auf diese Weise bestimmt wurde, bleiben die notwendigen Verdrahtungen zu berechnen. Der grundlegende Ablauf dafür wird in diesem Abschnitt beschrieben. Eine detaillierte Erläuterung der einzelnen Phasen erfolgt in den nachfolgenden Unterkapiteln.

Jedes der beiden Module, für welche die Routensuche durchgeführt werden soll, kann sich aus mehr als einer Zelle zusammensetzen. Um die möglichen Verbindungswege zu bestimmen, werden alle Zellen der beiden zu verdrahtenden Module erfasst und in einer Liste gespeichert. Die Liste der Zellen und alle bereits existierenden Pfade zwischen den Zellen werden beim Aufbau eines Wegenetz-Graphen zur Repräsentation der möglichen Verbindungsrouten ausgewertet. Eine einzelne Zelle wird durch ein Kantenbild zu jeder Seite dargestellt. Ferner steht ein Hüllpolygon zur Verfügung bzw. kann aus den geometrischen Daten der Zelle abgeleitet werden. Mit Hilfe dieser Umrandung werden die Verbindungskanten des Wegenetz-Graphen erzeugt. Die vorhandenen Leitungen werden hingegen in den Verbindungsgewichten des Graphen berücksichtigt. Aufgrund der Tatsache, dass im Rahmen dieser Arbeit schwerpunktmäßig die Platzierung und Verdrahtung analoger integrierter Schaltungen untersucht werden soll, wird eine Routenführung über die Zellen nicht in Betracht gezogen. Diese, bei digitalen Schaltungen häufiger anzutreffende, Vorgehensweise ist, bedingt durch

die resultierenden Streukapazitäten, für analoge Schaltungen nicht zu empfehlen. Nach dem Aufbau und der Initialisierung des Wegenetz-Graphen werden die gemeinsamen Netze der beiden zu verdrahtenden Module ermittelt. In jedem Netz sind Informationen zum Leitungstyp und eine benutzerdefinierte Priorität abgelegt. Bei den Leitungstypen wird zwischen Signal-, Takt-, Eingangs- und Versorgungsleitungen unterschieden. Die höchste Rangordnung erhalten die Signalleitungen, den niedrigsten Rang bekommen die Versorgungsleitungen zugewiesen. Diese Reihenfolge kann jedoch in den Konfigurationsdateien des Verdrahters durch den Benutzer verändert werden. Auf diese Weise wird eine Ordnung auf den Netzen definiert.

Die Abfolge innerhalb einer der vier Gruppen ist durch nutzerspezifische Priorität desentsprechenden Netzes festgelegt. Bei einer Übereinstimmung dieses Wertes wird die Reihenfolge zufällig bestimmt. Das beschriebene Ordnungsschema wird zur Sortierung der zu verbindenden Netze verwendet. Die Suche nach dem jeweiligen kostenoptimalen Pfad für jedes Netz erfolgt in einer Schleife über die sortierten gemeinsamen Netze. Zu jedem Netz können mehrere Anschlüsse existieren, so dass alle verschiedenen Anschlusskombinationen bei der Routenwahl zu berücksichtigen sind. Aus den Kombinationen wird die günstigste Route, bestehend aus den beiden Anschlüssen und einer Folge von Wegpunkten ausgewählt. Für diese Route wird das vorläufige Layout erstellt. In jedem Netz ist die zu erwartende Stromdichte abgespeichert. Der Wert kann aus einer DC-Simulation der Schaltung ermittelt werden. Gemäß Gl. (6-25), die in Abschnitt 6.4 vorgestellt wird, ist es damit möglich, die Leitungsbreite in Abhängigkeit von der Stromdichte zu berechnen. Dadurch wird der Effekt der Elektromigration bei der Pfaderstellung mit in Betracht gezogen. Nach der Layouterstellung wird der vorläufige Pfad in einer temporären Liste abgelegt. Die Kantengewichte des Wegenetz-Graphen werden im Hinblick auf den neuen Pfad aktualisiert.

Algorithmus 6-1: Genereller Ablauf der Verdrahtung

```
erfasse Geometrie-Daten der zu verbindenden Module A und B;
initialisiere Wegpunktsystem;
suche nach gemeinsamen Netzen der Module A und B;
Suchergebnisse im Vektor CommonNets speichern;
anhand der gemeinsamen Netze alle möglichen Terminalkombinationen ermitteln;
CommonNets nach Typ und Priorität der jeweiligen Netze sortieren;
für i:=1 bis |CommonNets|
   berechne für net_i∈CommonNets die möglichen Verbindungen und wähle
   kostengünstigste Verdrahtung aus;
   erstelle vorläufiges Layout für das jeweilige Netz;
   speichere erzeugten Pfad in der List PathList ab;
   aktualisiere Kosten des Wegpunktsystems;
für_ende
sortiere PathList nach Anzahl der Segmente der jeweiligen Pfade;
für i:=1 bis |PathList|
   führe für P_i∈PathList Prüfung der Entwurfsregeln durch;
für_ende
```

Im Anschluss an die Suche nach geeigneten Routen für jedes Netz und der Layouterstellung der entsprechenden Pfade, wird eine Entwurfsregelprüfung durchgeführt. Dazu werden die Pfade bzgl. der Anzahl ihrer Leitungselemente aufsteigend sortiert. Bei einer Übereinstimmung der Segmentanzahl werden der Leitungstyp und die Priorität als Sortierkriterium hinzugezogen. In einer anschließenden Iteration über die auf diese Weise sortierten Pfade wird das Layout jeder Route auf Abstandsverletzungen und Kurzschlüsse mit anderen Zellen und Pfaden untersucht. Dabei ermittelte Fehler werden automatisch korrigiert. Die Layouts der verifizierten Pfade werden in einer neuen Zelle

zusammen mit den beiden kompaktierenden Zellblöcken abgelegt. Eine Zusammenfassung des Verdrahtungsablaufs ist in Algorithmus 6-1 dargestellt.

6.2 Aufbau eines Wegenetz-Graphen

Für die Ausführung der Algorithmen zur Routensuche wird, basierend auf den Geometrie-Daten der Zellen, eine angemessene logische Repräsentation erzeugt. Der Kern des logischen Datenmodells ist ein Wegenetz, bestehend aus einem ungerichteten Graphen, welcher die Kanäle um und zwischen den Zellen modelliert. Die Kanten des so gebildeten Graphen repräsentieren eine spezifische Art von Weg. Für die Wege können Angaben zu Qualitätsmerkmalen und den Weg beeinflussende Hindernisse gemacht werden. Das Wegenetzwerk bildet die Basis für die Abbildung von Routen. Die Knoten des Graphen für das Wegenetz gehen aus den Eckpunkten der umhüllenden Polygone der Zellen, sowie beim Übergang von einem Hüllpolygon ins nächste, hervor.

Zur Erstellung des Wegenetz-Graphen ist zunächst die orthogonale Polygonhülle der beteiligten Zellen zu bestimmen. Diese wird während der Initialisierungsphase des Platzierungs- und Verdrahtungswerkzeugs für jede Geometrie der Zellen generiert. Es sind dazu die Eckpunkte der Polygone und Rechtecke, aus denen sich die jeweilige Geometrie einer Zelle zusammensetzt, in einer Punktmenge zu erfassen. Anschließend soll die konvexe Hülle für diese Punktmenge errechnet werden. Aus dem resultierenden Hüllpolygon wird die gesuchte orthogonale Hülle der Zelle abgeleitet. Die konvexe Hülle einer Punktmenge M ist die kleinste konvexe Punktmenge, die M umfasst. Die konvexe Hülle ist ein Polygon, dessen Eckpunkte eine Teilmenge von M bilden, falls M eine endliche Menge ist. Das Problem der konvexen Hülle besteht darin, zu einer gegebenen, endlichen Punktmenge M der Größe n eine Liste von Punkten $p_1,\ldots p_H$ zu berechnen, so dass die p_i die Ecken der konvexen Hülle von M in der richten Reihenfolge sind. Zur Bestimmung der konvexen Hüllen existieren verschiedene Vorgehensweisen, nämlich der Graham-Scan [Grah72], der Jarvis-March [Jarv73] oder der Quickhull-Algorithmus [Prep85].

Beim Verfahren von Jarvis [Jarv73] wird die Menge der Punkte in der Ebene wie mit einer Schnur umwickelt. Den Startpunkt dabei bildet der Punkt mit der niedrigsten y-Koordinate. Der nächste Eckpunkt ist jeweils der, der den kleinsten Polarwinkel in Bezug auf den als letztes hinzugefügten Punkt der Hülle aufweist. Der Polarwinkelvergleich wird mit Hilfe von Kreuzprodukten durchgeführt. Das Verfahren endet, wenn der Ausgangspunkt wieder erreicht ist. Die Suche nach dem Punkt mit der minimalen y-Koordinate erfordert eine Laufzeit von O(n). Für die Bestimmung des nächsten Eckpunkts sind h Schritte notwendig, wobei h die Anzahl der Ecken des konvexen Hüllpolygons ist. Daraus ergibt sich insgesamt eine Zeitkomplexität von $O(n \cdot h)$. Der Zeitbedarf ist stark abhängig von der Anzahl der Ecken des Hüllpolygons. Im schlechtesten Fall befinden sich die meisten Punkte der zu untersuchenden Punktmenge auf dem Rand der konvexen Hülle. Die Zeitkomplexität liegt damit zwischen O(n) und $O(n^2)$.

Der Quickhull-Algorithmus [Prep85] verwendet eine vergleichbare Technik wie das Sortier-Verfahren Quicksort. Die Menge der zu untersuchenden Punkte wird in Teilmengen gegliedert. Für die Partitionierung wird in jeder zu teilenden Menge eine Gerade genutzt, die aus den Extremalpunkten der Ausgangsmenge erzeugt wird. Je nach Lage der Punkte in Bezug auf die Gerade werden diese in zwei Teilmengen verteilt. Die resultierenden Mengen werden rekursiv weiter unterteilt. Im besten Fall werden mit dieser Vorgehensweise in jedem Rekursionsschritt stets so viele Punkte in das Innere des bis dahin erzeugten Polygons übertragen, wie noch außerhalb verbleiben. Die Laufzeit beträgt

dann O(n · log n). Im schlechtesten Fall befindet sich der überwiegende Teil der Punkte auf dem Rand des Hüllpolygons. Die Zeitkomplexität beträgt dann $O(n^2)$.

Im Rahmen der vorliegenden Arbeit wird zur Berechnung der konvexen Hülle der Graham-Scan verwendet. Die Idee des Verfahrens von Graham [Grah72] besteht darin, aus der Punktmenge zunächst ein sternförmiges Polygon zu konstruieren und dieses anschließend in ein konvexes Polygon umzuformen. Dazu wird aus der Punktmenge zunächst der Punkt ausgewählt, der die kleinste y-Koordinate aufweist. Falls es mehrere Punkte gibt, wird von diesen der Punkt mit minimaler x-Koordinate verwendet. Ausgehend von diesem Punkt werden die Winkel zu allen anderen Punkten berechnet. Die Menge der Punkte wird nach den ermittelten Winkeln sortiert und zu einem Linienzug verbunden. Die Sortierung erfolgt in O(n · log n). Das resultierende Sternpolygon wird beginnend von eingangs ermittelten minimalen Punkten durchlaufen. Während der Iteration werden die konkaven Ecken überbrückt. Für den Ablauf wird ein Stapelspeicher verwendet, dessen Inhalt die konvexe Hülle der bisher betrachteten Punkte darstellt. Im k-ten Schritt der Schleife wird der Punkt p_k dahingehend betrachtet, inwieweit die bisherige konvexe Hülle durch den Punkt verändert wird. Infolge der Sortierung der Punktmenge liegt p_k stets außerhalb der Hülle der vorhergehenden Punkte. Befindet sich der Punkt p_k links von den obersten beiden Punkten des Stapelspeichers, so wird p_k auf dem Stack abgelegt. Das gleiche gilt, wenn der Stack weniger als zwei Elemente enthält. Falls sich der Punkt p_k rechts von den obersten beiden Punkten des Stapelspeichers befindet, so werden die Punkte aus dem Stack entfernt und die nächsten beiden oberen Punkte untersucht. Nach Abschluss der Iteration enthält der Stack die Eckpunkte der konvexen Hülle. Während des Schleifendurchlaufs wird jeder Punkt höchstens einmal auf dem Stack ablegt bzw. entfernt. Es finden damit lediglich O(n) Operationen auf dem Stack statt. Ferner sind O(n) Kreuzprodukte und Vergleiche vorzunehmen. Die Laufzeitkomplexität des Verfahrens wird damit von der Sortierung der Punktmenge dominiert, so dass sich eine Laufzeit von O(n · log n) einstellt.

Zur Sortierung der Punkte im Hinblick auf den Polarwinkel zum Punkt p_{min}, ist es nicht notwendig, den Winkel zu berechnen. Es ist lediglich von Bedeutung, welcher von je zwei zu vergleichenden Punkten p_i und p_j den größeren Winkel aufweist. Dazu ist der orientierte Flächeninhalt des Dreiecks, dass durch die Punkte p_{min}, p_i und p_j aufgespannt wird, zu untersuchen. Der doppelte Flächeninhalt des Dreiecks ist durch die Beziehung

$$X(p_i) \cdot Y(p_j) - X(p_j) \cdot Y(p_i) \tag{6-1}$$

gegeben. Falls die resultierende Fläche gleich Null ist, d.h. beide Punkte p_i und p_j weisen den gleichen Winkel auf, wird die Manhattan-Distanz zum Punkt p_{min} für den Vergleich verwendet. Ergibt sich eine Fläche mit negativem Vorzeichen, so ist der Winkel von p_i zu p_{min} kleiner als der Winkel von p_j zu p_{min}. Für die Sortierung der Punkte wird das "Merge Sort"-Verfahren verwendet [Knut98]. Das Sortierverfahren weist eine Zeitkomplexität von O(n · log n) auf. Nach der Berechnung der konvexen Hülle, deren Eckpunkte im Stack S gespeichert sind, erfolgt eine weitere Iteration über die Punkte des Stacks. Dabei wird das endgültige orthogonale Hüllpolygon der Zelle erstellt. Während des Schleifendurchlaufs werden alle aufeinanderfolgenden Punkte p_i und p_{i+1} näher untersucht, die nicht durch eine rechtwinklig verlaufende Polygonkante miteinander verbunden sind. In diesem Fall wird zunächst das Maximum der horizontalen bzw. vertikalen Seite des Steigungsdreiecks zwischen p_i und p_{i+1} bestimmt. Ferner wird das betragsmäßige Maximum der Distanzen von p_i nach p_{i-1} bzw. von p_{i+1} nach p_{i+2} unter Verwendung der Manhattan-Distanz berechnet. Falls der Vergleich zu Gunsten der orthogonalen Verbindung zwischen p_i und p_{i+1} ausfällt, wird ein neuer Punkt zwischen p_i und p_{i+1}, in

Abhängigkeit von der Lage von p_i und p_{i+1} zueinander, eingefügt. Andernfalls wird eine Koordinate von p_{i+1} modifiziert und der Punkt p_i aus dem Hüllpolygon entfernt, so dass die orthogonale Struktur für das Hüllpolgyon erhalten bleibt. Mit Hilfe des Faktors α im Vergleich der Maxima kann die Auflösung der orthogonalen Hülle der Zelle festgelegt werden. Betragsmäßig große Werte für α führen dabei zu einer vereinfachten Darstellung der Hülle, bis hin zu einer rechteckigen Umgebungsbox. Umgekehrt bedingt eine detaillierte Repräsentation der Zelle, infolge von kleinen Werten für α, eine längere Laufzeit sowie eine erhöhten Speicherbedarf bei der Routensuchen unter Nutzung des Wegenetz-Graphen.

Algorithmus 6-2: Berechnung des orthogonalen Hüllpolygons einer Zelle

```
suche Punkt pmin mit minimaler y-Koordinate;
entferne pmin aus der Punktmenge M;
sortiere Punktmenge M nach ihrem Winkel zu pmin;
erstelle Stack S;
S.push(pmin);
S.push(first(M));
k:=2;
solange k<|M|
   pi:=S.peek();
   wenn pi-1,pi,pk konvex sind dann
      S.push(pk);
      k:=k+1;
   sonst
      S.pop();
   wenn_ende
solange_ende
für alle pi∈S
   wenn (X(pi)≠X(pi+1) und (Y(pi)≠Y(pi+1)) dann
      Δ:=max{|X(pi)-X(pi+1)|,|Y(pi)-Y(pi+1)|};
      Δm:=max{dist(pi,pi-1),dist(pi+1,pi+2)};
      wenn α·Δm<Δ dann
         erstelle neuen Punkt p';
         wenn (X(pi+1)≥X(pi) und Y(pi+1)≥Y(pi))
               oder (X(pi+1)≤X(pi) und Y(pi+1)≤Y(pi)) dann
            X(p'):=X(pi);
            Y(p'):=Y(pi+1);
         sonst
            X(p'):=X(pi+1);
            Y(p'):=Y(pi);
         wenn_ende
      sonst
         wenn (X(pi+1)≥X(pi) und Y(pi+1)≥Y(pi))
               oder (X(pi+1)≤X(pi) und Y(pi+1)≤Y(pi)) dann
            Y(pi+1):=Y(pi);
         sonst
            X(pi+1):=X(pi);
         wenn_ende
         entferne pi aus S;
      wenn_ende
   wenn_ende
für_ende
```

Basierend auf den Hüllpolygonen der Zellen erfolgt die Erstellung des Wegenetz-Graphen. Dazu werden aus den Eckpunkten der rechtwinkligen Zellhüllen die Knoten des Graphen generiert. Jeder Knoten beinhaltet somit Informationen über seine Lage in der Topologie der Zellen. Die Knoten von aufeinanderfolgenden Eckpunkten eines Polygons werden im Graphen mit einer Kante verbunden, wodurch die Kanten den Außenseiten der Zellhüllen entsprechen. In einer Kante werden neben den Kosten, um von einem Knoten der Kante zum anderen zu gelangen, ein Attributfeld "Zellhülle/Sichtlinie" abgelegt, aus dem hervorgeht, ob die Kante Teil eines Hüllpolygons einer Zelle

ist oder eine Verbindung zwischen zwei Zellen repräsentiert. Ferner werden in einer Kante ein Verweis auf die jeweilige Zelle und die Lage der Kante bzgl. der Polygonhülle durch Angabe einer Haupthimmelsrichtung gespeichert, insofern die Kante aus einem Hüllpolygon hervorgeht.

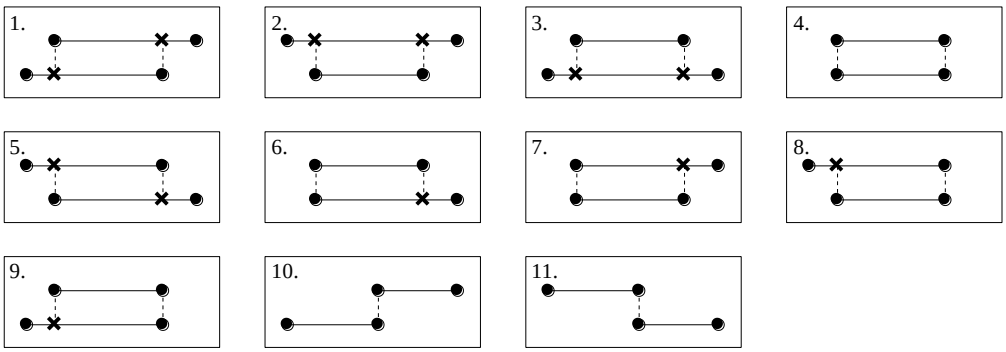


Abbildung 6-1: Mögliche Lage von zwei horizontalen Kanten

Die Überführung der rechtwinkligen Polygonhüllen liefert zunächst einen nicht zusammenhängenden Graphen. Im anschließenden Schritt werden die möglichen Verbindungen zwischen den Polygonen in den Graphen übertragen, Mittels einer Iteration über alle Kanten mit dem Attribut "Zellhülle" werden die möglichen Verbindungen ermittelt, um einen zusammenhängenden Graphen für das Wegenetz zu erhalten. Innerhalb der Schleife werden für die jeweils aktive Kante e_i alle Kanten gesucht, die die nachfolgenden Bedingungen erfüllen. Zunächst müssen die entsprechenden Kanten parallel zu e_i verlaufen und bzgl. der gespeicherten Himmelsrichtung den gegenüberliegenden Kardinalspunkt aufweisen. Die zu suchenden Kanten gehören nicht zu der gleichen Zelle wie e_i. Die Kanten dürfen hinsichtlich der Sichtbarkeit zueinander nicht disjunkt sein, d.h. es muss beispielsweise für horizontale Kanten einer der in Abbildung 6-1 dargestellten Fälle vorliegen. Die Prüfung der vertikalen Ausrichtung erfolgt auf analoge Weise. Es darf sich zwischen e_i und der zu prüfenden Kante keine Zellhülle befinden, die eine der beiden Kanten vollständig verdeckt.

Falls Kanten vorliegen, die die genannten Bedingungen erfüllen, werden im nächsten Schritt die Verbindungen zwischen den Knoten der entsprechenden Kanten berechnet. Die Möglichkeiten, die sich für die verschiedenen Verbindungen im horizontalen Fall ergeben, sind der Abbildung 6-1 zu entnehmen. Die in der Abbildung gestrichelt dargestellten Strecken beschreiben die neu zu erstellenden Verbindungskanten, während die Kreuze den jeweils neu zu erzeugenden Knoten entsprechen. Ausgehend von den Knoten der parallel verlaufenden Kanten werden Halbgeraden, die Sichtlinien, gebildet, die orthogonal zur jeweiligen Kante verlaufen. Für den Schnitt mit der gegenüberliegenden Kante ergeben sich dabei zwei Möglichkeiten. Der aus dem Schnitt zwischen der Kante und der Sichtlinie resultierende Punkt ist deckungsgleich mit einem Knoten. In diesem Fall ist lediglich eine neue Kante im Graphen einzufügen, die die beiden Knoten miteinander verbindet. Andernfalls wird für den Schnittpunkt ein neuer Knoten erstellt und die entsprechende Kante in zwei Kanten gespalten. Alle Kanten, die neu erzeugt werden, bekommen das Attribut "Sichtlinie" zugewiesen. Nach Abschluss der Iteration über die Verbindungskanten, die aus den Polygonhüllen hervorgehen, liegt der Graph des Wegenetzes vor.

Der Ablauf für die Erstellung des Wegenetz-Graphen ist in Abb. 6-2 anhand eines Beispiels zusammenfassend dargestellt. Zunächst erfolgt die vorläufige Generierung des Graphen aus der orthogonalen Hülle der Zellen. Anschließend werden die rechtwinkligen Sichtlinien ermittelt, weitere Verbindungskanten und ggf. neue Knoten erstellt, um den zusammenhängenden Graphen des

Wegenetzes zu erzeugen. Auf diese Weise ist ein logisches Modell für die Verdrahtungskanäle entstanden, das für die im Nachfolgenden beschriebene Pfadsuche genutzt werden kann.

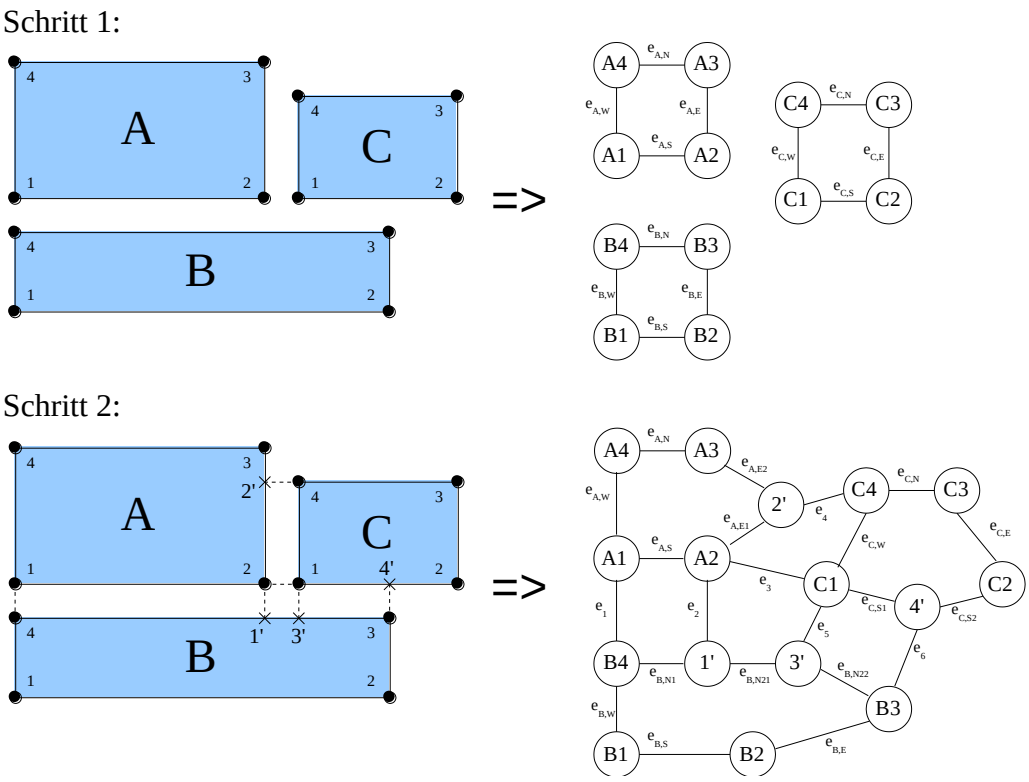


Abbildung 6-2: Erstellung eines Wegenetz-Graphen basierend auf der Topologie der Zellen

6.3 Routensuche

Zur Durchführung der Routensuche wurden verschiedene Vorgehensweisen implementiert, die in diesem Abschnitt vorgestellt werden. In Abhängigkeit der topologischen Lage zweier zu verbindender Anschlüsse erfolgt die Verdrahtung durch einfache Verdrahtungsformen bzw. durch die Suche eines Pfades im Wegenetz-Graphen. Es werden alle möglichen Anschlusskombinationen für jedes zu verdrahtende Netz untersucht. Infolge der Verwendung des hierarchischen Kompaktierers bei dem Aufbau der Platzierung, sind für die Wegsuche lediglich Zwei-Punkt-Verbindungen zu betrachten. Es können dennoch Mehr-Punkt-Netze entstehen, da bereits erstellte Pfade in die Auswahl der Anschlusskombinationen mit einbezogen werden.

Für die Routensuche unter Verwendung von vorgegeben Verdrahtungsformen stehen die folgenden Elemente zur Verfügung: Gerade Leitungselemente (G-Shapes), Leitungselemente mit einem Knick (L-Shapes) und Leitungselemente mit zwei Knicken in unterschiedlichen Richtungen (Z-Shapes). Die jeweiligen Verdrahtungselemente verbinden zwei Anschlüsse in Manhattan-Geometrie auf dem kürzest möglichen Weg. Die Erstellung der ggf. erforderlichen Durchkontaktierungen erfolgt während der Layoutgenerierung des jeweiligen Pfades bzw. bei der Entwurfsregelprüfung. Die resultierenden Vias werden nach Möglichkeit an den Symmetriepunkten der Verdrahtungselemente positioniert. Die Verdrahtung im Rahmen dieser Arbeit ist nicht auf einer Verdrahtungsebene je Richtung festgelegt. Jedem Verdrahtungs-Segment werden die Pfadbreite, der Layer, die Netz-Informationen und die Verbindungskosten zugewiesen. Die Netz-Informationen beinhalten neben der Netz-Id und der zu erwartenden Stromdichte, den Leitungstyp. Es wird zwischen Signal-, Takt-, Eingangs-und

Versorgungsleitungen unterschieden. Die Layoutgenerierung der Pfade erfolgt erst, nachdem alle Routen zwischen den beiden kompaktierten Zellblöcken bestimmt wurden.

Mögliche neue Wege setzen sich bei der Routensuche aus einer Folge von Wegpunkten zusammen. Ferner ist die voraussichtliche Pfadbreite, die aus der Stromdichte des entsprechenden Netzes abgeleitet werden kann (siehe Kap 6.4, Gl. (6-25)), für jede der Wegpunkt-Folgen mit angegeben. Basierend auf diesen Werten können die Streukapazitäten zwischen den einzelnen Pfaden mit abgeschätzt werden.

Die Kosten eines Pfades setzen sich aus der Weglänge und den kapazitiven Einkopplungen zusammen. Eine vereinfachende Darstellung ist durch die nachfolgende Gleichung gegeben:

$$c(W) = \alpha_L \cdot c_L(W) + \alpha_{\text{cap}} \cdot c_{\text{cap}}(W). \tag{6-2}$$

wobei $W = \{v_1,\ldots,v_n\}$ eine Folge von Wegpunkten repräsentiert, die den Verlauf des jeweiligen Pfades beschreiben, $\alpha_L \cdot c_L(W)$ sind die gewichteten Kosten für die Weglänge und $\alpha_{cap} \cdot c_{cap}(W)$ entsprechend die gewichteten Kosten für die kapazitiven Einkopplungen. Die Kosten für die Weglänge sind gegeben durch

$$c_L(W) = \sum_{i=1}^{|W|-1} \|v_{i+1} - v_i\|, \tag{6-3}$$

wobei $\|\cdot\|$ die Manhattan-Distanz zwischen zwei Punkten ist. Die Kosten für die kapazitiven Einkopplungen lassen sich durch die nachfolgende Gleichung beschreiben:

$$c_{\text{cap}}(W) = \sum_{i=1}^{|W|-1} \delta \cdot \|v_{i+1} - v_i\|/\gamma, \tag{6-4}$$

wobei $\gamma \in \mathbb{R}^+$ einen Faktor darstellt, der vom Leitungstyp von W und W_i abhängig ist und für δ gilt:

$$\delta = \begin{cases} 1 & \text{falls ein } W_i \text{ existiert mit } \overline{v_{i+1}v_i} \cap W_i \neq \emptyset \\ 0 & \text{sonst} \end{cases}. \tag{6-5}$$

Falls W und W_i Versorgungsleitungen sind, so gilt $\gamma \geq 1$, andernfalls gilt $0 < \gamma < 1$. Auf diese Weise wird ein Maß für die Anzahl der Kurzschlüsse im Layout festgelegt. Die Suche nach einer Überlappung zwischen einem Leitungssegment von W und einem existierenden Pfad ist gegeben durch:

$$\left\{\overline{v_{j+1}v_j} \cap \overline{v_{k+1}v_k} \middle| v_j \in W, v_k \in W_i\right\}. \tag{6-6}$$

wobei über alle bereits erzeugten Pfade W_i zu iterieren ist.

Wie bereits erwähnt wurde, erfolgt die Routensuche nicht ausschließlich mit Hilfe geeigneter Verfahren auf dem Wegenetz-Graphen, sondern es wird zuerst geprüft, inwieweit eine Verbindung der Anschlüsse mit Hilfe eines einfachen Leitungselements erfolgen kann. Zunächst wird die Verwendung eines geraden Elements untersucht. Dazu müssen die Terminals parallel zueinander verlaufen und am gleichen Verdrahtungskanal angrenzen. Hinsichtlich der Sichtbarkeit zwischen den beiden Anschlüssen liegt eine Überlappung vor und keiner der Anschlüsse wird durch eine weitere Zelle vollständig verdeckt. Die möglichen Ausrichtungen für horizontal verlaufende Anschlüsse sind in Abbildung 6-3 gezeigt. Es ergeben sich vier verschiedene Varianten. Die Ausführungen für den vertikalen Verlauf erfolgen auf analoge Weise.

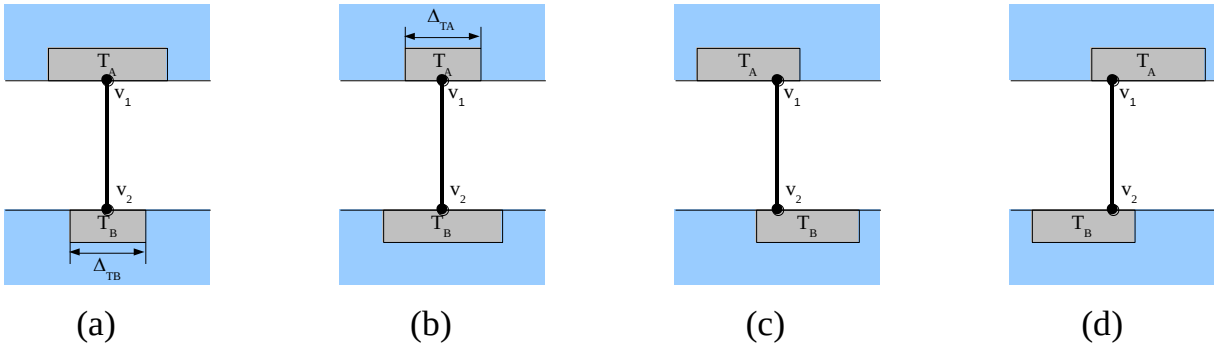


Abbildung 6-3: Erstellung der Wegpunkte für gerade Leitungselemente

Die x-Koordinaten der Wegpunkte für die vier verschiedenen Fälle berechnen sich nach Gl. (6-7) bis Gl. (6-10), die y-Koordinate entspricht der des jeweiligen Anschlusses. Es ist

$$X(v_1) = X(v_2) = \big(X_l(T_B) + X_r(T_B)\big)/2, \tag{6-7}$$

$$X(v_1) = X(v_2) = \big(X_l(T_A) + X_r(T_A)\big)/2, \tag{6-8}$$

$$X(v_1) = X(v_2) = \big(X_r(T_A) + X_l(T_B)\big)/2, \tag{6-9}$$

$$X(v_1) = X(v_2) = \big(X_l(T_A) + X_r(T_B)\big)/2, \tag{6-10}$$

wobei $X_l(T)$ bzw. $X_r(T)$ der jeweiligen X-Koordinate des linken bzw. rechten Eckpunkts des Anschlusses entspricht, T_A den Anschluss A und T_B entsprechend das Terminal B bezeichnet. Falls die Breite einer der beiden Anschlüsse kleiner als die gewünschte Pfadbreite ist, erfolgt die Konstruktion des Pfades lediglich in der Breite des entsprechenden Terminals. Die Einhaltung der Entwurfsregeln ist an dieser Stelle sichergestellt, da jedes Terminal die in der jeweiligen Technologie geforderte Mindestweite einhält. Für den Fall, dass die Weite des überlappenden Bereichs kleiner als das technologiebedingte Minimum ausfällt, erfolgt die Verbindung mit Hilfe eines Z-förmigen Elements.

Wie bei den geraden Leitungselementen müssen die zu verbindenden Terminals bei einem Z-förmigen Element parallel zueinander verlaufen und dürfen nicht vollständig von einer Zelle verdeckt werden. Der mögliche Verlauf der Wegpunkte ist in Abbildung 6-4 dargestellt. Es ergeben sich vier verschiedene Anordnungsmöglichkeiten.

Der Ablauf zur Bestimmung der Wegpunkte von Z-förmigen Verdrahtungselementen ist in Algorithmus 6-3 dargestellt. Darin wird zunächst die Ausrichtung der Anschlüsse unterschieden. Für den horizontalen Verlauf der Terminals können die y-Koordinaten der Punkte sofort berechnet werden, während die x-Koordinaten mittels einer Fallunterscheidung zu bestimmen sind. Analog dazu können die x-Koordinaten bei einer vertikalen Ausrichtung der Terminals direkt angegeben werden, wohingegen die y-Werte durch weitere Unterscheidungen zu ermitteln sind.

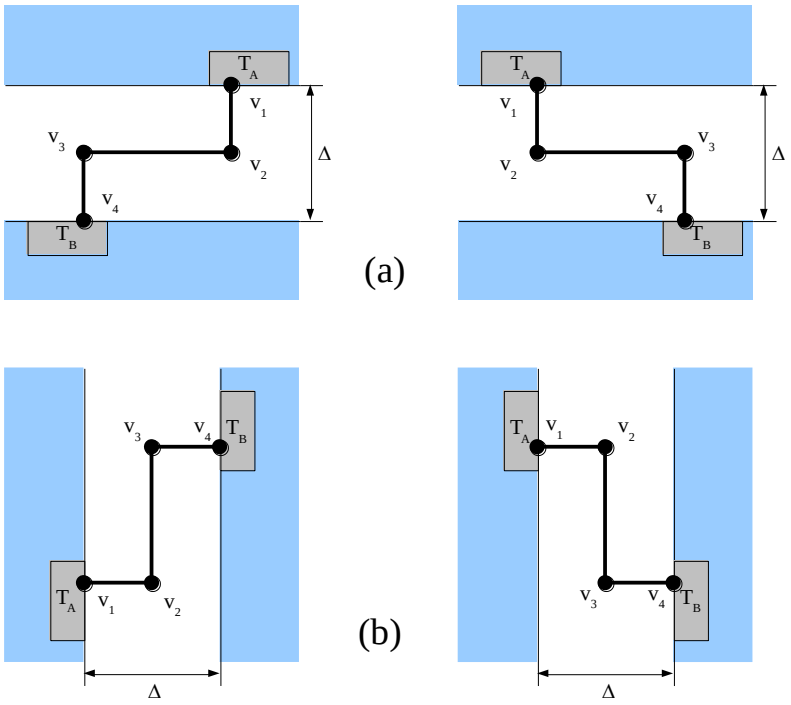


Abbildung 6-4: Erstellung der Wegpunkte für Z-förmige Verbindungen zwischen (a) horizontal und (b) vertikal verlaufenden Anschlüssen

Algorithmus 6-3: Generierung der Wegpunkte eines Z-förmigen Leitungselements

```
wenn (IsHorizontal(T_A) und IsHorizontal(T_B)) dann
   // Erstellung der Wegpunkt für horizontal verlaufende Anschlüsse
   Y(v_1):=Y_l(T_A); Y(v_4):=Y_l(T_B);
   Y(v_2):=Y(v_3):=(Y_l(T_A)+Y_L(T_B))/2;
   wenn (Len(T_A)<W_Path) dann
      X(v_1):=(X_l(T_A)+X_r(T_A))/2;
   sonst
      wenn (X_l(T_A)>X_r(T_B)) dann
         X(v_1):=X_l(T_A)+W_Path/2;
      sonst
         X(v_1):=X_r(T_A)-W_Path/2;
      wenn_ende
   wenn_ende
   X(v_2):=X(v_1);
   wenn (Len(T_B)<W_Path) dann
      X(v_1):=(X_l(T_B)+X_r(T_B))/2;
   sonst
      wenn (X_l(T_A)>X_r(T_B)) dann
         X(v_1):=X_l(T_B)+W_Path/2;
      sonst
         X(v_1):=X_r(T_B)-W_Path/2;
      wenn_ende
   wenn_ende
   X(v_3):=X(v_4);
sonst
   // Erstellung der Wegpunkte für vertikal verlaufende Anschlüsse
   …
wenn_ende
```

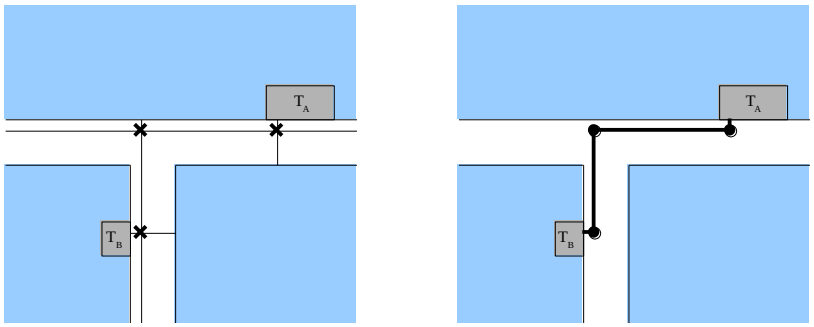


Abbildung 6-5: Verdrahtung mittels eines L-förmigen Leitungselements

Algorithmus 6-4: Erstellung der Wegpunkte für L-Förmige Elemente

```
wenn (TA⊥TB) dann
   gA:=Parallele zum Anschluss TA;
   gB:=Parallele zum Anschluss TB;
   S:=TA∩TB;
   // Erstelle Wegpunkte v1,v2 am Anschluss TA
   wenn (IsHorizontal(TA)) dann
      wenn (Len(TA)<WPath) dann
         X(v1):=(Xl(TA)+Xr(TA))/2;
      sonst
         wenn (X(S)≤Xl(TA)) dann
            X(v1):=Xl(TA)+WPath/2;
         sonst
            X(v1):=Xr(TA)-WPath/2;
         wenn_ende
      wenn_ende
      X(v2):=X(v1); Y(v1):=Yl(TA); Y(v2):=Y(S);
   sonst
      wenn (Len(TA)<WPath) dann
         Y(v1):=(Yl(TA)+Yr(TA))/2;
      sonst
         wenn (Y(S)≤Yl(TA)) dann
            Y(v1):=Yl(TA)+WPath/2;
         sonst
            Y(v1):=Yr(TA)-WPath/2;
         wenn_ende
      wenn_ende
      Y(v2):=Y(v1); X(v1):=Xl(TA); X(v2):=X(S);
   wenn_ende
   // Erstelle Wegpunkte v3,v4 am Anschluss TB
   …
wenn_ende
```

Wenn eine Verbindung weder mit einem geraden noch mittels eines Z-förmigen Verdrahtungselements durchgeführt werden kann, wird noch die Verwendung eines L-förmigen Elements untersucht. Dazu müssen die Anschlüsse hinsichtlich ihrer Ausrichtung orthogonal zueinander verlaufen. Ausgehend von den Terminals werden Geraden erstellt, die parallel zum jeweiligen Anschluss ausgerichtet sind. Für die Bestimmung der Lage einer Geraden wird der extremale Rand der zum Anschluss gehörigen Zelle ermittelt. D.h. falls sich beispielsweise das Terminal auf der Nordseite der Zelle befindet, ist diejenige Polygonkante des Hüllpolygons zu suchen, die die größte y-Koordinate aufweist. Auf den resultierenden Wert wird ein geringer Betrag, z.B. die minimale Gitterweite, aufaddiert bzw. abgezogen. Es bleiben senkrechte Verbindungen von den Anschlüssen zu den Geraden der L-Form zu erstellen, um eine Verbindung sicherzustellen. Nachdem die parallel zu den Terminals verlaufenden Geraden bestimmt wurden, sind die Strecken zwischen dem Schnittpunkt der beiden Geraden und den Anschlusspunkten zu untersuchen. Falls sich der Schnittpunkt nicht in einer Zelle befindet bzw. die Strecken keine Zelle schneiden, kann ein Weg über

den erzeugten Linienzug verlaufen. Die Abfolge für die Erstellung einer L-förmigen Verbindung ist in Abbildung 6-5 anhand eines Beispiels gezeigt.

Die Generierung der Wegpunkte für ein L-förmiges Leitungselement ist in Algorithmus 6-4 zusammengefasst. Die Wegpunkte setzen sich aus dem Schnittpunkt der Geraden und den Eckpunkten der Strecken, die von den Anschlüssen zu den Geraden führen, zusammen. Für jeden Anschluss wird eine Fallunterscheidung ausgeführt, ob der Pfad in der gewünschten Breite umgesetzt werden kann. In diesem Fall werden die Wegpunkte möglichst nah zum Geradenschnittpunkt der L-Form hin verlegt. Andernfalls wird der Mittelpunkt des jeweiligen Anschlusses als Wegpunkt verwendet.

Algorithmus 6-5: Routensuche für zwei zu verbindende Terminals T_A und T_B

```
useGraph:=FALSE;
wenn ((TA||TB) und (OppositeOf(TA,TB)=TRUE) dann
   wenn (GeometriesBetween(TA,TB)=TRUE) dann
      useGraph:=TRUE;
   sonst
      wenn (HasOverlap(TA,TB)=TRUE) dann
         erstelle Wegpunkte für G-Form, um TA und TB zu verbinden;
      sonst
         erstelle Wegpunkte für Z-Form, um TA und TB zu verbinden;
      wenn_ende
   wenn_ende
sonst
   wenn (TA⊥TB) dann
      erstelle Parallele gA zu TA;
      erstelle Parallele gB zu TB;
      S:=Schnittpunkt von gA und gB;
      wenn ((S existiert) und (S nicht innerhalb einer Zelle)
           und (gA schneidet keine Zelle)
           und (gB schneidet keine Zelle)) dann
         erstelle Wegpunkte für L-Form, um TA und TB zu verbinden
      sonst
         useGraph:=TRUE;
      wenn_ende
   sonst
      useGraph:=TRUE;
   wenn_ende
wenn_ende
wenn (useGraph=TRUE) dann
   mit Hilfe des Wegenetz-Graphen eine Verbindung zwischen den beiden
   Anschlüssen TA und TB errechnen;
wenn_ende
```

Sollte der Fall eintreten, dass keines der drei vorgestellten Leitungselemente verwendet werden kann, um eine Verbindung der beiden Anschlüsse zu gewährleisten, wird der Wegenetz-Graph für die Routensuche ausgewertet. Dafür ist zunächst ggf. eine Aktualisierung der Kantengewichte durchzuführen. Alle Routen, die bis zu diesem Zeitpunkt ermittelt wurden, aber im Graphen bislang keine Berücksichtigung finden, sind in die Gewichte der Kante miteinzubeziehen. Zusammenfassend ist der Ablauf für die Wegsuche zwischen zwei Anschlüssen in Algorithmus 6-5 dargestellt.

Die notwendigen Schritte zur Routensuche mit Hilfe eines Graphen werden in den nachfolgenden Abschnitten beschrieben. Es wird in Kap. 6.3.1 eine Übersicht zu möglichen Verfahren für die Wegsuche in einem gewichteten, ungerichteten Graphen gegeben. Eine genaue Analyse und Beschreibung des in dieser Arbeit verwendeten Verfahrens, basierend auf dem Algorithmus nach Floyd und Warshall [Floy62, Wars62] wird in Kap. 6.3.2 durchgeführt.

6.3.1 Darstellung und Analyse von Routensuchverfahren

Im Folgenden werden ausgewählte Routensuchverfahren zur Wegsuche in einem Graphen G= (V, E) präsentiert und näher erläutert. Zunächst werden der Algorithmus von Dijkstra und das A*-Verfahren beschrieben. Sie dienen zur Berechnung eines kürzesten Pfades zwischen einem Startknoten und einem beliebigen Knoten in einem gewichteten Graphen. Anschließend wird das Verfahren nach Moore-Bellman diskutiert. Insgesamt werden eventuelle Besonderheiten und der Zeitbedarf der jeweiligen Vorgehensweisen erläutert. Das im Rahmen dieser Arbeit verwendete Verfahren nach Floyd und Warshall wird in Kap. 6.3.2 ausführlich untersucht.

6.3.1.1 Routensuche nach Dijkstra

Die Grundidee des Verfahrens nach Dijkstra besteht darin, ab einem Startknoten die kürzesten möglichen Wege weiter zu verfolgen und voraussichtliche Umwege auszuschließen [Dijk59]. Der Graph, auf dem der Algorithmus ausgeführt wird, muss positiv gewichtet sein. Im ersten Schritt des Verfahrens werden allen Knoten des Graphen die Eigenschaften Distanz und Vorgänger zugewiesen. Der Startknoten wird mit der Distanz 0 und die verbleibenden Knoten mit ∞ initialisiert. In einer anschließenden Iteration werden alle Distanzen der Nachbarknoten zum aktiven Knoten berechnet. Die besuchten Knoten werden ferner markiert. Falls die errechnete Distanz für einen Nachbarknoten kleiner als der gespeicherte Wert ist, wird der Wert in diesem Knoten aktualisiert und der aktuelle Knoten als Vorgänger eingesetzt. Aus der Menge der Nachbarknoten wird derjenige ausgewählte, der die geringste Distanz aufweist. Dieser bildet den nächsten aktiven Knoten. Die Schleife wird nun solange fortgesetzt, bis alle Knoten des Graphen markiert sind. Der Verlauf des kürzesten Weges kann durch eine Iteration über die Vorgänger, ausgehend vom Zielknoten, erhalten werden.

Die Zeitkomplexität des Algorithmus ist im erheblichen Maß von der Datenstruktur abhängig, die zur Speicherung der nicht markierten Knoten verwendet wird. Bei Benutzung einer Prioritätswarteschlange zur Suche des nächsten Kandidaten aus der Menge der nicht markierten Knoten, kann gezeigt werden, dass sich ein Gesamtaufwand von $O(|V|^2 + |E|)$ ergibt [Rose96]. In [Spir73] ist eine Verbesserung der Minimumsuche durch eine Darstellung der Prioritätswarteschlange in Form eines Heaps präsentiert worden. Der Heap ist ein fast vollständiger Binärbaum mit der Eigenschaft, dass die Werte der Knoten auf dem Weg von der Wurzel zu einem Blatt monoton steigend sind. Das Einsortieren bzw. Entfernen in einem Heap erfordert O(log n) Schritte, wobei n die Anzahl der Knoten des Baumes ist. Es ergibt sich damit ein Gesamtaufwand für den Dijkstra-Algorithmus von $O(|V| \cdot \log|V|^2)$. Eine weitere Verringerung des Zeitbedarfs wurde in [Fred90] erzielt. Anstelle eines Heap wird ein Fibonacci-Heap verwendet [Fred87]. Es handelt sich dabei um einen Wald heapgeordneter Bäume mit jeweils disjunkten Knotenmengen. Heapgeordnet bedeutet, dass die Nachfolgerknoten stets einen größeren Schlüsselwert haben als deren Elternknoten. Vom Elternknoten verweist lediglich ein Zeiger auf nur einen Nachfolger. Die Knoten innerhalb einer Ebene sind dafür untereinander in einer zyklisch geschlossenen, doppelt verketteten Liste miteinander verknüpft. Die Verwendung des Fibonacci-Heaps ermöglicht eine Laufzeit des Verfahrens nach Dijkstra von $O(|E| + |V| \cdot \log|V|)$.

6.3.1.2 A-Algorithmus*

Das informierte Suchverfahren A* bildet eine Erweiterung des Dijkstra-Algorithmus [Hart68, Sedg86]. Es berechnet ebenso wie das Verfahren nach Dijkstra die Distanzen zwischen einem aktiven Knoten und dessen Nachbarn. Der Unterschied besteht jedoch darin, dass für jeden erreichbaren Nachbarn mit Hilfe einer Heuristik eine Schätzung abgegeben wird, wie teuer es ist, vom jeweiligen Nachbarknoten zum Zielknoten zu gelangen. Für jeden dieser Nachbarknoten v addiert das A*-

Verfahren die von der Heuristik geschätzten Kosten bis zum Ziel zu den bisherigen Kosten, um vom aktiven Knoten u nach v zu gelangen. Formal lässt sich die Abschätzung durch die nachfolgende Gleichung beschreiben:

$$f(v) = g(v) + h(v), \tag{6-11}$$

hierbei stellt g(v) die gesamten Kosten vom Startknoten bis zum Knoten v dar, während h(v) die voraussichtlichen Kosten dokumentiert, um von v zum Zielknoten zu gelangen. Die heuristische Funktion darf die tatsächlichen Kosten nicht übersteigen. Es ist vielmehr von Vorteil, wenn der Wert der Heuristik für einen Knoten v kleiner als der Wert für jeden Nachfolgerknoten ist, d.h.

$$h(v) \leq \tilde{h}(v), \tag{6-12}$$

wobei $\tilde{h}(v)$ die tatsächlichen Kosten zum Ziel sind. Eine derartige Heuristik ist beispielsweise durch die euklidische Metrik gegeben.

Die Zeitkomplexität hat eine geringere Bedeutung, da der A*-Algorithmus eine zielgerichtete Suche darstellt und im Vergleich zur Gesamtanzahl der Knoten in der Regel nur einen kleinen Teil dieser Knoten untersucht. Bei Labyrinthen ist dies jedoch oft nicht möglich, so dass die tatsächliche Laufzeit im schlechtesten Fall $O(|V|^2)$ beträgt. Der Zeitbedarf ist zum einen besonders von der Heuristik abhängig, zum anderen von der Datenstruktur zur Suche der nächsten Knoten. Ebenso ist die Qualität der ermittelten Wege, für die zu verlegenden Leitungen, stark abhängig von der Heuristik. Beschränkt diese sich schwerpunktmäßig auf die kürzeste Distanz zum Ziel, werden Routen, die bzgl. dieses Faktors höhere Kosten verursachen, aber mehr auf die Anforderungen der Schaltung eingehen, wie beispielsweise eine Minimierung der Streukapazitäten, vom A*-Verfahren nicht gefunden.

6.3.1.3 Routensuche nach Moore-Bellman

Der Moore-Bellman-Algorithmus, der unabhängig voneinander von Moore und Bellman vorgeschlagen wurde, soll in diesem Abschnitt kurz erläutert werden [Bell58, Moor59]. Zu diesem Verfahren gibt es eine Vielzahl von Verbesserungsvorschlägen [Glov85, Lawl76, Sysl83]. Die verschiedenen Versionen des Moore-Bellman-Algorithmus unterscheiden sich darin, in welcher Reihenfolge die Knoten und Kanten abgearbeitet werden.

Die Idee hinter dem Algorithmus lässt sich wie folgt beschreiben. Ausgehend von einem Startknoten v_s soll zu allen anderen Knoten v ein kürzester Weg bestimmt werden. Jeder Knoten erhält zunächst die Eigenschaften Distanz und Vorgänger. Das Feld Distanz des Startknoten wird mit 0, für die übrigen Knoten mit ∞ initialisiert. In einer anschließenden Iteration wird versucht, sukzessiv die Distanzwerte der Knoten zu reduzieren. Liegt dabei eine Verbindung zwischen einem Knoten v_i mit v vor und gilt ferner

$$\mathrm{dist}(v_i) + c(v_i, v) < \mathrm{dist}(v), \tag{6-13}$$

wobei dist() der gespeicherte Distanzwert des jeweiligen Knoten und c:E→IR die Kostenfunktion sind, dann gilt:

$$\mathrm{dist}(v) = \mathrm{dist}(v_i) + c(v_i, v). \tag{6-14}$$

Ferner wird das Feld Vorgänger von v auf den Knoten v_i gesetzt. Die Schleife wird solange fortgesetzt, bis kein Distanzwert weiter reduziert werden kann. Durch die unterschiedlichen Distanzbewertungen

kann bei beliebiger Wahl des nächsten Kandidatenknoten nicht mehr garantiert werden, dass später gewählte Knoten zu einem günstigeren Weg führen. Weiterhin kann es somit zu einer ständigen Wiederholung der Verbesserung der schon bearbeiteten Knoten kommen.

Die Laufzeit des Verfahrens beträgt insgesamt $O(|V| \cdot |E|)$. Im ungünstigsten Fall ist der Graph ein dichter Graph, somit $|E|$ fast $|V|^2$ und die Zeitkomplexität für den Gesamtalgorithmus fast $O(|V|^3)$. Falls das Moore-Bellman-Verfahren angewandt wird, um kürzeste Wege von jedem Knoten zu jedem anderen Knoten zu finden, beträgt die Laufzeitkomplexität $O(|V|^2 \cdot |E|)$ bzw. im schlechtesten Fall sogar $O(|V|^4)$.

6.3.2 Routensuche mittels Floyd-Warshall Algorithmus

Bevor eine Erläuterung der Grundlagen des Verfahrens zur Routensuche nach Floyd und Warshall erfolgt, werden im Folgenden die Schritte beschrieben, um eine Verbindung zweier Terminals unter Verwendung des Wegenetz-Graphen durchzuführen. Zu jedem Anschluss werden die nächstgelegenen, parallel verlaufenden Verbindungskanten aus dem Graphen ermittelt. Die topologischen Informationen zu einer Kante können aus den Knoten, die durch die Kante verbunden werden, abgeleitet werden. Stimmen beispielsweise die x-Koordinaten der Wegpunkte, die durch die beiden Knoten repräsentiert werden, überein, kann die Kante als eine vertikale Verbindungsstrecke betrachtet werden. In jeder Kante ist ferner ein Verweis auf eine Zelle enthalten, wenn die Kante aus dem Hüllpolygon der Zelle hervorgeht.

Mittels einer Iteration über alle Verbindungskanten des Graphen können diejenigen Kanten ermittelt werden, die parallel zum jeweiligen Terminal verlaufen, sich auf der gleichen Seite der entsprechenden Zelle befinden und die mindestens eine Eckpunkt des entsprechenden Terminals verdecken. Wenn eine überlappende Kante aus dem Graphen bestimmt wurde, bleibt zu prüfen, ob eine bereits zuvor gefundene Kante eine größere Distanz zum Terminal aufweist. In diesem Fall wird die neue Kante, mit dem geringeren Abstand zum Anschluss, gespeichert.

Ein Terminal kann sich über die gesamte Seite einer Zelle erstrecken, wohingegen eine Kante des Wegnetz-Graphen nicht notwendigerweise über diese Distanz verläuft. Vielmehr ist es möglich, dass auf einem solchen Abschnitt mehrere Abzweigungen existieren, so dass eine Seite einer Zelle durch eine Folge von Kanten beschrieben wird. Aus diesem Grund wird eine mögliche Verbindungskante dahingehend untersucht, inwiefern einer der Eckpunkte des jeweiligen Anschlusses die Kante verdeckt wird. In Algorithmus 6-6 ist die Suche der nächstgelegenen Verbindungskanten zu einem Terminal zusammenfassend dargestellt. Als Eingabedaten dienen das zu untersuchende Terminal T sowie der Wegenetz-Graph G= (V, E). Als Ergebnis resultieren maximal zwei Kanten $e_{n,Lo}$ und $e_{n,Hi}$, die die jeweiligen Eckpunkte des Anschlusses T überdecken und den geringsten Abstand zu T aufweisen.

Vor Beginn der Berechnung der möglichen Routen für die Verbindung der beiden Anschlüsse, wird bei jeder Graphenkante, die mit dem zuvor beschriebenen Algorithmus ermittelt wurde, derjenige Knoten bestimmt, der den geringsten Abstand zum entsprechenden Eckpunkt des jeweiligen Terminals aufweist. Somit ergeben sich für jedes Terminal zwei Knoten aus dem Wegnetz-Graphen und dadurch vier mögliche Routen, um von einem Anschluss zum anderen zu gelangen. Die Auswertung des Wegenetz-Graphen zur Bestimmung dieser Routen erfolgt mit Hilfe des Floyd-Warshall Algorithmus. Dazu werden im nachfolgenden Abschnitt 6.3.2.1 die wichtigsten Eigenschaften kürzester Wege in einem ungerichteten, gewichteten Graphen aufgezeigt. Diese Eigenschaften werden zur rekursiven Berechnung der kürzesten Wege genutzt. Eine detaillierte

Beschreibung hierzu ist in Kap. 6.3.2.2 gegeben. Die Extraktion der kürzesten Wege aus dem Wegenetz-Graphen ist Bestandteil von Kap 6.3.2.3.

Algorithmus 6-6: Suche nach nächstgelegenen Verbindungskanten zu einem Terminal

```
e_n,Lo:=e_n,Hi:=∅;
für alle e_i∈E
   wenn (Orientierung(T)=Orientierung(e_i)) dann
      wenn (IsHorizontal(T)=TRUE) dann
         disjunct:=(X_r(T)<X_l(e_i)) oder (X_l(T)>X_r(e_i));
      sonst
         disjunct:=(Y_r(T)<Y_l(e_i)) oder (Y_l(T)>Y_r(e_i));
      wenn_ende
      wenn (disjunct=FALSE) dann
         // prüfen, ob e_i näher an T liegt als eine zuvor bereits
         // ermittelte Kante e_n,Lo
         wenn (e_n,Lo≠∅) dann
            wenn (IsHorizontal(T)=TRUE) dann
               Δ_min:=|Y_l(T)-Y_l(e_n,Lo)|;
               Δ:=|Y_l(T)-Y_l(e_i)|;
               overlap:=(X_l(T)≥X_l(e_i)) und (X_l(T)≤X_r(e_i));
            sonst
               Δ_min:=|X_l(T)-X_l(e_n,Lo)|;
               Δ:=|X_l(T)-X_l(e_i)|;
               overlap:=(Y_l(T)≥Y_l(e_i)) und (Y_l(T)≤Y_r(e_i));
            wenn_ende
            wenn ((Δ<Δ_min) und (overlap=TRUE)) dann
               e_n,Lo:=e_i;
            wenn_ende
         sonst
            wenn (((IsHorizontal(T)=TRUE) und (X_l(T)≥X_l(e_i)) und (X_l(T)≤X_r(e_i)))
               oder ((IsHorizontal(T)=FALSE) und (Y_l(T)≥Y_l(e_i)) und (Y_l(T)≤Y_r(e_i))))
            dann
               e_n,Lo:=e_i;
            wenn_ende
         wenn_ende
         // prüfen, ob e_i näher an T liegt als eine zuvor bereits
         // ermittelte Kante e_n,Hi
         …
      wenn_ende
   wenn_ende
für_ende
gib e_n,Lo und e_n,Hi zurück
```

Das Verfahren nach Floyd und Warshall liefert nach seiner Ausführung zu jedem Knoten des Graphen den kostengünstigsten Weg zu einem beliebigen anderen Knoten des Graphen, sofern eine Verbindung existiert.

Nach Auswahl derjenigen Route, die die geringsten Kosten aufweist, bleiben der Anfangs- und der Endpunkt zu berechnen, um eine direkte Verbindung am realen Anschluss zu gewährleisten. Der erste und der letzte Wegpunkt des ermittelten Pfades leiten sich aus denjenigen Knoten des Wegenetz-Graphen ab, die die geringste Distanz zu den entsprechenden Eckpunkten der Terminals aufweisen. Bei einem horizontal verlaufenden Anschluss kann die y-Koordinate des ersten bzw. letzten Wegpunktes zur Bestimmung des neuen Wegpunktes übernommen werden. Entsprechend wird bei einem vertikalen Terminal der x-Wert verwendet. Die fehlende Koordinate berechnet sich in Abhängigkeit zum einen von der Lage des bisherigen Anfangs- bzw. Endpunkts zum Anschluss und zum anderen von der voraussichtlichen Leitungsbreite des Pfades. Im Fall eines horizontal verlaufenden Terminals berechnet sich die x-Koordinate des neuen Wegpunktes nach Gl. (6-15),

wenn der bisherige Anfangs- bzw. Endpunkt sich linksseits des Anschlusses befindet. Falls dieser Referenzpunkt rechtsseits angeordnet ist, erfolgt die Berechnung mit Hilfe von Gl. (6-16).

$$x_p = X_l(T) + W_{\text{Path}}/2, \tag{6-15}$$

$$x_p = X_r(T) + W_{\text{Path}}/2. \tag{6-16}$$

hierbei ist W_{Path} die geforderte Leitungsweite und T das entsprechende Terminal. Der x-Wert des Mittelpunkts des Terminals wird verwendet, wenn die geforderte Leitungsbreite größer als die Weite des Anschlusses ist. Die Berechnungen der Koordinaten des Wegpunktes bei einem vertikal verlaufenden Anschluss erfolgen auf analoge Weise.

Abschließend ist in der Regel ein weiterer Wegpunkt zu erstellen. Die Anschlüsse werden auf das Hüllpolygon der Zelle abgebildet. Das reale Terminal befindet sich in den überwiegenden Fällen innerhalb der Zelle. Aus diesem Grund ist eine Senkrechte zum bisherigen bzw. nachfolgenden Routenverlauf im neu erstellten Wegpunkt zu bilden. Damit ist eine unterbrechungsfreie Verbindung des Pfades sichergestellt. Im letzten Schritt werden durch eine lineare Iteration über alle Wegpunkte der ermittelten Route alle kollinearen Punkte entfernt.

6.3.2.1 Eigenschaften kürzester Wege

Seien v_i und v_j zwei Knoten aus V und W ein elementarer Weg von v_i nach v_j. Alle Knoten auf dem Weg W, die von v_i und v_j verschieden sind, werden innere Knoten genannt. Das Verfahren nach Floyd und Warshall nutzt die Beziehungen zwischen einem Weg von v_i nach v_j und den entsprechenden kürzesten Wegen mit ihren inneren Knoten aus. Genauer gesagt beschränkt sich die Suche nach kürzesten Wegen auf diejenigen, die in der k-ten Iteration lediglich innere Knoten aus der Menge $\{v_1,\ldots,v_k\} \subseteq V$ verwenden. Sei im Folgenden für zwei Knoten $v_i, v_j \in V$ der kürzeste, elementare Weg W mit den inneren Knoten aus der Menge $\{v_1,\ldots,v_k\}$ gegeben. Für einen inneren Knoten v_k, aus der Menge der möglichen inneren Knoten, ergeben sich die nachfolgenden beiden Fälle.

Wenn v_k kein innerer Knoten von W ist, dann bildet W einen kürzesten Pfad von v_i nach v_j mit inneren Knoten aus der Menge $\{v_1,\ldots,v_{k-1}\}$. Angenommen, dass W kein kürzester Weg von v_i nach v_j wäre, dann gäbe es einen Weg W' von v_i nach v_j, der nur innere Knoten aus $\{v_1,\ldots,v_{k-1}\}$ hat. Die Kosten wären echt kleiner als die des Weges W. Damit wäre jedoch der Weg W' kürzer als W bei inneren Knoten aus $\{v_1,\ldots,v_k\}$. Das ist jedoch ein Widerspruch zur Voraussetzung.

Falls andererseits v_k einen inneren Knoten von W bildet, dann kann v_k in W nur einmal auftreten, da nach Voraussetzung der Weg elementar sein soll. Der Weg W wird an der Stelle v_k in zwei Teile gespalten. Die resultierenden Teilpfade von v_i nach v_k und von v_k nach v_j sind selbst kürzeste Pfade. Mit anderen Worten sei $W = \{v_i,\ldots,v_j\}$ der kürzeste Weg vom Knoten v_i nach v_j, und $c:E \rightarrow IR$ die Kostenfunktion. Wenn für jedes Paar k, l mit $i \le k \le l \le j$ ein durch $W_{kl} = \{v_k,v_{k+1},\ldots,v_l\}$ vom Knoten v_k nach v_l gegeben ist, dann W_{kl} ist ein kürzester Pfad von v_k nach v_l. Es gilt weiterhin

$$c(W) = c(W_{ik}) + c(W_{kl}) + c\big(W_{lj}\big). \tag{6-17}$$

Angenommen, dass W_{kl} kein kürzester Weg von v_k nach v_l wäre, dann gäbe es einen kürzesten Weg $\widetilde{W}_{kl}$ mit $c(\widetilde{W}_{kl}) < c(W_{kl})$. Damit würde auch gelten:

$$c(W) = c(W_{ik}) + c(W_{kl}) + c\big(W_{lj}\big) > c(W_{ik}) + c\big(\widetilde{W}_{kl}\big) + c\big(W_{lj}\big). \tag{6-18}$$

Nach Voraussetzung ist jedoch W der kürzeste Weg von v_i nach v_j. Aus diesem Grund führt Gl. (6-18) zu einem Widerspruch. Es gilt, dass W_{kl} ein kürzester Pfad von v_k nach v_l ist.

6.3.2.2 Rekursive Berechnung der Kosten der kürzesten Wege

Basierend auf den Fallunterscheidungen des vorhergehenden Abschnitts können die Rekursionsgleichungen zur Berechnung der Kosten der kürzesten Wege aufgestellt werden. Sei dazu $c_{ij}^{(k)}$ die Kosten eines kürzesten Weges vom Knoten v_i nach v_j mit den inneren Knoten aus der Menge $\{v_1,\ldots,v_k\}$. Falls k = 0 ist, dann enthält der Pfad keine inneren Knoten und die Kosten sind gegeben durch

$$c_{ij}^{(0)} = c\left(v_i, v_j\right) \tag{6-19}$$

Damit lässt sich die Rekursionsgleichung wie folgt formulieren:

$$c_{ij}^{(k)} = \begin{cases} c\left(v_i, v_j\right) & \text{falls } k = 0 \\ \min\left\{c_{ij}^{(k-1)}, c_{ik}^{(k-1)} + c_{kj}^{(k-1)}\right\} & \text{sonst} \end{cases}. \tag{6-20}$$

Der Aufwand zur Bestimmung der Kosten des neuen kürzesten Pfades ist konstant. Es sind lediglich ein Vergleich und maximal eine Zuweisung erforderlich. Für k ≥ 1 gelten aufgrund der Zuweisung $c_{ij}^{(k)} = \min\left\{c_{ij}^{(k-1)}, c_{ik}^{(k-1)} + c_{kj}^{(k-1)}\right\}$ die nachstehenden Ungleichungen:

$$c_{ij}^{(0)} \geq c_{ij}^{(1)} \geq \cdots \geq c_{ij}^{(n)}. \tag{6-21}$$

Da die kürzesten Wege elementar sind, müssen keine weiteren Knoten betrachtet werden. Weiterhin enthält der Graph keine negativen Zyklen. Dadurch ist ein kürzester Weg von v_i nach v_j, insofern dieser existiert, stets elementar. Aus der Monotonie der $c_{ij}^{(k)}$ aus Gl. (6-21) und der Eigenschaft, dass elementare Wege höchstens n Knoten haben, folgt:

$$c_{ij}^{(n)} = c_{ij}. \tag{6-22}$$

d.h. mittels der Rekursion aus Gl. (6-20) lassen sich die Kosten berechnen, um vom Knoten v_i zum Knoten v_j zu gelangen, wenn eine Verbindungsroute existiert. Damit lässt sich ebenfalls mit einem geringen Aufwand ein Algorithmus für die Wege mit den geringsten Kosten herleiten. Sei n die Anzahl der Knoten des Graphen G=(V, E) und $C^{(0)} = \left(c_{ij}^{(0)}\right)$ eine n×n Matrix, die die Kosten enthält. Dann ergibt sich der in Algorithmus 6-7 gezeigte Ablauf zur Routensuche nach Floyd und Warshall.

Neben der Kostenmatrix C wird im Algorithmus 6-7 die Vorgängermatrix Π berechnet. Während der Iteration wird eine Folge von Matrizen $\Pi^{(0)}$, $\Pi^{(1)}$,…, $\Pi^{(n)}$ aufgestellt, wobei $\Pi = \Pi^{(n)}$ und $\pi_{ij}^{(k)}$ der Vorgänger des Knoten auf dem kürzesten Weg vom Knoten v_i mit inneren Knoten aus der Menge $\{v_1,\ldots,v_k\}$ sind. Falls der jeweilige kürzeste Weg keine inneren Knoten besitzt, d.h. der Fall k = 0 vorliegt, so wird durch $\pi_{ij}^{(0)}$ der Knoten v_i beschrieben. Wenn keine Verbindung zwischen v_i und v_j existiert, dann verweist $\pi_{ij}^{(0)}$ hingegen auf die leere Menge. Auf diese Weise kann die Initialisierung der Vorgängermatrix durch die nachfolgende Gleichung formuliert werden

$$\pi_{ij}^{(0)} = \begin{cases} \emptyset & \text{wenn i=j oder } c\left(v_i, v_j\right) = \infty \\ v_i & \text{sonst} \end{cases}. \tag{6-23}$$

wobei c:E→IR die Kostenfunktion darstellt. Für den Fall, dass k ≥ 1 gilt, sei W der kürzeste Weg vom Knoten v_i nach v_j, der den inneren Knoten v_k enthält, wobei v_k von v_j verschieden ist. Damit kann der Weg W in die beiden kürzesten Teilpfade W_1 und W_2 gegliedert werden. W_1 beschreibt den Pfad von v_i nach v_k und W_2 entsprechend dem Pfad von v_k nach v_j. Der Vorgänger von v_j im Weg W ist dabei gerade der Vorgänger von v_j in W_2. Diese Überlegungen führen zu folgender Rekursionsgleichung:

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{ij}^{(k-1)} & \text{falls } c_{ij}^{(k-1)} \leq c_{ik}^{(k-1)} + c_{kj}^{(k-1)} \\ \pi_{kj}^{(k-1)} & \text{sonst} \end{cases} . \tag{6-24}$$

wobei k ≥ 1 und $\left(c_{ij}^{(k-1)}\right) = C^{(k-1)}$ die Kostenmatrix in der (k-1)-ten Iteration sind. Die Vorgängermatrix wird benötigt, um den Verlauf der kürzesten Wege zwischen je zwei Knoten zu ermitteln. Die notwendigen Schritte zur Auswertung der Matrix werden im nächsten Abschnitt beschrieben.

Algorithmus 6-7: Routensuche der kürzesten Wege nach Floyd und Warshall

```
für i:=1 bis n
   für j:=1 bis n
      für k:=0 bis n
         c_ij^(k) := ∞;
         π_ij^(k) :=∅
      für_ende
      c_ij^(0) :=calcCost(v_i,v_j);
      wenn (i≠j) und (c_ij^(0)<∞) dann
         π_ij^(0) :=v_i;
      sonst
         π_ij^(0) :=∅;
      wenn_ende
   für_ende
für_ende
für k:=1 bis n
   für i:=1 bis n
      für j:=1 bis n
         wenn (c_ik^(k-1)=∞) oder (c_kj^(k-1)=∞) dann
            c_ij^(k) :=c_ij^(k-1);
            π_ij^(k) :=π_ij^(k-1);
         sonst
            wenn (c_ij^(k-1)>c_ik^(k-1)+c_kj^(k-1)) dann
               c_ij^(k) := c_ik^(k-1)+c_kj^(k-1);
               π_ij^(k) :=π_kj^(k-1);
            sonst
               c_ij^(k) :=c_ij^(k-1);
               π_ij^(k) :=π_ij^(k-1);
            wenn_ende
         wenn_ende
      für_ende
   für_ende
für_ende
```

6.3.2.3 Erstellung eines kürzesten Weges

Während die nxn Matrix Caus Kap. 6.3.2.2 die Kosten der kürzesten Wege zwischen zwei Knoten beschreibt, kann die Vorgängermatrix $\Pi = (\pi_{ij})$ verwendet werden, um den Verlauf der endgültigen kürzesten Wege zu ermitteln. Bei der Suche nach den möglichen, kostenoptimalen Verbindungen zwischen zwei Knoten v_i und v_j sind bei den Einträgen der Vorgängermatrix drei Fälle zu unterscheiden. Bei der ersten Möglichkeit verweist der Matrixeintrag für die Wegsuche vom Knoten v_i nach v_j auf die leere Menge. In diesem Fall existiert keine Verbindung zwischen den beiden Knoten.

Wenn andererseits für $\pi_{ij} = v_i$ gilt, dann sind die beiden Knoten v_i und v_j direkt miteinander verbunden. Es ist ferner auf diese Weise der kürzeste Weg zwischen den beiden Knoten bestimmt. Die letzte verbleibende Möglichkeit besteht darin, dass $\pi_{ij} = v_k$ mit $v_i \neq v_k$. gilt, wobei sowohl innerer Knoten des kürzesten Weges von v_i nach v_j, als auch der direkte Vorgänger von v_j ist. Es bleibt nun der direkte Vorgänger von v_k zu ermitteln und die Beziehung zwischen v_i und v_k zu untersuchen. Der Vorgang wird iterativ solange fortgesetzt, bis der Knoten v_i bzw. die leere Menge aus der Vorgängermatrix ermittelt wird. Falls ein kürzester Weg von v_i und v_j existiert, so verläuft dieser über die inneren Knoten v_1, v_2,…, v_k. Der Ablauf der Extraktion der kürzesten Wege ist zusammenfassend in Algorithmus 6-8 dargestellt.

Algorithmus 6-8: Extraktion der kürzesten Wege

```
für i:=1 bis n
   für j:=1 bis n
      vec:=∅;
      zaehler:=0;
      k:=j;
      solange (π_ik≠∅) und (π_ik≠v_i)
         vec[zaehler]:=π_ik;
         k:=indexOf(π_ik);
      solange_ende
      wenn nicht ((zaehler=0) und (c_ik=∞)) dann
         erstelle neuen Weg W mit den Kosten c_ij;
         füge v_i an das Ende von W an;
         k:=zaehler-1;
         solange k>-1
            füge Knoten vec[k] an das Ende von W an;
         solange_ende
         füge Knoten v_j an das Ende von W an;
      wenn_ende
   für_ende
für_ende
```

Die Knoten des Graphen sind von 1 bis n durchzunummerieren, mit n =$|V|$. Es wird über alle möglichen Paarungen der Knoten iteriert. Im ersten Schritt der inneren Schleife werden nach der oben beschriebenen Vorgehensweise alle inneren Knoten der Verbindung von v_i nach v_j ermittelt, sofern ein Weg zwischen den jeweiligen Knoten existiert. Die inneren Knoten werden in einem temporären Vektor gespeichert. Mit Hilfe der Knotenmatrix erfolgt im anschließenden Schritt, ob ein kürzester Weg zwischen den beiden Knoten vorhanden ist. In diesem Fall wird aus dem Anfangs- und Endknoten, sowie den inneren Knoten, ein minimaler Weg erstellt.

Es bleibt noch anzumerken, dass die Kanten des Graphen, auf dem die Wegsuche ausgeführt wird, ungerichtet sind. Dadurch ergibt sich eine symmetrische Darstellung der Kosten- und Vorgängermatrix in Bezug auf die Hauptdiagonale. Aufgrund dieser Tatsache, werden für beide Matrizen echte obere Dreiecksmatrizen verwendet. Neben einer Ersparnis im Hinblick auf den benötigten Speicher, ergibt sich ebenso eine Verringerung der Laufzeit. Dennoch weist die Routensuche nach Floyd und Warshall eine Laufzeit in der Größenordnung $\Theta(n^3)$ auf.

6.4 Layouterstellung eines Pfades

In den vorherigen Abschnitten ist die Suche nach möglichen Verbindungen zwischen je zwei Anschlüssen bei einer vorgegebenen Anordnung von Zellen erläutert worden. Bis zu diesem Zeitpunkt ist ein Pfad lediglich durch eine Folge von Wegpunkten charakterisiert. Die Schichtzuweisung ist bislang vernachlässigt worden. Für die Verdrahtung stehen L stromführende

Schichten zur Verfügung. Die Anzahl L ist technologieabhängig. Im Nachfolgenden werden die Schritte zur Erstellung einer Geometrie der Verbindungswege, basierend auf der Aneinanderreihung vertikaler bzw. horizontaler Wegstücke, beschrieben. Für jede Strecke zwischen zwei aufeinanderfolgenden Wegpunkten wird ein Pfadsegment erzeugt. Es handelt sich dabei grundlegend um ein die Wegpunkte umhüllendes Rechteck. Ferner ist jedem Segment eines Netzes eine stromführende Schicht zuzuweisen.

In einem Segment werden der Layer des aktuellen Wegstücks, die Leitungsbreite sowie die Informationen über das Netz gespeichert. Es enthält ferner Zeiger auf die zugehörigen Wegpunkte und ggf. auf Vias, die sich an einem der Wegpunkte befinden können. Mit Hilfe der Vias ist eine Verbindung zu einem nachfolgenden bzw. vorhergehenden Segment mit einem anderen Layer aber dem gleichen Knotenpotential gewährleistet. Die Wegpunkte liegen in der Regel nicht auf dem Rand des Segmentrechtecks, sondern werden von dem Rechteck umhüllt. Lediglich der erste und der letzte Wegpunkt befinden sich auf der Mitte der Stirnseite des jeweiligen Segmentrechtecks. Der Grund für diese Vorgehensweise wird im weiteren Verlauf des Abschnitts näher erläutert. Der Abstand zum Rand entspricht der Hälft der Leitungsbreite (siehe Abb. 6-7 (b)). Die Folge der Segmente eines Pfades ist damit vergleichbar mit einem Gliedermaßstab, wobei die Wegpunkte die jeweiligen Achsen bilden. Operationen, wie das Strecken, Stauchen oder Verschieben eines Segments, vereinfachen durch die beschriebene Modellierung die Aktualisierung der nachfolgenden bzw. vorhergehenden Segmente. Ebenso wird die Erstellung von Vias vereinfacht. Es liegt auf diese Weise bereits ein Überlappungsgebiet zwischen zwei aufeinanderfolgenden Segmenten vor, in welches die Vias zur Verbindung eingefügt werden können. Weitere Vorteile ergeben sich bei der Entwurfsregelprüfung, die sich der Layouterstellung des Pfades anschließt. Zur Prüfung der Entwurfsregeln wird das in [Wolf96] vorgestellte Kompaktierungsverfahren genutzt. Dafür ist es notwendig, die geometrischen Elemente des Pfades auch durch ihr Kantenbild darstellen zu können. Die Funktionen zur Erstellung und Speicherung eines Kantenbildes sind ebenfalls in jedem Segment enthalten. Es wird neben den Kanten des Rechtecks und diejenigen von Vias erfasst, die in einem Segment enthalten sein können.

Weiterhin wird in einem Segment die Verdrahtungsrichtung gespeichert. Diese wird durch die Kardinalspunkt Nord, Ost, Süd, West charakterisiert und kann aus der Lage der beiden Wegpunkte eines Segments zueinander ermittelt werden.

Insgesamt dienen als Eingabeparameter für die Layouterstellung, neben der Liste der Wegpunkte, das Start- und Zielterminal, sowie die gewünschte Mindestbreite des Pfades. Der Wert für die Leitungsbreite kann mit Hilfe einer DC Simulation der Schaltung bestimmt werden. Dazu wird der maximal zu erwartende Strom für jedes Netz ermittelt. Nach [Lien03] ist es damit möglich, die minimal zulässige Pfadbreite wie folgt zu berechnen:

$$w_{\min} = \frac{|I_{\max}| \cdot s}{t_{\text{Layer}} \cdot J_{\max}(T_{\text{ref}})}, \tag{6-25}$$

wobei I_{max} der aus der Simulation ermittelte größte zu erwartende Strom und s ein Sicherheitsfaktor mit einem typischen Wert von $1.1 \leq s \leq 1.2$ sind. Es werden ferner die maximal zulässige Stromdichte J_{max} für eine bestimmte Leitungstemperatur T_{ref} und die Dicke der Leiterbahn t_{Layer} miteinbezogen. Die Werte für die letztgenannten Parameter sind den Angaben des jeweiligen Technologie-Anbieters zu entnehmen. Wenn keine Angaben zum maximalen Strom gemacht werden bzw. der aus Gl. (6-25) resultierende Wert den prozessabhängigen Mindestwert unterschreitet, wird für die Pfadbreite der in der Technologie definierte minimale Wert benutzt.

Der Ablauf für die Layouterstellung eines Pfades ist in Algorithmus 6-9 dargestellt.

Algorithmus 6-9: Vorgehensweise für die Layouterstellung eines Pfades

```
wenn |WayPoints|>2 dann
   für i:=2 bis |WayPoints|
      wenn i=2 dann
         erstelle Segment für die Verbindung Terminal-Wegpunkt;
         wenn L(T_A)≠L(T_B) dann
            erstelle ein Via an der Stelle WayPoints[2], um von der Maske L(T_A)
            auf die Maske L(T_B) zu wechseln;
         wenn_ende
      sonst wenn i=|WayPoints| dann
         erstelle Segment für die Verbindung Wegpunkt-Terminal;
      sonst
         erstelle Segment für die Verbindung Wegpunkt-Wegpunkt;
      wenn_ende
   für_ende
sonst
   wenn L(T_A)=L(T_B) dann
      erstelle Segment, dass die beiden Terminals T_A und T_B verbindent;
   sonst
      erzeuge einen neuen Wegpunkt v_neu zwischen dem ersten und letzten Wegpunkt;
      Segment für die Verbindung von T_A nach v_neu mit L(T_A) erstellen;
      Segment für die Verbindung von v_neu nach T_B mit L(T_B) erstellen;
      an der Stelle v_neu ein Via generieren;
   wenn_ende
wenn_ende
```

Zu Beginn des Verfahrens ist die Anzahl der Wegpunkte festzustellen. Konstruktionsbedingt besteht ein Pfad aus mindestens zwei Wegpunkten, nämlich dem Start- und dem Endpunkt, die aus den entsprechenden Terminals hervorgehen. Es wird bei der Layouterstellung zwischen Pfaden mit genau zwei Wegpunkten und mehr als zwei Punkten unterschieden. Bei n Wegpunkten sind lediglich n-1 Pfadsegmente zu erstellen. Aus diesem Grund beginnt die Iteration erst ab dem zweiten Index. Es werden die Segmente zwischen dem aktuellen Wegpunkt und dessen Vorgänger erzeugt. Der erste und der letzte Durchlauf der Schleife werden gesondert behandelt. Beim Segment zwischen dem Startpunkt und seinem Nachfolger befindet sich der erste Wegpunkt auf dem Rand des Segmentrechtecks (siehe Abbildung 6-6 (b)). Analog dazu befindet sich beim letzten Segment der Endpunkt des Pfades auf dem Segmentrand (siehe Abbildung 6-6 (a)).

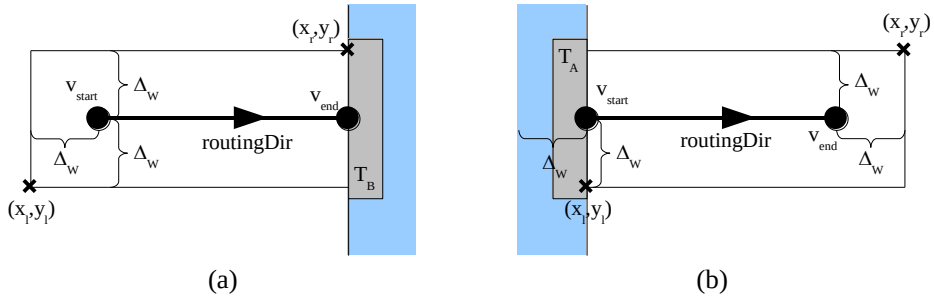


Abbildung 6-6: Erstellen eines Segments für die Verbindung (a) Wegpunkt-Terminal und (b) Terminal-Wegpunkt

Die Unterscheidung ist notwendig, um einen unerwünschten Überhang der Pfadsegmente in die entsprechende Zelle zu vermeiden. Eine Koordinate des ersten bzw. letzten Wegpunktes ist identisch mit dem jeweiligen Terminalwert. Ein Terminal wird durch eine Kante dargestellt. Die Kante ihrerseits ist normalerweise durch zwei Punkte definiert, bei einer vertikalen Kante durch einen oberen und unteren Punkt und bei einer horizontalen Kante entsprechend durch einen linken und rechten Punkt.

Da in beiden Fällen eine Koordinate identisch ist, kann jede Kante durch ein Dreier-Tupel eindeutig beschrieben werden. Eine horizontale Kante mit den Punkten P1 und P2 wird damit durch das Tripel $e_H = \{x_{P1}, y_{P1}, x_{P2}\}$ identifiziert. Bei einem horizontal verlaufenden Terminal stimmt somit die y-Koordinate des Start- bzw. Endpunkts mit der des Terminals überein. Entsprechend ist die x-Koordinate des ersten bzw. letzten Wegpunktes mit der des entsprechenden vertikalen Terminals identisch. Das Terminal selber ist Teil eines Rechtecks innerhalb der Zelle. Um eine direkte Verbindung zwischen dem Terminal und dem zugehörigen Pfad sicherzustellen, wird die genannte Unterscheidung bei der Segmenterstellung durchgeführt. Die Koordinaten für die in Abbildung 6-6 dargestellten Segmentrechtecke, mit der Verdrahtungsrichtung Ost, sind in Gl. (6-26) bzw. Gl. (6-27) beschrieben. Die Berechnung der Eckpunkte für die verbleibenden drei Richtungen erfolgt auf analoge Weise.

$$\begin{matrix} x_l = x_{vstart} - \Delta_W & x_r = x_{vend} = x_{TB} \\ y_l = y_{vstart} - \Delta_W & y_r = y_{vend} + \Delta_W \end{matrix}, \tag{6-26}$$

$$\begin{matrix} x_l = x_{vstart} = x_{TA} & x_r = x_{vend} + \Delta_W \\ y_l = y_{vstart} - \Delta_W & y_r = y_{vend} + \Delta_W \end{matrix}, \tag{6-27}$$

wobei Δ_W die Hälfte der Leitungsbreite, x_{TA} die x-Koordinate des Startterminals und die x_{TB} x-Koordinate des Zielterminals darstellen.

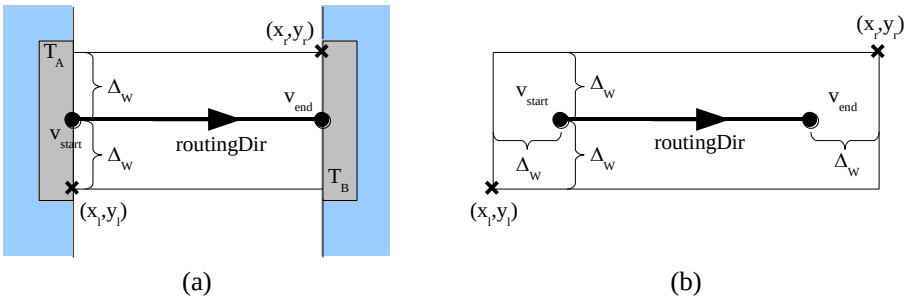


Abbildung 6-7: Erstellen eines Segments für die Verbindung (a) Terminal-Terminal und (b) Wegpunkt-Wegpunkt

Die Konstruktion der Segmente für zwei Wegpunkte, die weder der Start- noch der Endpunkt des Pfades sind, ist in Abbildung 6-7 (b) gezeigt. Die Eckpunkte des Rechtecks für den dargestellten Fall, dass die Verdrahtung in Richtung Ost erfolgt, sind der nachfolgenden Gl. (6-28) zu entnehmen. Die anderen Richtungen lassen sich auf die gleiche Weise berechnen.

$$\begin{matrix} x_l = x_{vstart} - \Delta_W & x_r = x_{vend} + \Delta_W \\ y_l = y_{vstart} - \Delta_W & y_r = y_{vend} + \Delta_W \end{matrix}, \tag{6-28}$$

wobei Δ_W erneut die halbe Pfadbreite repräsentiert. Wenn sich die Layer des Start- und Zielterminals unterscheiden, wird im ersten Schleifendurchlauf ein Via an der Stelle des zweiten Wegpunktes erstellt. Das erste Segment gehört damit dem gleichen Layer wie das Startterminal an, während bei allen nachfolgenden Segmenten der Layer mit dem des Zielterminals übereinstimmt.

Sofern der Pfad sich aus lediglich zwei Wegpunkten zusammensetzt, werden die im Folgenden beschriebenen Schritte ausgeführt. Bei einer Übereinstimmung der Layer des Start- und Zielterminals ergibt sich ein Pfadsegment, wie in Abbildung 6-7 (a) für den horizontalen Verdrahtungsfall dargestellt ist. Die Koordinaten der Eckpunkte des umhüllenden Rechtecks sind durch Gl. (6-29) gegeben. Die Berechnung der drei weiteren Fälle erfolgt analog.

$$\begin{aligned} x_l &= x_{vstart} = x_{TA} \quad x_r = x_{vend} = x_{TB} \\ y_l &= y_{vstart} - \Delta_W \quad y_r = y_{vend} + \Delta_W \end{aligned}, \tag{6-29}$$

wobei x_{TA} die x-Koordinate des Startterminals, x_{TB} die x-Koordinate des Zielterminals und Δ_W die halbe Leitungsweite repräsentieren.

Falls andererseits die Layer der beiden Terminals verschieden sind, wird zunächst ein weiterer Wegpunkt zwischen den beiden bisherigen Punkten eingefügt. Die Lage des neuen Punktes P ist für eine horizontale Verdrahtungsrichtung durch Gl. (6-30) und im vertikalen Fall durch Gl. (6-31) gegeben.

$$x_p = (x_{vstart} + x_{vend})/2 \quad y_p = y_{vstart} = y_{vend}, \tag{6-30}$$

$$x_p = x_{vstart} = x_{vend} \quad y_p = (y_{vstart} + y_{vend})/2. \tag{6-31}$$

Anschließend werden die Segmente für die Wegstrecke zwischen einem Terminal und einem Wegpunkt sowie dem umgekehrten Fall generiert. Die Berechnung der Koordinaten erfolgt nach der in Gl. (6-26) und Gl. (6-27) gezeigten Vorgehensweise. Es bleibt noch ein Via an der Stelle des neu erzeugten Wegpunkts zu erstellen, um die Verbindung zwischen den beiden Segmenten sicherzustellen.

6.4.1 Erstellen einer Durchkontaktierung

Die vertikale elektrische Verbindung der Leiterbahnen wird mittels einer Durchkontaktierung bzw. einem Via (Vertical Interconnect Access) erzielt. Als Eingabeparameter sind der Anfangs und Ziellayer, die gewünschten Leitungsweite und der Mittelpunkt, um den das Via aufgebaut werden soll, anzugeben. Es wird ferner davon ausgegangen, dass die einzusetzende Technologie die Verwendung von Stacked-Vias erlaubt. Unter Stacked-Vias wird die Platzierung von mehreren Vias unmittelbar übereinander verstanden, was einem mehrfachen Ebenenwechsel, beispielsweise von der zweiten zur vierten Metall-Lage, entspricht. Der Ablauf für die Generierung einer Durchkontaktierung ist im nachfolgenden Algorithmus 6-10 skizziert.

Algorithmus 6-10: Generierung eines Vias

```
q1:=CreateSquare(Layer_start);
List_Layer:=IntermediateLayer(Layer_start,Layer_end);
für i:=1 bis |List_Layer|
   q2:=CreateSquare(List_Layer[i]);
   CreatContacts(q1,q2);
   q1:=q2;
für_ende
q2:=CreateSquare(Layer_end);
CreateContacts(q1,q2);
```

Im ersten Schritt wird um den Mittelpunkt ein Quadrat mit der Maske $Layer_{start}$ und der Kantenlänge der gewünschten Leitungsweite erstellt. Die Leitungsbreite wird in der Methode CreateSquare vor Erstellung des Quadrats dahingehend geprüft, ob der in der Technologie definierte Mindestwert für die jeweilige Maske eingehalten wird. Falls der Wert unterschritten wird, erfolgt automatisch eine Anpassung auf das vorgegebene Minimum. Nach Erstellen des ersten geometrischen Elements werden diejenigen Layer bestimmt, die sich zwischen dem Start- und dem Ziellayer befinden und in der Liste $List_{Layer}$ gespeichert. In einem Schleifendurchlauf über die ermittelten Layer wird in jedem Schritt zunächst ein Quadrat mit der jeweiligen Maske erzeugt. Daran schließt sich die Generierung der Kontakte an, die die beiden Ebenen miteinander verbinden sollen. Dazu ist zunächst jeweils die

Mindestüberlappung zwischen den Kontakten und den beiden Quadraten zu ermitteln. Im weiteren Verlauf wird der betragsmäßig größere Wert ausgewählt. Damit kann nach Gl. (6-32) und Gl. (6-33) die Anzahl der Kontakte in horizontaler und vertikaler Richtung berechnet werden.

$$\#\mathrm{Cont}_x = \frac{a-(2\cdot \mathrm{ovlp})+\Delta_{\mathrm{cnt}}}{L_{\mathrm{cnt}}+\Delta_{\mathrm{cnt}}}, \tag{6-32}$$

$$\#\mathrm{Cont}_y = \frac{a-(2\cdot \mathrm{ovlp})+\Delta_{\mathrm{cnt}}}{W_{\mathrm{cnt}}+\Delta_{\mathrm{cnt}}}. \tag{6-33}$$

wobei a die Kantenlänge des kleineren Quadrats, ovlp der zuvor ermittelte Überlappungswert, Δ_{cnt} der Mindestabstand zwischen zwei Kontakten und L_{cnt} sowie W_{cnt} die minimalen Abmessungen eines Kontaktes sind. Wenn infolge der Entwurfsregeln kein Kontakt eingefügt werden kann, sind die beiden Quadraten nach Gl. (6-34) und Gl. (6-35) um dX bzw. dY zu erweitern.

$$dX = L_{\mathrm{cnt}} + 2\cdot \mathrm{ovlp} - a, \tag{6-34}$$

$$dY = W_{\mathrm{cnt}} + 2\cdot \mathrm{ovlp} - a. \tag{6-35}$$

Falls die Möglichkeit gegeben ist, mehr als ein Kontakt in horizontaler bzw. vertikaler Richtung einzufügen, bleibt der tatsächliche Abstand zwischen den Kontakten zu berechnen (siehe Gl. (6-36) und Gl. (6-37)).

$$\mathrm{dist}_x = \frac{a-(2\cdot \mathrm{ovlp})-(\#\mathrm{Cont}_x\cdot L_{\mathrm{cnt}})}{\#\mathrm{Cont}_x-1}, \tag{6-36}$$

$$\mathrm{dist}_y = \frac{a-(2\cdot \mathrm{ovlp})-(\#\mathrm{Cont}_y\cdot W_{\mathrm{cnt}})}{\#\mathrm{Cont}_y-1}, \tag{6-37}$$

Die Startkoordinaten des ersten Kontakts sind durch die nachfolgenden Gl. (6-38) und Gl. (639) gegeben.

$$x_s = \Big(x_{Q,l} + x_{Q,r} - \big(\#\mathrm{Cont}_x \cdot (L_{\mathrm{cnt}} + \mathrm{dist}_x)\big)\Big)/2, \tag{6-38}$$

$$y_s = \Big(y_{Q,l} + y_{Q,r} - \big(\#\mathrm{Cont}_y \cdot \big(W_{\mathrm{cnt}} + \mathrm{dist}_y\big)\big)\Big)/2. \tag{6-39}$$

wobei ($x_{Q,l}$, $y_{Q,l}$) und ($x_{Q,r}$, $y_{Q,r}$) die Eckpunkte des kleineren Quadrats beschreiben. Die Kontakte werden äquidistant und symmetrisch in die Quadrate eingefügt. Auf diese Weise wird eine vertikale Verbindung ausgehend vom Startlayer hin zum Ziellayer um einen gegebenen Mittelpunkt erreicht.

6.5 Prüfung der Entwurfsregeln und der elektrischen Regeln

Jede Leitung weist eine Mindestbreite und einen Mindestabstand zu Leitungen der gleichen Schicht auf. Beide Größen sind technologieabhängig. Bei der Layouterstellung der Pfade wird zwar auf eine Einhaltung der Entwurfsregeln in Bezug auf die Mindestbreite der jeweiligen Pfade geachtet, jedoch können Verletzungen der Mindestabstände zu benachbarten Zellen bzw. Pfaden auftreten. Aus diesem Grund schließt sich der Konstruktionsphase für die Pfade eine Entwurfsregelprüfung an, die nicht nur die Fehler entdeckt, sondern sie auch - wenn möglich automatisch korrigiert.

Neben dem genannten Unterschreiten von prozessbedingten Mindestabständen, können Kurzschlüsse von Leitungen auftreten. Diese liegen vor, wenn sich beispielsweise zwei Leitungen mit unterschiedlichen Knotenpotentialen, aber identischen Zeichnungslayern, kreuzen bzw. überlappen.

Im nachfolgenden Abschnitt wird die Suche nach möglichen Regelverletzungen beschrieben, während in Kap. 6.5.2 die notwendigen Korrekturen erläutert werden.

6.5.1 Auffinden von Regelverletzungen

Jeder Pfad wird durch eine Menge an Wegpunkten und einer Folge von geraden Leitungselementen, die je zwei aufeinanderfolgende Wegpunkte umschließen, dargestellt. Zusätzlich können an den Symmetriepunkten Durchkontaktierungen vorhanden sein, die eine elektrische Verbindung zweier Segmente, mit verschiedenen Layern, am gemeinsamen Wegpunkt sicherstellen. Die Pfade, Leitungselemente und Vias werden durch die gleiche Datenstruktur wie die Zellen mit den entsprechenden Kantenbildern dargestellt. Dadurch kann der Kompaktierungsalgorithmus zur Berechnung der Mindestabstände und des ggf. erforderlichen Verschiebungsvektors genutzt werden.

Algorithmus 6-11: Suche nach benachbarten Zellen und Pfadsegmenten

```
List_objects in Abhängigkeit von der Suchrichtung searchDir sortieren;
wenn ((searchDir=NORTH) oder (searchDir=SOUTH)) dann
   Limit_Low:=MaxWest(Element_ref);
   Limit_Hi:=MaxEast(Element_ref);
sonst
   Limit_Low:=MaxSouth(Element_ref);
   Limit_Hi:=MaxNorth(Element_ref);
wenn_ende
Listresults:=∅;
für alle element∈List_objects
   Limit_Element,Low:=∅;
   Limit_Element,Hi:=∅;
   canTest:=FALSE;
   wenn ((searchDir=NORTH) oder (searchDir=SOUTH)) dann
      wenn (((MaxNorth(element)>MaxSouth(Element_ref)) und (searchDir=NORTH))
         oder ((MaxSouth(element)<MaxNorth(Element_ref)) und (searchDir=SOUTH)))
      dann
         Limit_Element,Low:=MaxWest(element);
         Limit_Element,Hi:=MaxEast(element);
         canTest:=TRUE;
      wenn_ende
   sonst
      wenn (((MaxEast(element)>MaxWest(Element_ref)) und (searchDir=EAST))
         oder ((MaxWest(element)<MaxEast(Element_ref)) und (searchDir=WEST)))
      dann
         Limit_Element,Low:=MaxSouth(element);
         Limit_Element,Hi:=MaxNorth(element);
         canTest:=TRUE;
      wenn_ende
   wenn_ende
   wenn (canTest=TRUE) dann
      wenn (nicht((Limit_Element,Hi<Limit_Low) oder (Limit_Element,Low>Limit_Hi))) dann
         List_results:=List_results∪element;
      wenn_ende
      wenn (Limit_Element,Hi>Limit_Hi) dann
         Limit_Hi:=Limit_Element,Hi;
      wenn_ende
      wenn (Limit_Element,Low<Limit_Low) dann
         Limit_Low:=Limit_Element,Low;
      wenn_ende
   wenn_ende
für_ende
gib List_results zurück;
```

Im Rahmen der Prüfung dieser Entwurfsregeln sind die zu untersuchenden Pfade im Hinblick auf die Anzahl ihrer Pfadsegmente aufsteigend sortiert. Bei einer identischen Anzahl der Leitungselemente werden der Leitungstyp und schließlich die Priorität als Vergleichskriterium hinzugezogen. Ferner

werden die Geometrien der Zellen und der vorhandenen Pfade in einer sortierbaren Liste zusammengefasst. Anschließend wird über die Menge der sortierten Pfade zum Auffinden und Korrigieren der Regelverletzungen iteriert. Für jeden zu prüfenden Pfad werden alle seine Leitungselemente, in der Reihenfolge ausgehend vom Startterminal, einzeln betrachtet. Zu jedem zu untersuchenden Pfadsegment $Element_{ref}$ werden in jeder Richtung die benachbarten Elemente ermittelt. Der Ablauf der Suche ist in Algorithmus 6-11 skizziert. Zunächst wird die Menge der Geometrien, bestehend aus den Zellen und Leitungselementen der bereits erstellten und verifizierten Pfade, in Abhängigkeit von der Suchrichtung sortiert. Dabei gilt beispielsweise für den Fall einer in nördlicher Richtung verlaufenden Suche, dass alle Objekte bzgl. der kleinsten y-Koordinate aufsteigend sortiert werden. Bei einer Übereinstimmung zwischen zwei oder mehr Elementen wird die kleinste x-Koordinate als Ordnungskriterium ausgewählt. Für eine westliche Suchrichtung wird die Menge der Zellen und Leitungselemente nach der jeweiligen maximalen x-Koordinate bzw. bei Vorliegen einer Gleichheit nach dem maximalen y-Wert absteigend sortiert. Auf analoge Weise erfolgt die Sortierung für die beiden verbleibenden Suchrichtungen. Bedingt durch die Tatsache, dass jedes Objekt, wie Module, Leitungselemente oder Vias, von einer Zellgeometrie abgeleitet ist, können die Extrema durch die Methoden MaxNorth, MaxEast, MaxSouth und MaxWest abgefragt werden. Dabei werden die Kantenbilder zur entsprechenden Seite für die Suche des geforderten Extremalwertes ausgewertet. Im Anschluss werden die vorläufigen Grenzen, innerhalb denen die benachbarten Elemente gesucht werden, in Abhängigkeit von der Suchrichtung initialisiert. Anschließend werden alle zuvor sortierten geometrischen Objekte durchlaufen. Für jeden Eintrag werden abhängig von der Suchrichtung die Elementgrenzen bestimmt. Wenn eine Überlappung der Grenzwerte vorliegt, wird das Objekt der Ergebnisliste hinzugefügt. Es wird ferner geprüft, ob der bisherige Suchradius durch das ermittelte Elemente zu erweitern ist. Auf diese Weise werden alle Einträge ermittelt, die mit dem Leitungselement $Element_{ref}$ benachbart sind, sowie alle Objekte die wiederum Nachbarn des aktuellen Elements sind. Infolge der Sortierung kann die Menge der Zellen und Pfadsegmente linear durchlaufen werden. Nach Abschluss der Iteration stehen in einer Liste alle Objekte zu einer Seite der zu untersuchenden Pfadsegments $Element_{ref}$ zur Verfügung, die von einer Verschiebeoperation mit betroffen sein können.

Algorithmus 6-12: Suche nach zu $Element_{ref}$ parallelen Leitungen

```
wenn (IsHorizontal(RoutingDir(Element_ref))=TRUE) dann
   List_neighbors,1:=Objekte aus der Suche nach Nachbarn in nördlicher Richtung;
   List_neighbors,2:=Objekte aus der Suche nach Nachbarn in südlicher Richtung;
sonst
   List_neighbors,1:=Objekte aus der Suche nach Nachbarn in östlicher Richtung;
   List_neighbors,2:=Objekte aus der Suche nach Nachbarn in westlicher Richtung;
wenn_ende
List_par:=∅;
für i:=1 bis 2
   für alle element∈List_neighbors,i
      wenn (type(element)≠Segment) dann
         Abbruch;
      wenn_ende
      wenn (RoutingDir(element)||RoutingDir(Element_ref)) dann
         List_par:=List_par∪element;
      wenn_ende
   für_ende
für_ende
```

Es sind für die Prüfung der Regeln zwei weitere Mengen zu bestimmen. Zum einen ist eine Liste von zu $Element_{ref}$ parallel verlaufenden Leitungen zu ermitteln, die dem gleichen Verdrahtungskanal

angehören. Dazu werden die beiden Mengen der benachbarten Elemente ausgewertet, die aus der Suchrichtung hervorgehen und die senkrecht zur Verdrahtungsrichtung von $Element_{ref}$ verlaufen. Jedes Leitungselement eines Pfades enthält eine Angabe zur Verdrahtungsrichtung. Diese ergibt sich aus der Lage der beiden Wegpunkte, die vom Leitungselement umhüllt werden, zueinander. Die Verbindungsstrecke der beiden Punkte ist konstruktionsbedingt rechtwinklig im Verlauf. Somit kann aus der Reihenfolge und der Lage der beiden Punkte die Verdrahtungsrichtung abgeleitet werden. Nachdem die beiden Listen bestimmt worden sind, wird jede linear durchlaufen. Solange das aktuelle Objekt ein Leitungselement darstellt, dessen Verdrahtungsrichtung parallel zu $Element_{ref}$ verläuft, wird dieses in die Menge der parallelen Leitungen aufgenommen. Andernfalls erfolgt der Abbruch der Schleife. Der Ablauf zur Suche der parallel verlaufenden Leitungselemente ist in Algorithmus 6-12 dargestellt. Als Eingabe dienen die Menge der sortierten Nachbarschaftselemente und das Leitungselement $Element_{ref}$, für das die parallelen Leitungen zu bestimmen sind.

Abschließend ist eine Menge von Leitungen, die von der Leitung $Element_{ref}$ gekreuzt werden, aufzubauen. Die Vorgehensweise ist vergleichbar mit der Bestimmung der parallelen Leitungen in Algorithmus 6-12. Es werden diejenigen Listen der benachbarten Objekte gesucht, die aus der Suchrichtung hervorgehen, die nicht orthogonal zur Verdrahtungsrichtung von $Element_{ref}$ verlaufen. Jede der beiden Mengen wird ebenfalls linear durchlaufen. Sobald es sich beim aktuellen Element nicht um ein Leitungssegment handelt, wird die Iteration abgebrochen. Wenn die Verdrahtungsrichtung des jeweiligen Leitungselements orthogonal zu der von $Element_{ref}$ ausgerichtet ist, bleibt zu prüfen, ob die beiden Elemente sich kreuzen. Dieser Test kann auf einen Schnitt von Rechtecken zugeführt werden. Für jedes der beiden Leitungssegmente wird ein Rechteck als äußere Hülle angenommen. Liegt ein Schnitt bzw. eine Überlappung der beiden Hüllrechtecke vor, wird das Leitungselement der Menge der gekreuzten Leitungen hinzugefügt. Andernfalls ergibt sich eine der in Abbildung 6-8 dargestellten Lagebeziehungen der beiden Leitungen.

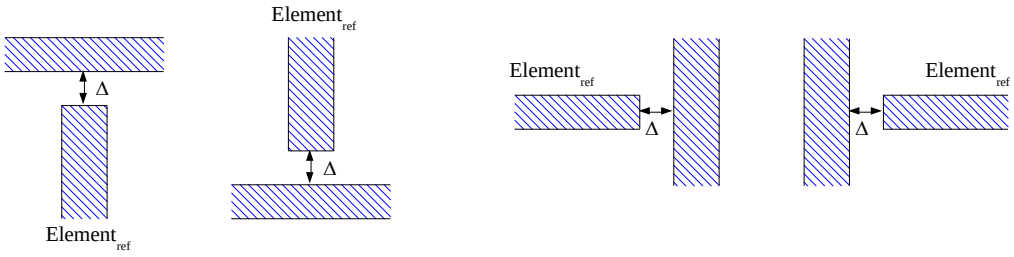


Abbildung 6-8: Mögliche Lage zweier Leitungselemente bei der Suche nach gekreuzten Leitungen

In diesem Fall ist mit Hilfe der Kantenbilder der beiden Elemente und des Kompaktierungsalgorithmus der Abstand zwischen den beiden Segmenten zu berechnen. Ist dieser kleiner als ein technologiebedingtes bzw. durch eine Nebenbedingung definiertes Minimum, wird das betroffene Leitungselement ebenfalls in die Liste der gekreuzten Leitungen aufgenommen.

Eine Nebenbedingung geht aus der Kapazitäts-Sensitivitätsmatrix hervor. Überlappende geometrische Elemente auf unterschiedlichen elektrischen Potentialen erzeugen zusätzliche parasitäre Kapazitäten. In Abhängigkeit von der Funktion der Zellblöcke können diese zusätzlichen Kapazitäten gewünscht sein, wie beispielsweise bei einem Stromspiegel eines Referenznetzwerks. Hier können parasitäre Kapazitäten zur Spannungsstabilisierung genutzt werden. Zur Kontrolle dieser Kapazitäten wurde in [Wolf97] eine Kapazitäts-Sensitivitätsmatrix für die Modulgeneratorumgebung MOGLAN eingeführt. Auf diese Weise können Überlappungen zwischen verschiedenen Netzen

definiert bzw. verhindert werden. Falls keine Überlappungen erwünscht sind bzw. ein größerer Abstand eingehalten werden soll, werden diese Vorgaben bei der Positionierung der Leitungen mit berücksichtigt. Es werden dabei Leitungselemente mit unterschiedlichen Layern derart angeordnet, dass keine Überlappungen auftreten.

6.5.2 Korrektur von Regelverletzungen

Im vorhergehenden Abschnitt ist die Suche nach benachbarten Objekten zu einem Pfadsegment beschrieben worden. Aus den resultierenden Mengen ist nun die horizontale bzw. vertikale Verschiebungsrichtung zu bestimmen. Dazu können die nachfolgenden Gleichungen verwendet werden. Es ist

$$m_{\mathrm{Horz}} = \begin{cases} \mathrm{WEST} & \text{falls } \sum_{i=1}^{|\mathrm{List_{West}}|} \alpha_i A(\mathrm{List_{West}}[i]) < \sum_{i=1}^{|\mathrm{List_{East}}|} \alpha_i A(\mathrm{List_{East}}[i]) \\ \mathrm{EAST} & \text{sonst} \end{cases}, \tag{6-40}$$

$$m_{\mathrm{Horz}} = \begin{cases} \mathrm{SOUTH} & \text{falls } \sum_{i=1}^{|\mathrm{List_{South}}|} \alpha_i A(\mathrm{List_{South}}[i]) < \sum_{i=1}^{|\mathrm{List_{North}}|} \alpha_i A(\mathrm{List_{North}}[i]) \\ \mathrm{NORTH} & \text{sonst} \end{cases}, \tag{6-41}$$

wobei $\mathrm{List_{North}}$, $\mathrm{List_{East}}$, $\mathrm{List_{South}}$ und $\mathrm{List_{West}}$ die sortierten Mengen der zum aktuellen Leitungselement benachbarten Objekte darstellen. Die Funktion A: {Element}→IR berechnet für ein geometrisches Element dessen Flächeninhalt. Der Wert α_i ist ein Gewichtungsfaktor für das jeweilige Objekt. Auf diese Weise wird die Verschiebungsrichtung nicht nur durch die Größe und Anzahl der Objekte festgelegt. Für die Geometrien der Module wird die Gewichtung direkt durch den Benutzer vorgegeben. Bei einem Leitungselement leitet dieser Wert sich aus der Priorität des zugehörigen Pfades ab.

Bei der Layouterstellung der Pfade ist bislang der Abstand zu den benachbarten Zellen bzw. Pfaden nicht berücksichtigt worden. Dadurch können Verletzungen der Minimalabstand - Regel auftreten, wie in Abbildung 6-9(a) zu sehen ist. In dem Beispiel liegt zum einen eine Überlappung des Leitungselements $\mathrm{Element_{ref}}$ mit einem Objekt $\mathrm{Element_i}$ vor, zum anderen wird der minimale Abstand zwischen dem Leitungselement $\mathrm{Element_{ref}}$ und dem Objekt $\mathrm{Element_j}$ unterschritten. Für das vorliegende Beispiel erfolgt die Verschiebung in östlicher Richtung, das Objekt $\mathrm{Element_i}$ beschreibt eine Zelle und $\mathrm{Element_j}$ repräsentiert ein Leitungselement mit dem gleichen Layer wie $\mathrm{Element_{ref}}$, jedoch mit einem anderen Knotenpotential. Die Fälle für die anderen Verschiebungsrichtungen ergeben sich entsprechend. Ebenso sind für die Untersuchung eines Leitungssegments im Hinblick auf Abstandsverletzungen lediglich zwei Seiten des Elements zu prüfen. Wenn das Verdrahtungselement senkrecht verläuft, sind alle Objekte, die sich links- bzw. rechtsseits befinden, zu untersuchen. Entsprechend sind bei einer horizontalen Verdrahtungsrichtung alle Elemente oberhalb bzw. unterhalb des Segments zu berücksichtigen. Infolge der Überlappung zweier aufeinanderfolgender Pfadsegmente am gemeinsamen Wegpunkt werden die Abstände zu den geometrischen Objekten der beiden verbleibenden Richtungen durch das Nachfolgesegment des aktuellen Leitungssegments geprüft und ggf. berichtigt.

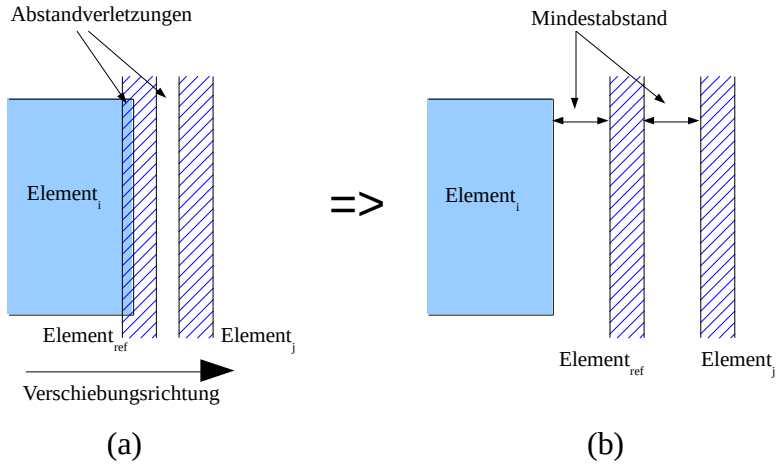


Abbildung 6-9: Abstandverletzungen benachbarter Objekte (a) und nach der Korrektur der Regelverletzung (b)

Mit Hilfe der Kantenbilder der Elemente und dem Kompaktierungsalgorithmus wird der minimal einzuhaltende Abstand zwischen dem Leitungselement Element_{ref} und dem Objekt Element_i bestimmt. Der Verschiebungsvektor berechnet sich in diesem Fall nach der folgenden Gleichung

$$\text{mv}_1 = \text{minDist}(\text{Element}_i, \text{Element}_{ref}) + X_r(\text{Element}_i) - X_l(\text{Element}_{ref}), \qquad (6\text{-}42)$$

wobei $\text{minDist}(\text{Element}_i, \text{Element}_{ref})$ die Minimaldistanz zwischen dem Objekt Element_i und Element_{ref} beschreibt. Die Verschiebung um den errechneten Wert mv1 wird nur auf das Leitungselement Element_{ref} angewandt. Es bleibt zu prüfen, ob der Abstand zu den Elementen auf der anderen Seite von Element_{ref}, wie in Abb. 6-9(a) dargestellt, verletzt wird. Die Berechnung des Verschiebungsvektors entspricht in diesem Fall weitestgehend der Gl. (6-42). Es ist jedoch der bereits errechnete Wert mv1 mit zu berücksichtigen. Damit gilt

$$\text{mv}_2 = \text{mv}_1 + \text{minDist}(\text{Element}_{ref}, \text{Element}_j) + X_r(\text{Element}_{ref}) - X_l(\text{Element}_j). \qquad (6\text{-}43)$$

Es werden infolge der östlichen Verschiebungsrichtung das Objekt Element_j und dessen östliche Nachbarn um mv_2 verschoben. Die Berechnung der Verschiebungsvektoren und die Ausführung der Verschiebungen erfolgt für die übrigen Richtungen entsprechend.

Es werden alle Leitungsabschnitte, die vom aktuell zu untersuchenden Pfadsegment Element_{ref} gekreuzt werden, in einer temporären Liste gespeichert. Die Liste wird basierend auf der Verdrahtungsrichtung des Referenzsegments Element_{ref} sortiert. Falls das Leitungssegment in horizontaler Richtung verläuft. werden die x-Koordinaten der Symmetriepunkte der geschnittenen Leitungen als Ordnungskriterium verwendet, andernfalls die y-Werte. Erfolgt die Verdrahtung durch das Referenzsegment in nördlicher bzw. östlicher Richtung, wird die Liste aufsteigend, andernfalls in absteigender Folge sortiert.

In Kap. 6.5.2 ist der Ablauf zur Erfassung aller Leitungselemente, die vom Element_{ref} gekreuzt werden, beschrieben worden. Mit Hilfe der resultierenden Liste, die alle diese Leitungsabschnitte enthält, können mögliche Kurzschlüsse erkannt und entfernt werden. Dazu ist zunächst ein Vergleich mit den für die Verdrahtung zur Verfügungen stehenden Layern durchzuführen. Wenn in der Liste ein Element existiert, dass die gleichen Layer wie Element_{ref} verwendet, sich aber auf einem anderen Knotenpotential befindet, wird zunächst geprüft, ob ein Maskenwechsel für das gesamte Leitungselement Element_{ref} möglich ist. Die Voraussetzung für diesen Vorgang ist, neben der

Verfügbarkeit eines entsprechenden Layers, dass es sich bei dem Leitungselement nicht um ein Ankersegment handelt. Ein Ankersegment beschreibt das erste bzw. letzte Leitungselement eines Pfades. Dieses ist direkt mit den geometrischen Objekten des Terminals verbunden. Ein Wechsel des Layers für ein solches Element würde die Verbindung auflösen. Die Erstellung eines Vias unter Einhaltung der Entwurfsregeln direkt am Terminal kann nicht immer garantiert werden.

Falls für das betroffene Leitungssegment eine Möglichkeit zu einem Layerwechsel gegeben ist, bleiben die Verbindungen zum vorhergehenden bzw. nachfolgenden Segment in diesem Pfad zu prüfen. Die Vorgehensweise dafür ist in beiden Fällen identisch. Zunächst wird geprüft, ob am gemeinsamen Wegpunkt eine Durchkontaktierung existiert. In diesem Fall wird das Via entfernt. Stimmen die Layer der beiden Leitungselemente überein, kann der Vorgang beendet werden. Sind die Masken der beiden Segmente hingegen verschieden, wird ein neues Via am gemeinsamen Symmetriepunkt erstellt. Im Anschluss an den Ebenenwechsel und der ggf. erforderlichen Konstruktion eines Vias, wird das Kantenbild der beiden Leitungselemente aktualisiert.

Die Erstellung einer Brücke wird dagegen genau dann durchgeführt, wenn der Kurzschluss nicht mittels der zuvor beschriebenen Vorgehensweise aufgelöst werden kann. Dazu sind im Folgenden die vorläufigen Koordinaten für das erste Via zu berechnen. Für die nachstehenden Beschreibungen wird bei den gekreuzten Leitungen von einer Menge vertikaler verlaufender Segments ausgegangen, die bzgl. der x-Koordinate aufsteigend sortiert sind. Die Ausführungen für die übrigen Fälle erfolgen auf analoge Weise.

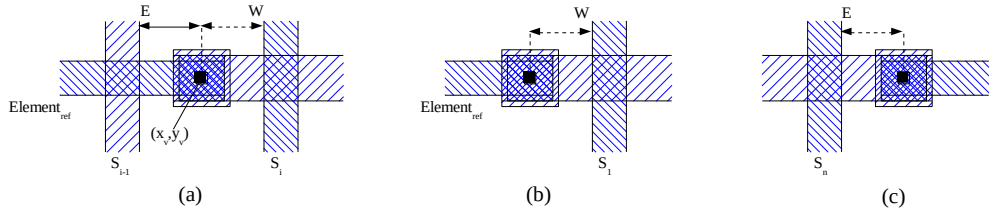


Abbildung 6-10: Erstellen eines Vias (a) zwischen zwei Leitungselementen, (b) vor dem ersten Leitungselement und (c) nach dem letzten Leitungslement

Für die Lage des neuen Vias ergeben sich in Bezug auf die vertikal verlaufenden Leitungselemente S_i zwei verschiedene Fälle (siehe Abbildung 6-10 (a) und (b)). Die y-Koordinate des vorläufigen Mittelpunkts für die Durchkontaktierung ergibt sich in beiden Fällen aus dem y-Wert des Wegpunktes des Leitungselements $\text{Element}_{\text{ref}}$, für das die Überbrückung zu erstellen ist. Die x-Koordinate berechnet sich in Abhängigkeit der Position an der das Via platziert werden soll. Falls sich das Via zwischen zwei Leitungselementen befindet, erfolgt die Berechnung des x-Wertes nach der folgenden Gleichung:

$$x_{\text{via}} = \big(\text{maxEast}(S_{i-1}) + \text{maxWest}(S_i)\big)/2. \tag{6-44}$$

Bis zu diesem Zeitpunkt stimmen die Layer von S_i und $\text{Element}_{\text{ref}}$ überein, während S_{i-1} und $\text{Element}_{\text{ref}}$ verschiedenen Ebenen zugeordnet sind. Falls S_i das erste Segment ist, berechnet sich der x-Wert des vorläufigen Mittelpunktes des Vias wie folgt:

$$x_{\text{via}} = \text{maxWest}(S_1) - W_{\text{Path,ref}}/2. \tag{6-45}$$

Der Wert $W_{\text{Path,ref}}$ beschreibt die Leitungsbreite für das Segment $\text{Element}_{\text{ref}}$. Im weiteren Verlauf wird das resultierende Via in die temporäre Liste der gequerten Leitungselemente entweder vor das

Segment S_i bzw. am Kopf der Liste eingetragen. Das Leitungselement $Element_{ref}$ wird in zwei Segmente gespalten. Der Mittelpunkt des neu erstellten Vias wird als neuer Wegpunkt dem Pfad von $Element_{ref}$ hinzugefügt. Die Einfügeposition ist durch die beiden Symmetriepunkt von $Element_{ref}$ gegeben. Das erste der beiden Segmente verläuft vom Startwegpunkt von $Element_{ref}$ zum Mittelpunkt des neu erstellten Via. Das zweite Segment $Element_{ref,neu}$ umfasst den Mittelpunkt des Via und den Zielwegpunkt von $Element_{ref}$. Im Folgenden wird das Leitungselement $Element_{ref,neu}$ betrachtet. Ausgehend vom gekreuzten Segment S_i erfolgt erneut eine Prüfung, ob ein Kurzschluss zwischen $Element_{ref,neu}$ und den auf S_i folgenden Leitungselementen vorliegt. Die bereits beschriebenen Schritte vom Untersuchen der Möglichkeit eines Maskenwechsels für das Segment $Element_{ref,neu}$ bis hin zur Erstellung einer Brücke werden im Fall eines Kurzschlusses solange wiederholt, bis das Ende der Liste der orthogonal verlaufenden Leitungen erreicht ist. Abschließend bleibt für das resultierende Leitungselement $Element_{ref}$ bzw. dem entsprechenden Element $Element_{ref,neu}$ zu prüfen, ob eine Verbindung zum nachfolgenden Segment bzw. Terminal existiert. Wenn $Element_{ref}$ bzw. $Element_{ref,neu}$ das letzte Leitungselement des Pfades ist und die Layer des Leitungselements und des Zielterminals verschieden sind, ist noch ein weiteres Via zu erstellen. Eine der beiden vorläufigen Koordinaten ist stets durch die Wegpunkte des Leitungselements gegeben. Bei einem vertikalen Verlauf des Leitungsabschnitts wird für den Mittelpunkt des Vias der x-Wert der Wegpunkte übernommen, andernfalls die y-Koordinate. Die fehlende Koordinate berechnet sich beispielsweise für den in Abb. 6-10 (c) dargestellten Fall nach der folgenden Gleichung:

$$x_{\text{via}} = \text{maxEast}(S_n) + W_{\text{Path,ref}}/2. \tag{6-46}$$

wobei $W_{Path,ref}$ die Leitungsbreite des Segments $Element_{ref}$ darstellt. Ausgehend vom Mittelpunkt des Vias, welcher dem Pfad von $Element_{ref}$ als weiterer Wegpunkt hinzugefügt wird, erfolgt eine Teilung des Leitungselement $Element_{ref}$ in zwei Segmente.

Besitzt $Element_{ref}$ hingegen ein nachfolgendes Segment, bleibt zu prüfen, ob die Layer der beiden Leitungselemente übereinstimmen. Ein bereits vorhandenes Via wird unabhängig vom Ergebnis der Fallunterscheidung entfernt. Eine neue Durchkontaktierung wird am gemeinsamen Wegpunkt erstellt, wenn die beiden Leitungselemente verschiedene Layer aufweisen.

Nach Abschluss der Iteration über alle Leitungselemente, die vom zu prüfenden Segment gekreuzt werden, ist genau dann eine Neukompaktierung der Leitungsabschnitte notwendig, wenn zusätzliche Vias eingefügt werden mussten. Die Reihenfolge der zu kompaktierenden Elemente ist in diesem Fall durch die Sortierung der Liste, in der die Leitungselemente und Vias gespeichert sind, vorgegeben. Es sind lediglich für das erste und das letzte Element dieser Liste die benachbarten Elemente mit der in Kap 6.5.1 beschriebenen Vorgehensweise zu bestimmen. Anschließend wird aus den ermittelten Objekten die Verschiebungsrichtung festgelegt.

6.6 Objektorientierte Implementierung des Verdrahters

Im Folgenden werden die Grundzüge der Programmstruktur umrissen. Die Klassen für das Verdrahtungswerkzeug wurden in der Programmiersprache C++ unter Verwendung der Entwicklungsumgebung Sun Studio auf einer SunBlade 2500 implementiert. Außerdem wurden verschiedene Klassen, Algorithmen und Funktionen der MOGLAN-Bibliothek [Wolf96, Wolf98b, Wolf99a] verwendet. In Abbildung 6-11 ist eine Übersicht der Hierarchie der wichtigsten Klassendefinitionen für die Realisierung des Verdrahters gegeben. Unter Ausnutzung spezieller Möglichkeiten der objektorientierten Programmierung, wie Vererbung und Laufzeit-Polymorphie, lassen sich gemeinsame Attribute und Methode konzeptionell verwandter Objekttypen einheitlich

modellieren. Unterschiedlichkeiten können durch entsprechende Spezialisierungen in den von den Basisklassen abgeleiteten Unterklassen berücksichtigt werden.

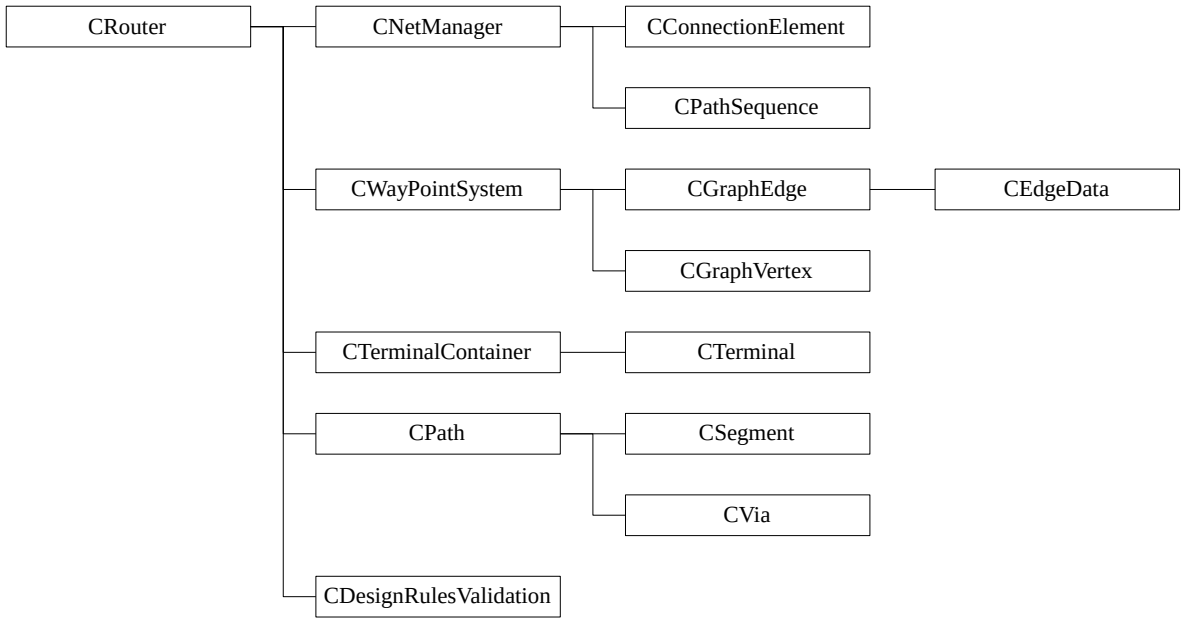


Abbildung 6-11: Klassenhierarchie des Verdrahtungswerkzeugs

Die Klassen lassen sich in fünf Gruppen aufteilen: Es existieren drei Klassen zur Verwaltung der Netzlisten und der Verdrahtungswege in Form von Punktlisten (CNetManager, CConnectionElement, CPathSequence), vier Klassen für die Umsetzung des Wegenetz-Graphen (CWayPointSystem, CGraphVertex, CGraphEdge, CEdgeData), zwei Klassen für Operationen auf den Anschlüssen der Zellen (CTerminalContainer, CTerminal), drei Klassen für die physikalische Repräsentation eines Pfades (CPath, CSegment, CVia) und eine Klasse für die Ausführung der Entwurfsregelprüfung. Alle genannten Klassendefinitionen laufen in der Klasse CRouter zusammen.

Die Klasse CRouter stellt eine Funktionsschnittstelle dar, um zwei kompaktierte Zellblöcke zu verdrahten. In ihr werden die Geometrien der Zellen, der Wegenetz-Graph und die zu verbindenden Anschlüsse abgelegt. Nach der Übergabe der beiden zu verbindenden Module werden alle Geometrien, bestehend aus den in den Modulen enthaltenen Zellen und Pfade, sowie die gemeinsamen Netze und die zugehörigen Terminals erfasst. Die Referenzen auf die entsprechenden Objekte werden jeweils in einer linearen Liste gespeichert. Es wird der Wegenetz-Graph erstellt und die Sortierung der zu verdrahtenden Netze gemäß den Vorgaben für die Routensuche veranlasst. Nach Ausführung der Verbindungssuche, der Layouterstellung der Pfade und deren Verifikation, werden die resultierenden Pfade durch die Klasse CRouter für die weitere Verarbeitung bereitgestellt.

Die Klasse CNetManager ist eine Ableitung einer doppelt verketteten Liste, die die Elemente vom Typ CConnectionElement enthält. In der Klasse werden alle zu verdrahtenden Netze abgelegt und entsprechend ihrer Eigenschaften sortiert. In einer Iteration über alle Anschlusskombinationen jedes Netzes wird für jede Anschlusspaarung ein Objekt vom Typ CConnectionElement erstellt und in der Liste gespeichert. Während der Aufbau eines Wegenetz-Graphen in der Klasse CRouter erfolgt, wird in der Klasse CNetManager dieser Graph zur Auswahl einer Route ausgewertet. Es wird für jede

Instanz von CConnectionElement die Route sowie die Verbindungskosten berechnet und schließlich für das entsprechende Netz die günstigste Verbindung ausgewählt.

Die Klasse CConnectionElement ist von einer Punktliste abgeleitet und stellt damit Methoden, wie das Entfernen kollinearer Punkte, bereit. Die Klasse ist ein Datencontainer, der neben den Wegpunkten einer Route, die beiden Anschlüsse und die Kosten für die Verbindung enthält.

Der Wegenetz-Graph wird durch Klasse CWayPointSystem bereitgestellt. Es wird neben Methoden zur Verwaltung der Knoten und Kanten, dargestellt durch Instanzen von CGraphVertex und CGraphEdge, die Routensuche mittels Floyd-Warshall-Algorithmus in der Klasse gekapselt. Ferner werden Referenzen der Zellgeometrien und die Menge der kürzesten Verbindungen zwischen den Knoten gespeichert. Die Gewichtung und der Typ einer Kante des Graphen werden durch den Datencontainer CEdgeData repräsentiert.

Die Klasse CTerminalContainer basiert auf einer doppelt verketteten Liste. Es stehen zusätzlich geometrische Operationen zur Translation bzw. Transformation der in der Liste ablegten Elemente zur Verfügung. Zu jedem Netz existiert eine Instanz CTerminalContainer. Die Elemente der Liste sind die Anschlüsse des jeweiligen Netzes, die durch CTerminal dargestellt werden. Die Klasse CTerminal basiert auf der Klasse CBasicEdge. Damit erhält das Terminal alle geometrischen Operationen und Datenfelder einer Kante. Es enthält weiterhin einen Verweis auf einen Pfad, sowie Methoden, um die Ausrichtung zu einem anderen Terminal zu bestimmen.

Die Methoden und Eigenschaften einer Geometrie erben die Klassen CPath, CSegment und CVia. Darin sind u.a. die Funktionen zur Verwaltung der Kantenbilder und zur Durchführung der Kompaktierung enthalten. Auf diese Weise kann jede Instanz der drei Klassen als eine Zelle betrachtet werden. Ferner verfügen die Klassen CSegment, die ein einzelnes Leitungselementes repräsentiert und CVia über Methoden zur Konstruktion der Layoutdaten. In der Klasse CPath stehen Funktionen zum Verwalten und Bearbeiten der Instanzen von CSegment und CVia bereit. Ebenso sind die Operationen für die Translation und Transformation der Vias und Leitungselemente, sowie Methoden zum Stauchen bzw. Strecken der Segmente unter Berücksichtigung der vorhergehenden und nachfolgenden Elemente implementiert.

Die Durchführung der Entwurfsregelprüfung ist Bestandteil der Klasse CDesignRulesValidation. Es sind zunächst die Referenzen auf alle existierenden Geometrien der Zellen und Pfade in den Klassen abzulegen. Anschließend kann der zu prüfende Pfad übergeben und untersucht werden.

7 Layoutbeispiele

In diesem Kapitel werden einige Layoutbeispiele vorgestellt, die mit Hilfe des entwickelten Platzierungs- und Verdrahtungswerkzeugs erstellt wurden. Zur Bewertung der Algorithmen, die im Rahmen dieser Arbeit entstanden sind, werden vier Beispielschaltungen verwendet. Infolge der Bedeutung von Operationsverstärkern in analogen Schaltungen werden drei verschiedene Verstärker und ein einfacher CMOS-Komparator ausgewählt. Neben der Aufteilung und der sich daraus ergebenden Modulwahl für jede Schaltung, werden die für die Platzierung relevanten Netze und die resultierenden Layouts präsentiert. Die Bulkanschlüsse der Transistoren sind in den Schaltbildern aus Gründen der Übersichtlichkeit nicht explizit eingezeichnet. Das Bulk der p-Kanal Transistoren ist stets mit V_{DD} verbunden und entsprechend für die n-Kanal Transistoren mit V_{SS}.

7.1 Zweistufiger Operationsverstärker mit Eins-Verstärker

Das Schaltbild des zweistufigen CMOS-Operationsverstärkers sowie die Aufteilung der Module sind in Abbildung 7-1 dargestellt. Die Transistoren M_1 und M_2 bilden die p-Kanal-Differenzstufe. Die aktive Last der Differenzstufe wird mit Hilfe der n-Kanal Transistoren M_3 und M_4 realisiert. Die Stromversorgung des Operationsverstärkers erfolgt durch eine Stromspiegelschaltung, bestehend aus den Transistoren M_6 bis M_8 und der Stromquelle I_{Bias}. Die Ausgangsstufe des Verstärkers bilden die Stromquelle M_6 und der n-Kanal Transistor M_5. Die Frequenzkompensation wird durch den p-Kanal Sourcefolger M_9, der zugehörigen p-Kanal Stromquelle M_{10} und der Kompensationskapazität C_C erzielt. Eine detaillierte Beschreibung des Verstärkers ist den Kapitel 3.1.3 zu entnehmen.

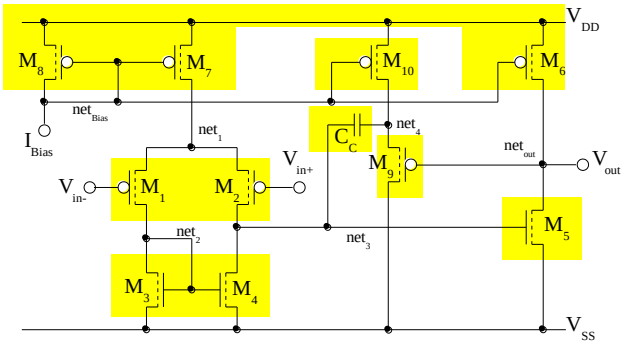


Abbildung 7-1: Schaltbild und Modulaufteilung des zweistufiger CMOS-Operationsverstärkers mit Eins-Verstärker

Die Transistoren M_1 und M_2 formen das Modul der Differenzstufe. Die differentielle Last (M_3, M_4) und die Transistoren M_6 bis M_8 der Stromversorgung werden jeweils durch ein Stromspiegelmodul zusammengefasst. Für die Kompensationskapazität C_C, den Sourcefolger M_9, den p-Kanal Transistor M_{10} und dem Ausgangstreiber M_5 werden je ein Modul erstellt.

Der nachfolgenden Tabelle 7-1 können die Informationen zu den wichtigen Netzen der Schaltung entnommen werden. Neben dem Netznamen ist jeweils der Typ, die Priorität und die aus einer DC Simulation resultierende Stromdichte angegeben.

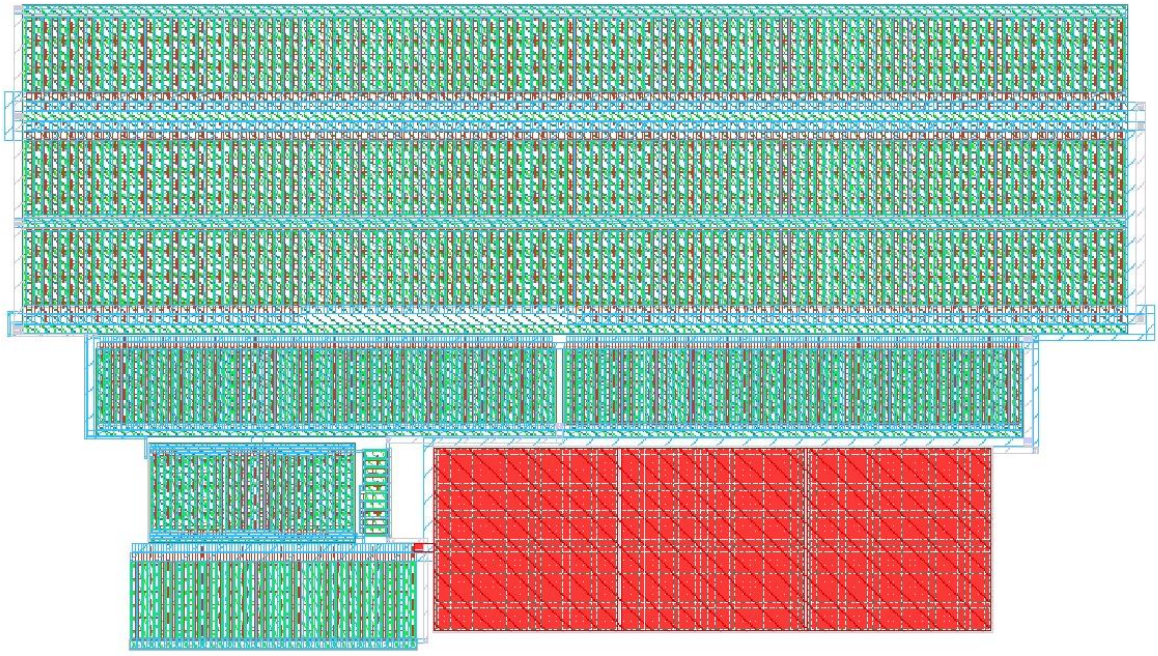


Abbildung 7-2: Platzierungs- und Verdrahtungsergebnis für den Operationsverstärkers mit Einsverstärker

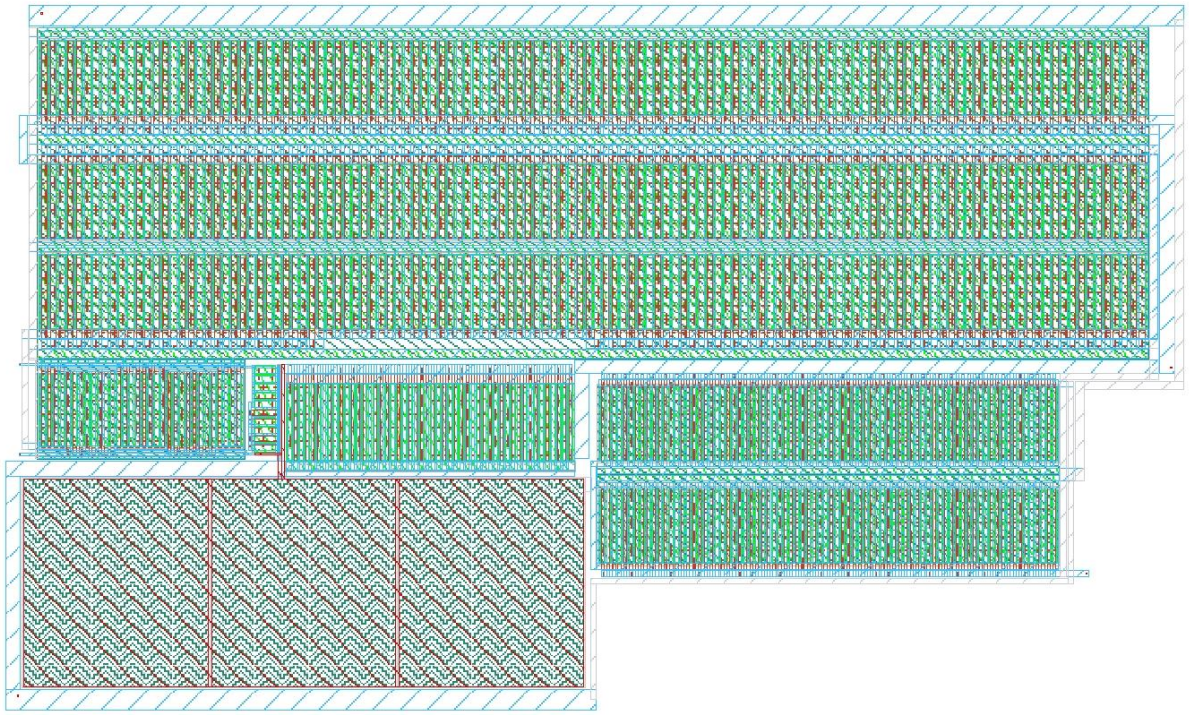


Abbildung 7-3: Handlayout des Operationsverstärkers mit Einsverstärker

Nach der Anwendung der Platzierungs- und Verdrahtungsverfahren ergibt sich das in Abbildung 7-2 gezeigte Layout für den Operationsverstärker mit Einsverstärker. Zum Vergleich ist in Abbildung 7-3 das per Hand erzeugte Layout des Verstärkers nochmals dargestellt. Der Tabelle 7-2 ist eine Gegenüberstellung der relevanten Kenndaten des Verstärkers zu entnehmen. Es werden dabei Leerlaufspannung, Phasenreserve, Bandbreite sowie die Fläche zwischen dem gemessenen, aus dem

Handlayout hervorgehenden Operationsverstärkers und den simulierten Werten des generierten Layouts verglichen.

Tabelle 7-1: Netzliste des Verstärkers aus Abbildung 7-1

Netzname	Typ	Priorität	Stromdichte
net_{Bias}	Supply	20	450 µA
net_1	Supply	30	450 µA
net_2	Signal	50	450 µA
net_3	Signal	40	450 µA
net_4	Signal	30	2 mA
net_{out}	Signal	40	3 mA

Tabelle 7-2: Gegenüberstellung des Operationsverstärkers mit Einsverstärker

Parameter	Messergebnisse des OpAmp aus Handlayout	Simulationsergebnisse des halbautomatisch erzeugten OpAmps
Versorgungsspannung	±1.5 V	±1.5 V
Leerlaufspannungsverstärkung	91 dB	92 dB
Phasenreserve	64°	60°
Bandbreite	70 MHz	80 MHz
Fläche (Länge * Weite)	400 * 240 µm²	400 * 245 µm²

7.2 Operationsverstärker mit gefalteter Kaskode

In Abbildung 7-4 ist das Schaltbild des Operationsverstärkers mit einer gefalteten Kaskode gezeigt. Es ist der Abbildung ebenso die Modulaufteilung zu entnehmen. Der Verstärker setzt sich aus zwei komplementären Differenzstufen (M_1, M_2 und M_3, M_4) sowie den beiden jeweils zugehörigen komplementären aktiven Lasten (M_5, M_6 und M_9, M_{10}) zusammen. Die Stromspiegel der aktiven Lasten sind mit den Kaskodentransistoren M_7 und M_8 verbunden. Die Stromversorgung der Schaltung erfolgt mittels eines p-Kanal Stromspiegels, bestehend aus den p-Kanal Transistoren M_{13}, M_{14}, M_{15} und M_{18}, sowie eines n-Kanal Stromspiegels, realisiert durch die Transistoren M_{11}, M_{12} und M_{17}, sowie einer Stromquelle I_{Bias}. Zur Kontrolle des Stroms durch den Ausgangszweig werden die MOS Dioden M_{16} und M_{19} verwendet. Eine weitergehende Beschreibung des Operationsverstärkers ist im Kapitel 3.4 zu finden.

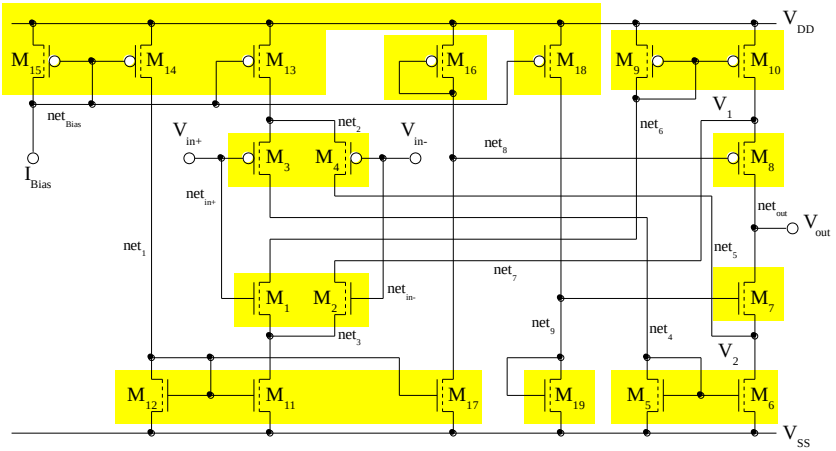


Abbildung 7-4: Schaltbild und Modulaufteilung des Operationsverstärkers mit gefalteter Kaskode

Die Transistoren der Differenzpärchen M_1 und M_2 bzw. M_3 und M_4 werden jeweils in einem Modul für eine Differenzstufe zusammengefasst. Die differentiellen Lasten M_5, M_6 sowie M_9, M_{10}, die p-Kanal Transistoren M_{13}, M_{14}, M_{15}, M_{18} und die n-Kanal Transistoren M_{11}, M_{12} und M_{17} werden je in einem Stromspiegelmodul umgesetzt. Für die MOS Dioden M_{16} und M_{19} sowie für die Kaskodentransistoren M_7 und M_8 wird in jedem Fall ein eigenes Modul verwendet.

In Tabelle 7-3 sind die Namen, die Priorität, die jeweiligen Netzcharakterisierungen sowie die simulierte Stromdichte der für die Verdrahtung und Platzierung zu verwendenden Netze dargestellt. Die Netznamen stimmen mit denen in der Abbildung 7-4 gezeigten Namen überein.

Tabelle 7-3: Netzliste des Verstärkers aus Abbildung 7-4

Netzname	Typ	Priorität	Stromdichte
net_{Bias}	Supply	20	400 µA
net_{out}	Signal	50	200 µA
net_{in+}	Input	40	-
net_{in-}	Input	40	-
net_1	Supply	30	400 µA
net_2	Supply	30	400 µA
net_3	Supply	30	400 µA
net_4	Signal	60	400 µA
net_5	Signal	50	400 µA
net_6	Signal	60	400 µA
net_7	Signal	50	400 µA
net_8	Supply	30	200 µA
net_9	Supply	30	200 µA

Das Ergebnis der Anwendung der Platzierungs- und Verdrahtungsalgorithmen ist in der Abbildung 7-5 präsentiert. Das per Hand erzeugte Layout des Verstärkers ist zum Vergleich in Abbildung 7-6 dargestellt. Eine Gegenüberstellung der Meßergebnisse des via Handlayout erstellten Verstärkers mit den Simulationsergebnissen, die aus einer Postlayout-Simulation des halbautomatisch erzeugten Layouts hervorgehen, ist der Tabelle 7-4 zu entnehmen.

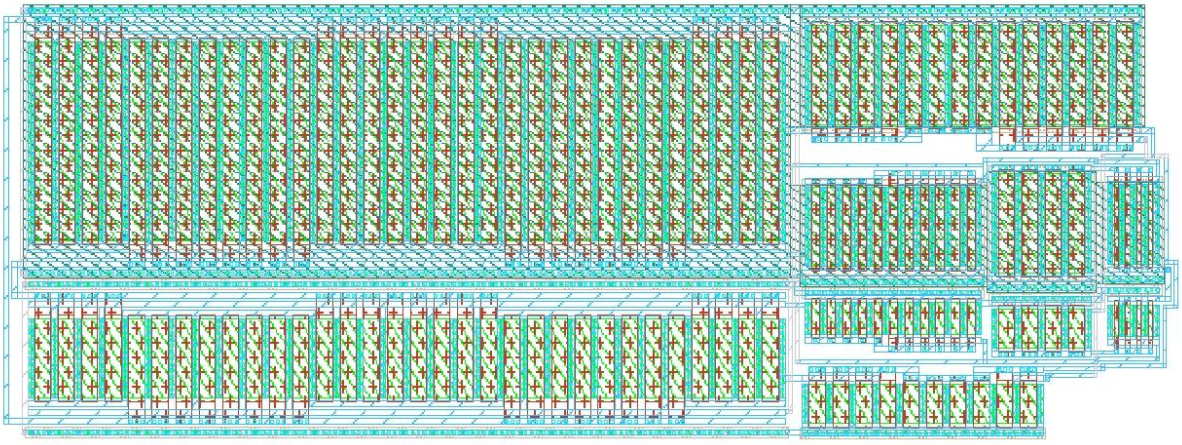


Abbildung 7-5: Platzierungs- und Verdrahtungsergebnis für den Operationsverstärkers mit gefalteter Kaskode

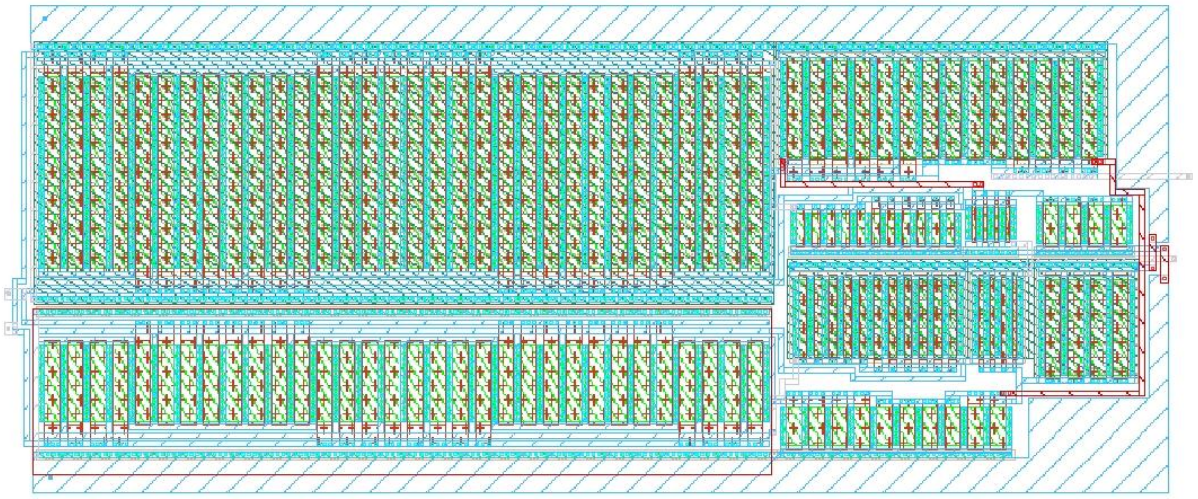


Abbildung 7-6: Handlayout des Operationsverstärkers mit gefalteter Kaskode

Tabelle 7-4: Gegenüberstellung der Meß- und Simulationsergebnisse des Operationsverstärkers mit gefalteter Kaskode

Parameter	**Messergebnisse des OpAmp aus Handlayout**	**Simulationsergebnisse des halbautomatisch erzeugten OpAmps**
Versorgungsspannung	±1.5 V	±1.5 V
Leerlaufspannungsverstärkung	85 dB	86 dB
Phasenreserve	55°	53°
Bandbreite	53 MHz	50 MHz
Fläche (Länge * Weite)	150 * 60 µm²	150 * 60 µm²

7.3 Zweistufiger Kaskoden-Operationsverstärker

Der Abbildung 7-7 ist das Schaltbild des zweistufigen Kaskoden-Operationsverstärkers zu entnehmen. Die Eingangsstufe bilden die beiden p-Kanal Differenzstufen der Transistoren M_1, M_2 und M_3, M_4. Die aktive Last wird durch die n-Kanal Transistoren M_7 und M_8 realisiert. Die Stromversorgung erfolgt mit Hilfe der Stromquelle I_{Bias} und der p-Kanal Transistoren M_{10} bis M_{13}. Die Ausgangsstufe wird durch den n-Kanal Transistor M_9 und dessen Stromquelle M_{10} geformt. Die Verwendung der Kaskodentransistoren M_5 und M_6 ermöglichen eine Verbindung des Source des Transistors M_6 mit dem Ausgang unter Verwendung der Kompensationskapazität C_1. Das zur Frequenzkompensation genutzte Prinzip der verschachtelten Miller-Kompensation erlaubt eine Verbindung des Gates des Transistors M_9 mit dem Ausgang durch die Kompensationskapazität C_2. Eine ausführliche Erläuterung des Kaskoden-Operationsverstärkers ist in Kapitel 3.2 dargestellt.

Die Transistoren der Differenzstufen M_1, M_2 und M_3, M_4 werden in je einem Modul zusammengefasst. Für die p-Kanal Transistoren M_{10} bis M_{13} sowie die n-Kanal Transistoren M_7 und M_8 wird jeweils ein Stromspiegelmodul verwendet. Die Kaskodenstufe, bestehend aus den Transistoren M_5, M_6 und M_{14}, die Kompensationskapazitäten C_1 und C_2 sowie die Ausgangsstufe M_9 werden entsprechend in jeweiligen eigenen Modulen umgesetzt.

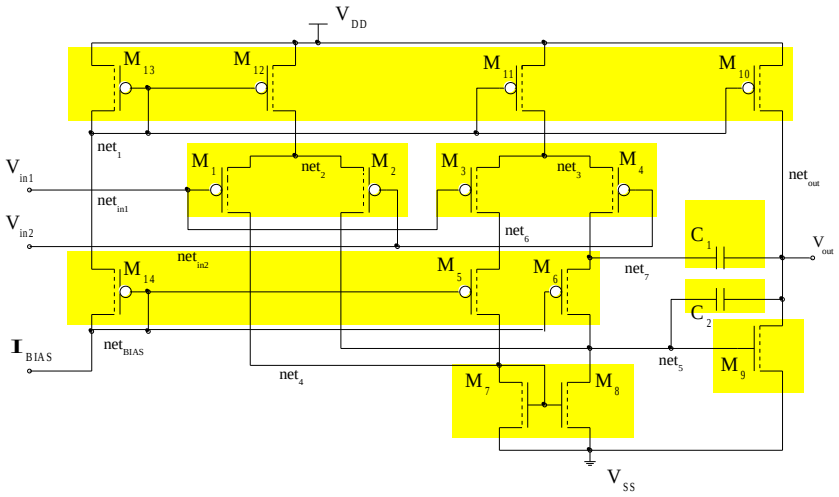


Abbildung 7-7: Schaltbild und Modulaufteilung des zweistufigen Kaskoden-Operationsverstärker

Aus der Tabelle 7-5 sind die Namen der Netze, deren jeweiliger Typ, die Priorität und die zu erwartende Stromdichte ersichtlich.

Tabelle 7-5: Netzliste des Verstärkers aus Abbildung 7-7

Netzname	Typ	Priorität	Stromdichte
net_{Bias}	Supply	20	200 µA
net_{in1}	Input	80	-
net_{in2}	Input	80	-
net_1	Supply	20	200 µA
net_2	Supply	30	200 µA
net_3	Supply	30	200 µA
net_4	Signal	60	200 µA
net_5	Signal	50	200 µA
net_6	Signal	60	200 µA
net_7	Signal	40	200 µA
net_{out}	Signal	40	2 mA

Es ergibt sich das in Abbildung 7-8 gezeigte Platzierungs- und Verdrahtungsergebnis für den zweistufigen Kaskoden-Operationsverstärker. Der Abbildung 7-9 ist das Handlayout des Verstärkers zum Vergleich zu entnehmen. In der

Tabelle 7-6 erfolgt eine Gegenüberstellung der Meßergebnisse des aus dem Handlayout hervorgehenden Verstärkers mit den Simulationswerten, die aus der Postlayout-Simulation des halbautomatisch erzeugten Layouts resultieren.

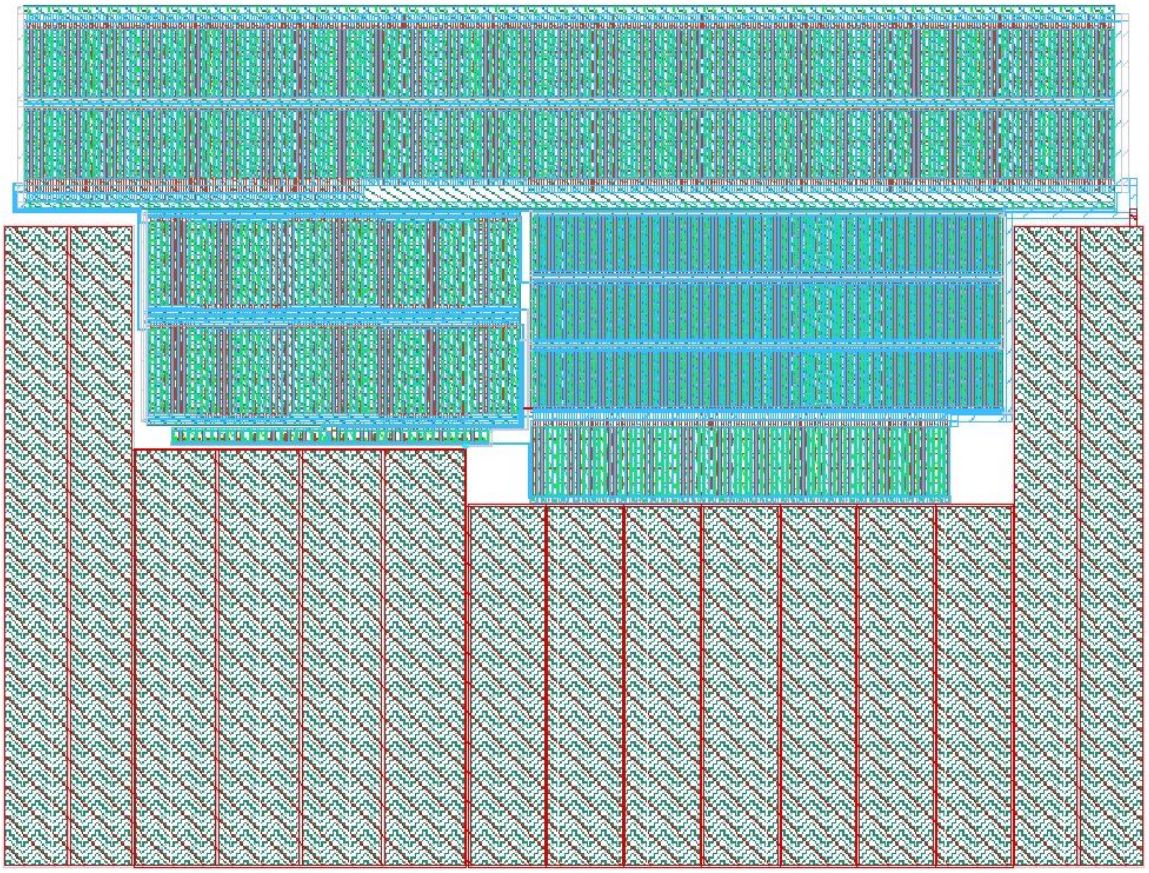


Abbildung 7-8: Platzierungs- und Verdrahtungsergebnis für den zweistufigen Kaskoden-Operationsverstärker

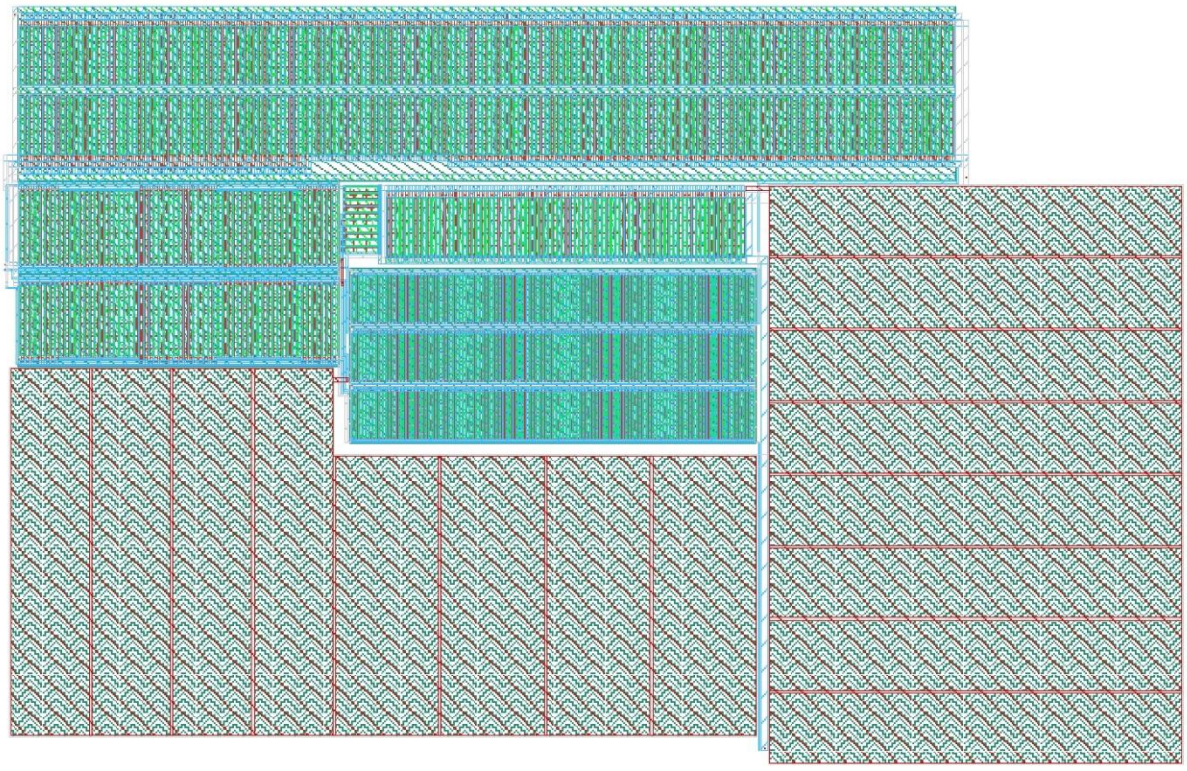


Abbildung 7-9: Handlayout des zweistufigen Kaskoden-Operationsverstärkers

Tabelle 7-6: Gegenüberstellung der Meß- und Simulationsergebnisse des zweistufigen Kaskoden-Operationsverstärkers

Parameter	Messergebnisse des OpAmp aus Handlayout	Simulationsergebnisse des halbautomatisch erzeugten OpAmps
Versorgungsspannung	±1.5 V	±1.5 V
Leerlaufspannungsverstärkung	97 dB	101 dB
Phasenreserve	62°	59°
Bandbreite	14.3 MHz	15 MHz
Fläche (Länge * Weite)	440 * 390 µm²	400 * 430 µm²

7.4 Einfacher CMOS-Komparator

Das Schaltbild eines einfachen CMOS-Komparators ist in der Abbildung 7-10 dargestellt. Die Eingangsstufe ist in Form einer p-Kanal Differenzstufe (M_1, M_2) und einer zugehörigen aktiven Last (M_3, M_4) realisiert. Dem folgt eine verstärkende Stufe, bestehend aus dem n-Kanal Transistor M_5 und dessen Stromquelle M_6. Der Ausgang wird durch einen CMOS Inverter umgesetzt (M_9, M_{10}). Die Stromversorgung ist durch die Stromquelle I_{Bias} und die p-Kanal Transistoren M_6 bis M_8 gewährleistet. Weiterführende Erläuterungen sind dem Kapitel 3.6 zu entnehmen. Bedingt durch das fehlende Kompensationsnetz ist die Konstruktion von Komparatoren im Vergleich zu den Operationsverstärkern einfacher.

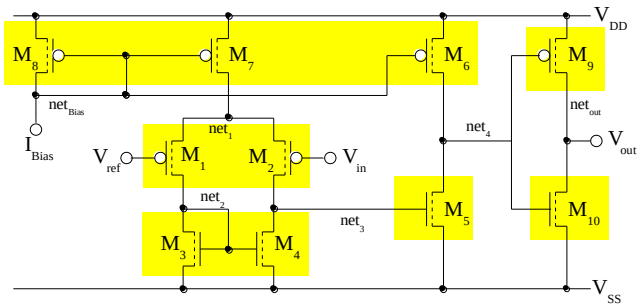


Abbildung 7-10: Schaltbild und Modulaufteilung des einfachen CMOS-Komparator

Die p-Kanal Transistoren M_6 bis M_8 sowie die n-Kanal Transistoren M_3 und M_4 sind jeweils in einem Stromspiegelmodul zusammengefasst. Ebenfalls bilden die Differenzstufentransistoren M_1 und M_2, der Treiber M_5 und die beiden Transistoren M_9 und M_{10} des Inverters jeweils ein eigenes Modul.

In der nachfolgenden Tabelle 7-7 sind die Daten des jeweiligen Netzes des Komparators dargestellt. Es sind die Leitungstypen, Prioritäten und Stromdichten der Netze aufgeführt.

Tabelle 7-7: Netzliste des Komparators aus Abbildung 7-10

Netzname	Typ	Priorität	Stromdichte
net_{Bias}	Supply	20	100 µA
net_1	Supply	30	100 µA
net_2	Signal	50	100 µA
net_3	Signal	50	100 µA
net_4	Signal	40	100 µA
net_{out}	Signal	50	20 µA

Das Ergebnis der Anwendung der Platzierungs- und Verdrahtungsalgorithmen auf den einfachen CMOS-Komparator ist in Abbildung 7-11 gezeigt. Zum besseren Vergleich ist in Abbildung 7-12 das Handlayout des Komparators dargestellt.

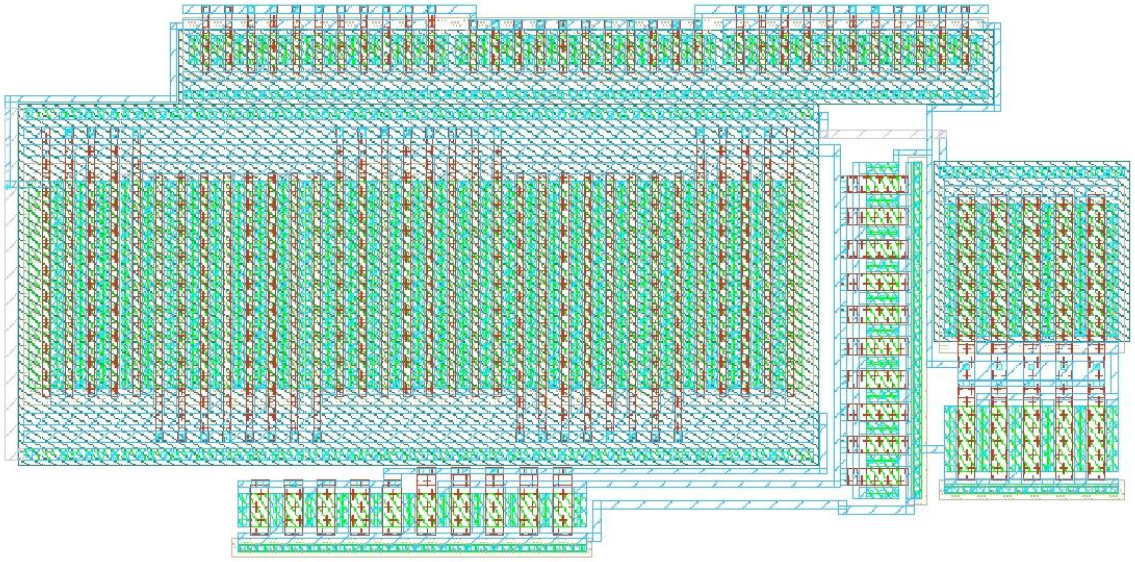


Abbildung 7-11: Platzierungs- und Verdrahtungsergebnis für den einfachen CMOS-Komparator

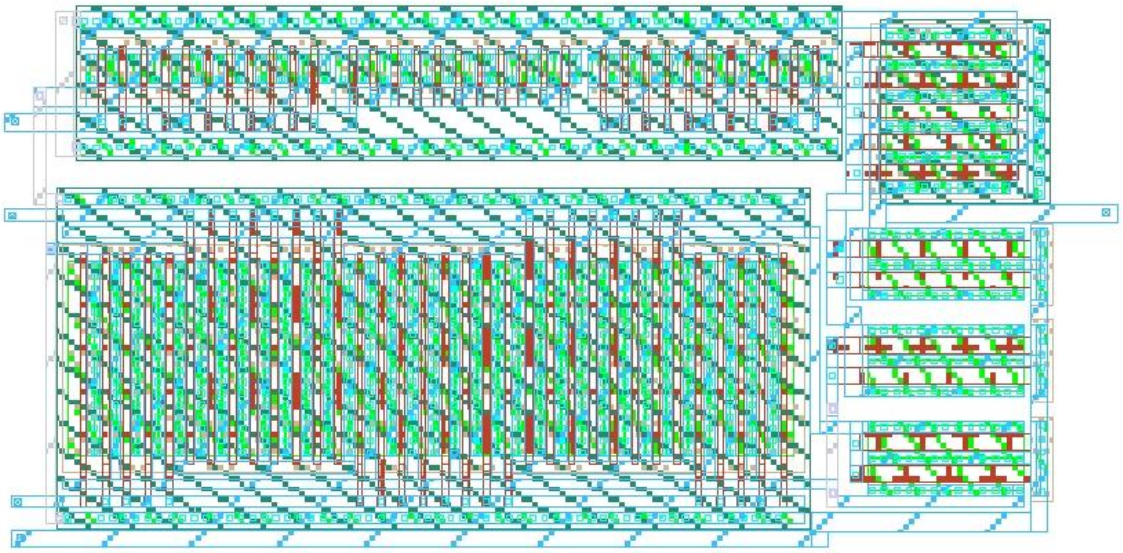


Abbildung 7-12: Handlayout des einfachen CMOS-Komparators

8 Zusammenfassung

In den vorangegangenen Kapiteln wurde ein neues Konzept zur Platzierung und Verdrahtung analoger, integrierter Schaltungen vorgestellt und die dazu entwickelten Algorithmen erläutert. Ebenso wurde die Implementierung der Verfahren beschrieben. Das vorliegende, abschließende Kapitel befasst sich mit der Zusammenfassung der Ergebnisse der Arbeit. Mit einem Ausblick auf zukünftige Arbeiten schließt das Kapitel.

8.1 Ergebnisse

Es sind Modulgeneratoren zur Erstellung von integrierten Präzisionswiderständen und -kapazitäten sowie symmetrischen, spiralen Induktivitäten vorgestellt worden. Neben der Genauigkeit sind die physikalischen Effekte kurz untersucht worden. Ebenso wurden die notwendigen Änderungen an der MOGLAN-Bibliothek durchgeführt, um die Elemente erzeugen zu können. Für die Induktivität wurde ein geeignetes Modell zur Simulation ausgewählt und beschrieben.

Im Rahmen der Arbeit sind diverse analoge Schaltungen entstanden. Anhand von Schaltungsbeispielen, inklusive deren Layouts und Postlayout-Simulationen, sind die verschiedenen Kompensationsarten zweistufiger Operationsverstärker vorgestellt worden. Es ist weiter ein neuartiger zweistufiger Kaskoden-Operationsverstärker entstanden, dessen Aufbau und Funktionsweise mittels eines geeigneten Kleinsignalmodells dargelegt wurde. Die Schaltung wurde in einer 0.25µm CMOS Technologie der Firma IHP GmbH gefertigt. Die Meßergebnisse sind präsentiert worden. Neben den genannten Verstärkern sind noch einstufige CMOS Operationsverstärker erstellt worden. Es handelt sich dabei um einen einfachen Operationsverstärker mit einer Highswing-Kaskode sowie einen Operationsverstärker mit einer gefalteten Kaskode. Der letztgenannte Verstärker wurde ebenfalls in der 0.25µm CMOS Technologie produziert und gemessen.

Die bekannten Lösungsansätze und -verfahren zur Platzierung und Verdrahtung integrierter Schaltungen und deren Anwendung auf analoge Schaltungen sind vorgestellt worden. Das in dieser Arbeit entwickelte Verfahren ermöglicht eine annähernd simultane Durchführung der Platzierung und der Verdrahtung. Zentrales Element des Algorithmus ist ein hierarchisches, sukzessives Kompaktieren der Zellen. Aus den zu platzierenden Modulen und einer zugehörigen Netzliste wird ein Verbindungsgraph erzeugt. Die Knoten des Graphen werden solange vereinigt, bis ein einziger Knoten übrig bleibt. Während des Verschmelzungsvorgangs entsteht ein binärer Platzierungsbaum, der den Zeitpunkt und die Reihenfolge der Kompaktierungen der Module festlegt. Die Auswahl der Topologien, Kompaktierungsrichtungen und ggf. erforderliche Transformationen erfolgt mit Hilfe eines Entscheidungsbaums. Eine Platzierungslösung wird mittels eines Schnittbaums dargestellt. Die Grundlagen zu den Entscheidungsbäumen und Schnittbäumen sind ebenso erläutert worden, wie die notwendigen Algorithmen zur Konstruktion und Auswertung der Datenstrukturen.

Nach einem erfolgreich ausgeführten Kompaktierungsschritt wird die Verdrahtung erzeugt. Die Darstellung der möglichen Verbindungswege wird durch einen Wegenetz-Graphen realisiert. Dazu ist die orthogonale Polygonhülle der Module auszuwerten. Zur Ausführung der Verdrahtung wird zunächst der Einsatz einfacher Leitungselemente geprüft. Andernfalls wird das Verfahren nach Floyd und Warshall auf den Wegenetz-Graphen angewandt, um eine kostengünstige Route zu ermitteln. Aufbauend auf den Wegpunkten einer Verbindung wird das Layout für einen Pfad automatisch erzeugt. Die Einhaltung der notwendigen Entwurfsregeln und eine ggf. erforderliche Korrektur

erfolgen selbständig. Die notwendigen Schritte und Verfahren für die Verdrahtung sind ausführlich dargelegt worden. Ebenfalls wurden weitere Möglichkeiten zur Wegesuche in einem Graphen beschrieben.

Anhand von Beispielen ist die Leistungsfähigkeit des implementierten Platzierungs- und Verdrahtungswerkzeugs gezeigt worden.

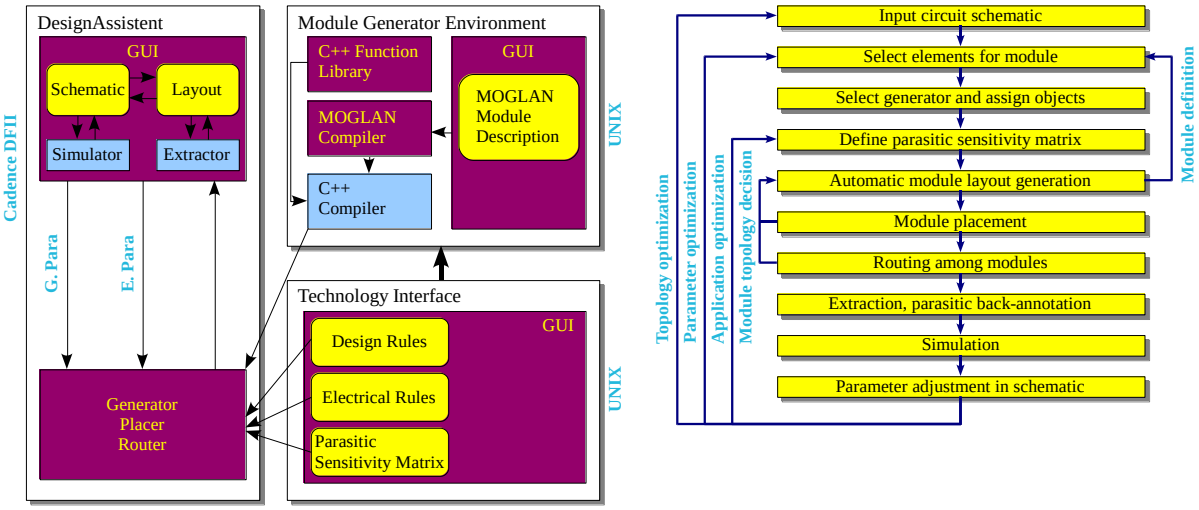
8.2 Ausblick

Während der Schaltungssynthese auf einer höheren Ebene werden die Effekte durch parasitäre Elemente im Layout in der Regel lediglich abgeschätzt. Die im Rahmen dieser Arbeit entwickelten und verwendeten Werkzeuge für die automatische Layouterzeugung erlauben eine bedeutend bessere Approximation der parasitären Effekte. Um die Geschwindigkeit der automatischen Synthese zu erhöhen, ist die Erweiterung der Verfahren auf eine parallele Abarbeitung ein erfolgsversprechender Aufgabenpunkt. Insbesondere der Algorithmus nach Floyd und Warshall zur Wegsuche in einem Wegenetz-Graphen lässt sich mit einem vertretbaren Aufwand auf ein paralleles System abbilden.

Der Schnittbaum zur Darstellung einer Platzierungslösung ist eine effektive Datenstruktur, die Raum für Optimierungen zulässt. Die in dieser Arbeit bereits entwickelten Methoden zur Rekonstruktion des Schnittbaums, ermöglichen eine dichtere Anordnung der Module. Eine Erweiterung der Funktionen in Bezug auf die Einhaltung von Symmetrie-Bedingungen oder weiteren nutzerspezifischen Angaben kann zu einer Verbesserung der Ergebnisse beitragen. Konstruktionsbedingt kann ein Entscheidungsbaum bei einer großen Datenmenge, die als Grundlage dient, einen sehr hohen Speicherbedarf aufweisen. Infolge der geringeren Anzahl an Modulen bei integrierten analogen Schaltungen ist die in dieser Arbeit verwendete manuelle Begrenzung der Suchtiefe für die Erstellung eines Entscheidungsbaums hinreichend. Eine Ausweitung des Verfahrens auf digitale Schaltungen und der damit meist verbundenen großen Menge an Modulen, kann den Bedarf an komplexeren Methoden zur Begrenzung der Größe des Entscheidungsbaums notwendig machen. In der Regel wird dafür auf heuristische Methoden zurückgegriffen, da es sich um ein NP-vollständiges Problem handelt. Weitergehende Forschungen in dieser Richtung bieten noch ein großes Potential.

A Erweiterungen von ALADIN

Das Werkzeug ALADIN dient zum Entwurf von analogen, integrierten Schaltungen. Es bietet die Möglichkeit, qualitativ hochwertige Layouts mit einem vergleichbar geringen Aufwand zu erstellen [Wolf96, Wolf98a, Wolf98b, Wolf99a, Wolf99b, Wolf01, Zhan00, Zhan01, Zhan06a, Zhan06b]. Die Abkürzung ALADIN steht für den Ausdruck Automatic Layout Design Aid for Analog Integrated Circuits. In Abbildung A-1(a) ist ein Überblick zu ALADIN gegeben.



(a) (b)

Abbildung A-1: (a) Strukturübersicht von ALADIN und (b) Ablauf eines Schaltungsentwurfs mittels ALADIN

Das gesamte System setzt sich hauptsächlich aus drei Komponenten zusammen: Design-Assistent, Modulgenerator-Umgebung und Technologie-Interface.

Um eine Optimierung einer analogen Schaltung vom Schaltbild bis hin zum Layout mit Hilfe von ALADIN durchführen zu können, wird durch den Design Assistenten eine graphische Benutzeroberfläche bereitgestellt. Diese ist in das Cadence Design Framework II integriert. Es ist jedoch möglich, mit einem geringen Aufwand, die Oberfläche an andere kommerzielle Software anzupassen. Die Modulgenerator-Umgebung erlaubt Schaltungsentwicklern Module zu erstellen, die unabhängig von der einzusetzenden Technologie sind und auch für andere Anwendungen weiterverwendet werden können. Das Konzept der Erstellung von Modulen in ALADIN besteht darin, dass die Layoutsynthese nicht an eine fest definierte Bibliothek von Generatoren gebunden ist. Es wird vielmehr eine einfache natürliche Beschreibungssprache angeboten, mit deren Hilfe existierende Topologien angepasst oder neue Modulgeneratoren erstellt werden können. Diese natürliche Beschreibungssprache, MOGLAN, die Abkürzung für Module Generator Language, versetzt die Designer in die Lage, hierarchisch parametrisierbare und von der jeweiligen Technologie unabhängige Module zu erstellen. MOGLAN offeriert dazu Sprachkonstrukte, wie Schleifen und Bedingungen. Es werden weiterhin Funktionen bereitgestellt, um einfache Geometrien zu erstellen und zu verdrahten, ohne auf absolute Koordinaten eingehen zu müssen. Die Entwurfsregeln werden automatisch vom Werkzeug erfasst und berücksichtigt. Die grundlegenden geometrischen Objekte

werden durch das Aufrufen von Primitiv-Funktionen generiert. Die Erstellung von komplexeren Modulen erfolgt mittels der Kompaktierung von einfachen geometrischen Objekten oder bereits hierarchisch zusammen gesetzten Strukturen.

Der Ablauf eines Schaltungsentwurfs unter Verwendung von ALADIN ist in Abbildung A-1 (b) veranschaulicht. In einem Schaltplaneditor wird eine bereits fertig dimensionierte Schaltung in diverse Zellen unterteilt. Die parasitären Kapazitäten im Layout der jeweiligen Zellen können durch eine Kapazitäts-Sensitivitätsmatrix kontrolliert werden. Die elektrischen Regeln, wie Elektromigration, Matching, etc., werden gleichzeitig bei der Erstellung der Zellen mit berücksichtigt. Die Schritte von der Auswahl der Bauelemente für eine Zelle bis hin zur automatischen Layouterstellung dieser Zellen werden solange wiederholt, bis alle Elemente zugewiesen und definiert sind. Während der Platzierung der Zellen, welche mit der Verdrahtung einhergeht, wird für jede Zelle eine geeignete Topologie aus den möglichen Varianten ausgewählt. Nach der Generation des Layouts wird ein Extraktionsprozess gestartet. Die resultierenden parasitären Elemente werden automatisch dem Schaltbild hinzugefügt. Es ist nun möglich, die gesamte Schaltung, unter Verwendung einer guten Abschätzung der parasitären Elemente, zu simulieren. Basierend auf den Simulationsergebnissen können die Parameter der Schaltung geändert bzw. Nebenbedingungen gesetzt oder neu definiert werden. Auf diese Weise beginnt die Optimierungsphase der Schaltung.

In diesem Kapitel werden die Erweiterungen von ALADIN, die im Rahmen dieser Arbeit entstanden sind, beschrieben. In [Wolf97] ist eine Beschreibung der Entwurfsregeln präsentiert worden, die eine Technologieunabhängigkeit ermöglicht. Diese Eigenschaft liefert eine hohe Wiederverwendbarkeit der Module. Um eine automatische Erfassung der Entwurfsregeln einer neuen Technologie zu ermöglichen, ist das Technologie-Interface dahingehend überarbeitet worden. Das resultierende Werkzeug wird im nächsten Abschnitt näher erläutert. Um ferner die Verlässlichkeit der Schaltungen zu verbessern, wurde der Designassistent um die Möglichkeit der Stromdichte- und Wärmesimulation erweitert. Die Grundlagen und Funktionsweisen der jeweiligen Programme sind ebenfalls Bestandteil dieses Kapitels. Abschließend erfolgt eine Beschreibung eines Modulgenerators, mit dessen Hilfe Waffel-Transistoren erzeugt werden können.

A.1 TechnoTool

Das Layout einer Schaltung ist einer Reihe von Einschränkungen und Regeln unterworfen. Mit Hilfe einer Entwurfsregelprüfung (Design Rule Check oder kurz DRC) wird getestet, inwiefern das Layout einer Reihe erforderlicher Parameter entspricht. Es ist damit ein entscheidender Schritt bei der physikalischen Verifikation des Schaltungsentwurfs. Bei den Designregeln wird zwischen geometrischen und elektrischen Regeln unterschieden. Um eine Unabhängigkeit von der einzusetzenden Technologie zu erreichen, liest die prozedurale Sprache MOGLAN die notwendigen Entwurfsregeln automatisch ein und beachtet diese während der Layouterstellung. Aufgrund der Tatsache, dass für die Modulgeneratorsprache lediglich die geometrischen Regeln von Interesse sind, sollen die elektrischen Regeln, die ebenfalls automatisch ausgelesen werden können, hier nicht weiter betrachtet werden.

In [Wolf97] ist eine Beschreibung der Entwurfsregeln präsentiert worden, die es der Generatorumgebung ermöglicht, die geforderten Regeln selbständig einzuhalten. Die Grundlage dafür wird durch das Technologie-Interface geschaffen. Zur Adaptation von Entwurfsregeln aus einer neuen Technologie ist im Rahmen der vorliegenden Arbeit ein Werkzeug entstanden, das die notwenigen Informationen eigenständig aus der jeweiligen Technologie extrahiert und in das für

MOGLAN erforderliche Format konvertiert. Das Tool extrahiert aus den in Cadence bereitgestellten Technologie-Daten die Informationen für einfache und komplexere Geometrien, wie beispielsweise die Mindestweite und -länge, Fläche oder die Mindestgröße von Ausbuchtungen. Es werden dazu die Dateien, die mittels des Technology File Manager bereitgestellt werden können und die Beschreibungsdateien des DIVA Design Rule Check ausgewertet. Dabei sind jedoch einige Details zu beachten. Für einige Schaltungselemente sind verschiedene Entwurfsregeln der gleichen Maske notwendig. Zum besseren Verständnis soll dies anhand eines Beispiels verdeutlicht werden.

In Abbildung A-2 sind zwei parallel verlaufende Leitungen dargestellt. Für die Leitungen soll die Maske METAL1 verwendet werden. Der minimale Abstand zwischen den beiden Leitungen bei der zu untersuchenden Technologie ist beispielsweise auf 0.5µm festgesetzt (siehe Abbildung A-2 (a)). Auf der anderen Seite beträgt dieser Mindestabstand 0.8µm, wenn mindestens eine der beiden Leitungen eine Weite von mehr als 10µm aufweist (siehe Abbildung A-2 (b)). Der SKILL-Befehl für die Abstandsprüfung der Anordnung aus Abbildung A-2 (a) hat beispielsweise folgende Gestalt:

```
(drc METAL1 (sep < 0.5) "Minimum METAL1 spacing=0.5")
```

Das Schlüsselwort drc leitet eine Entwurfsregelprüfung ein. Als nächster Parameter wird der Layer der zu prüfenden Elemente erwartet. Die Art der Regelprüfung wird durch ein Schlüsselwort gekennzeichnet. In diesem Fall wird das Schlüsselwort sep verwendet, wodurch eine Abstandsprüfung durchgeführt werden soll. Daneben können noch width, length, enc, notch bzw. area auftreten. Den Abschluss bildet eine Zeichenkette, die im Fall einer Regelverletzung dem Benutzer diese in Form einer Fehlermeldung angezeigt. Eine vollständige Beschreibung des drc-Befehls sowie der weiteren Kommandos kann der Diva Referenz von Cadence [Cade00] entnommen werden. Für die Prüfung der Anordnung aus Abbildung A-2 (b) ist die nachstehende Abfolge von SKILL-Befehlen zuständig:

```
WIDE_METAL1=(geomAnd (geomSize( geomSize METAL1 -5) 5) METAL1)

(drc WIDE_METAL1 METAL1 (sep < 0.8)
```

Mittels der ersten Befehlsfolge werden alle geometrischen Elemente des Layers METAL1 erfasst, die eine Weite von mehr als 10µm aufweisen. Die resultierenden Elemente werden unter der Maske WIDE_METAL1 zusammengefasst. Anschließend kann eine Überprüfung des Abstands zu den verbleibenden Elementen mit der Maske METAL1 durchgeführt werden.

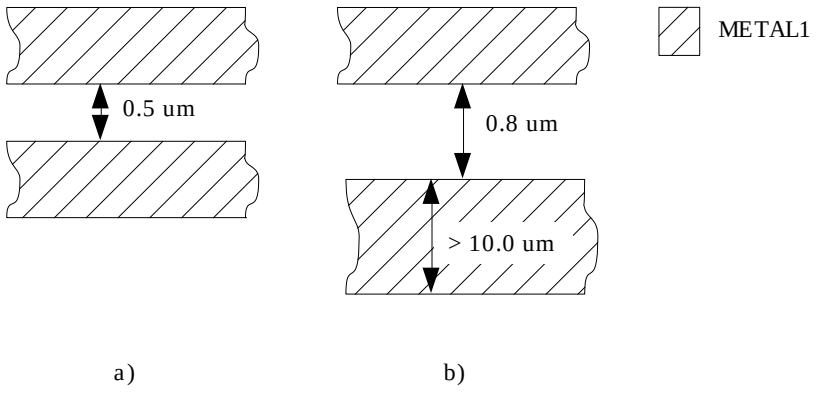


Abbildung A-2: Verschiedene Mindestabstände für METAL1-Leitungen

Um derartigen Regeln genügen zu können, sind durch das Werkzeug TechnoTool mit Hilfe eines Parsers neben den drc-Befehlen auch diejenigen Kommandos zu untersuchen, die zur Beschreibung komplexer Regeln benötigt werden.

Zur Vereinfachung der Prüfung der Entwurfsregeln sind in diversen Technologien zusätzliche Masken definiert worden. Diese Masken entsprechen jedoch keinem Zeichnungslayer. Am Beispiel eines einfachen MOS-Transistors soll dieser Sachverhalt erläutert werden (siehe Abbildung A-3). Mit Hilfe zweier überlappender Rechtecke, die die Layer DIFFUSION und POLY verwenden, wird ein MOS-Transistor erzeugt. Die Diffusionsfläche ist in drei Teile gegliedert: das aktive Gebiet und die Bereiche für Source und Drain. Die Knoten für Source und Drain werden durch die Maske DIFFUSION repräsentiert. Für das Schnittgebiet aus den Masken POLY und DIFFUSION, welches den aktiven Bereich bildet, kann eine neue Maske GATE erstellt werden. Der SKILL-Befehl weist in den meisten Entwurfsregelbeschreibungen die nachfolgende Form auf:

```
GATE = (geomAnd POLY DIFFUSION)
```

Obwohl GATE keinem Zeichnungslayer entspricht, existieren eine Reihe von Entwurfsregeln für die Maske. So wird auf diese Weise verhindert, dass die Kontaktlochreihen in den Gate-Bereich des Transistors eindringen. Ferner wird eine minimale Einfassung zwischen dem Transistor und der n-Wanne gewährleistet.

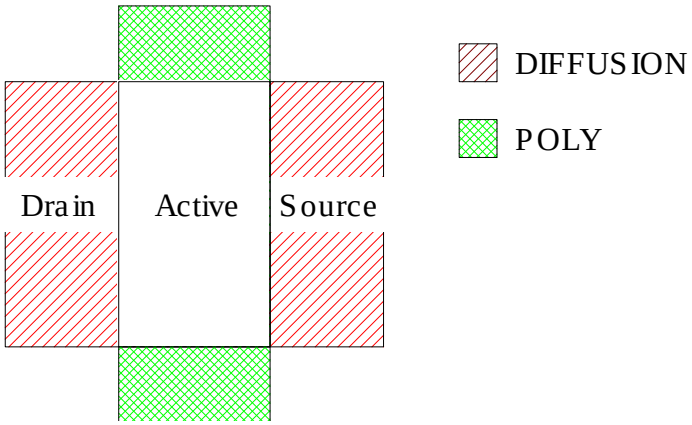


Abbildung A-3: Beispiel für einen MOS-Transistor

Zur Vereinfachung der Bedienung des Technologie-Interface ist eine graphische Benutzeroberfläche implementiert worden. Es ist dafür die Programmiersprache JAVA, unter Verwendung des SwingSets zur Gestaltung der Oberfläche, genutzt worden. Das resultierende Programm TechnoTool bietet neben der Funktion neue Technologien für die Modulgenerator-Umgebung einzulesen, die Möglichkeit bereits vorhandene Technologien zu verwalten und zu bearbeiten. Alle Vorgänge können über Dialoge und Mauseingaben abgewickelt werden. Es offeriert weiterhin Schaltungsentwicklern die Möglichkeit die Namen und die Anzahl der zu verwendenden Masken zu verändern. Wenn ein einheitliches Namenssystem verwendet wird, können die Modulgeneratoren problemlos für verschiedene Technologien genutzt werden, ohne dass diese überarbeitet werden müssen. Aus diesem Grund wird dringend empfohlen, die gleichen Maskennamen zu vergeben.

Das Programm TechnoTool greift grundlegend auf vier verschiedene Packages zurück (siehe Abbildung A-4). Die Klassen für die Elemente der Benutzeroberfläche sind im Package gui enthalten. Die Klassen für einzelnen Befehl der graphischen Oberfläche werden im untergeordneten Package

actions zusammengefasst. Durch das Package fileio werden die Klassen für den Zugriff und die Verwaltung von Dateien bereitgestellt. Zur graphischen Anzeige von Entwurfsregeln wird das Package visualizer verwendet. Es enthält die abstrakte Basisklasse ViewRules und bereits eine Reihe davon abgeleiteter Klassen, die eine Darstellung verschiedener Design-Regeln ermöglichen. Für jede anzuzeigende Maske ist dabei eine eigene Klasse zu erstellen. Das Package technology enthält schließlich die Klassen, mit denen zum einen die bereits extrahierten Technologie-Daten verwaltet werden können. Andererseits beinhaltet dieses Package diejenigen Klassen, aus denen sich der Parser und das Extraktionswerkzeug zur Auswertung der Cadence-Dateien zusammensetzen. Die Dialoge, die für die Ausführung des Parsers notwendig sind, werden in dem untergeordneten Package gui aufgeführt.

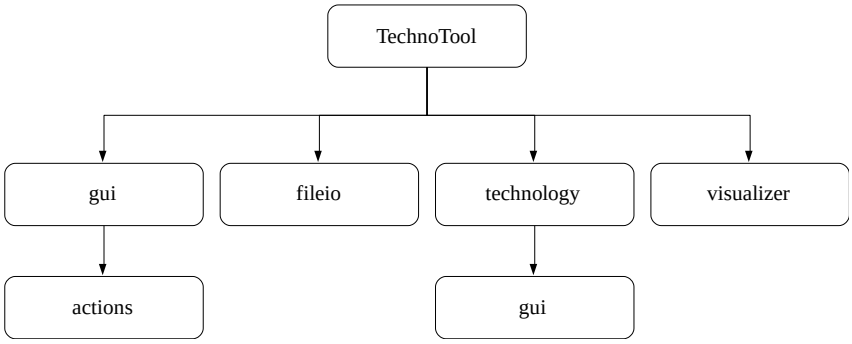


Abbildung A-4: Package-Hierarchie der Anwendung TechnoTool

Infolge der Verwendung der Sprache JAVA und dem modularen Aufwand kann das Programm um weitere Klassen für die Anzeige andere Entwurfsregeln erweitert werden. Fertig übersetzt Klassen können während der Laufzeit der Anwendung hinzugefügt werden. Als Basis dient die Klasse ViewRules des Package visualizer. Die Klasse enthält abstrakte Methoden, die zu überschreiben sind und eine visuelle Darstellung der gewünschten Regeln im Hauptprogramm ermöglichen.

In Abbildung A-5 ist die resultierende graphische Benutzeroberfläche dargestellt. Grundlegend ist die Oberfläche in drei Bereiche gegliedert. Die linke Seite enthält eine Liste der verfügbaren Layer der aktuellen Technologie. Eine visuelle Darstellung der Regelsätze des derzeit ausgewählten Layers ist im rechten oberen Bereich gegeben, sofern eine Klasse zur Anzeige der Regeln des jeweiligen Layers existiert. Unterhalb der graphischen Anzeige werden die Regeln in Textform präsentiert. Mit Hilfe von Befehlen aus der Menüzeile können dialogorientiert neue Technologien erfasst, vorhandene Technologie bearbeitet bzw. entfernt werden.

A.2 Stromdichtesimulation

Die Zuverlässigkeit von integrierten Schaltungen ist eine zentrale Forderung. Mit zunehmender Miniaturisierung kommen jedoch neue, die Zuverlässigkeit negativ beeinflussende Faktoren zum Tragen. Insbesondere ist hier die Elektromigration zu nennen. Dabei handelt es sich um einen Materialtransport infolge einer allmählichen Bewegung der Ionen in einem festen Leiter, verursacht durch den elektrischen Strom. Bedingt durch die Stromdichten erfahren die Verbindungsleitungen eine immer höhere Belastung. Daraus resultiert eine Materialabtragung durch Elektromigration, was zu einer Verminderung der Funktionsfähigkeit führt. Analoge Schaltungen und Stromversorgungsleitungen bei digitalen Schaltungen sind besonders gefährdet. Es handelt sich dabei jedoch nicht um ein Phänomen, welches einen sofortigen Schaltungsausfall verursacht.

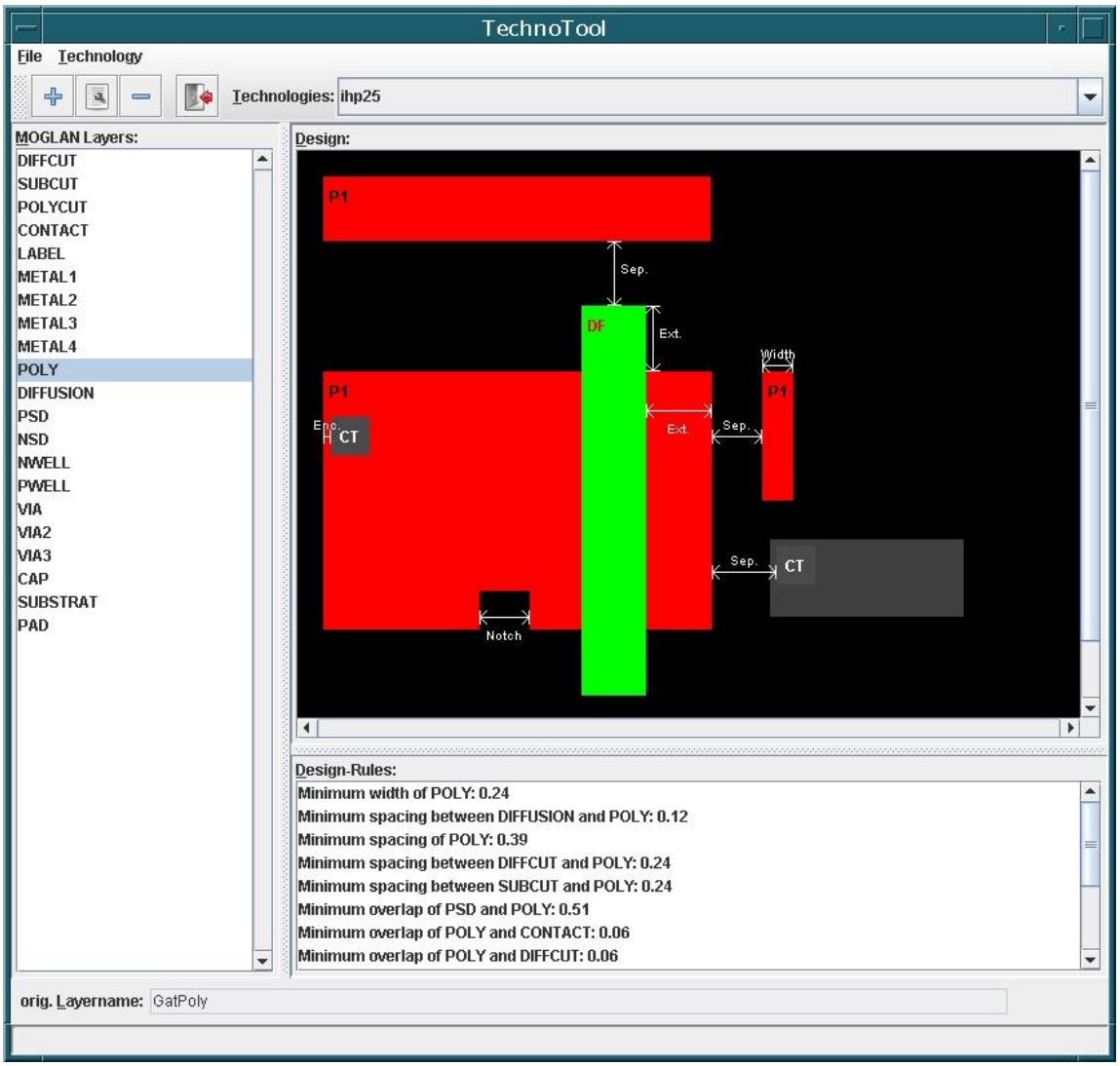


Abbildung A-5: Graphische Benutzeroberfläche des Technologie-Interface

Vielmehr kann es erst nach längerer Betriebszeit zu Widerstandserhöhungen, zu vollständigen Unterbrechungen oder auch zu Kurzschlüssen benachbarter Leitungen infolge von Materialansammlungen kommen.

Die treibende Kraft für die Diffusion der Atome ist die Stromdichte, die sich aus dem Betrag des Quotienten vom Strom zur Querschnittsfläche ergibt. Weiterhin wird bei den meisten Prozesstechnologien von einer festen Leiterbahnhöhe ausgegangen. Aus diesem Grund kann eine größere Beständigkeit gegen die Elektromigration durch eine Erhöhung der Leiterbahnbreite erzielt werden. Die Elektromigration hängt aber noch von vielen weiteren Faktoren ab, wie beispielsweise der Temperatur - eine erhöhte Temperatur führt zu einer deutlichen Beschleunigung des Diffusionsvorgangs - oder dem Verlauf der stromführenden Leitungen. Eine besondere Beachtung erfordern dabei Abknickungen von Leiterbahnen. Insbesondere sind 90°-Winkel in Bereichen mit hohen Stromdichten zu vermeiden. Die Belastungen in einem derartigen Eckpunkt liegen deutlich über denen von abgeflachten Ecken, wie den 135°-Winkel [Lien05, Lien06].

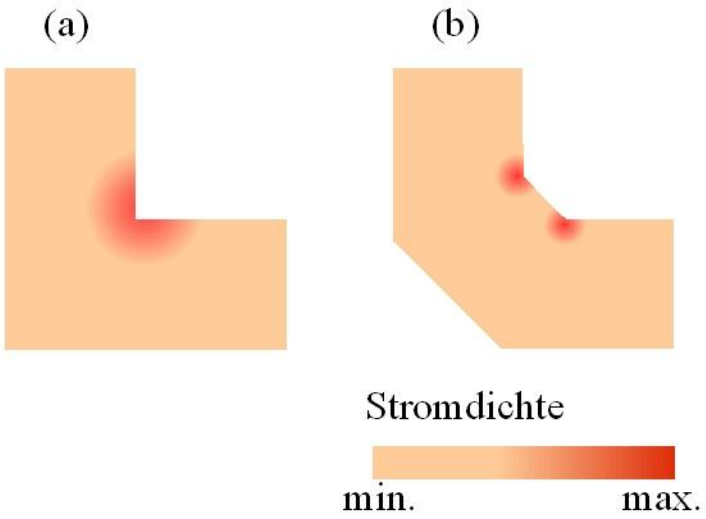


Abbildung A-6: Stromdichteverteilung an (a) 90°- und (b) 135°-Ecken

Im Allgemeinen ist die Extraktion von Widerständen deutlich aufwändiger als die von Kapazitäten. Diese können hinreichend genau aus der Fläche und dem Umfang der überlappenden Gebiete abgeschätzt werden. Die Berechnung ist sehr einfach, wenn die Kanten der Areale gegeben sind. Dahingegen hängt der Widerstand von der Art und Weise ab, wie der Strom durch das zu untersuchende Gebiet fließt. Die entscheidenden Faktoren sind dabei die Form des Bereichs und die Lage der möglichen Kontakte. Üblicherweise besteht das Ermitteln des Widerstands zwischen einer Reihe von Anschlüssen darin, die Laplace Gleichung $\nabla^2 \Psi = 0$, mit den Randbedingungen, die durch die Form der Fläche und den Kontakten gegeben sind, zu berechnen. Es gibt diverse Ansätze zur numerischen Lösung der Gleichungen [Bark85, Ramo65]. Diese beschränken sich jedoch in der praktischen Umsetzung auf kleine Gebiete und sind für sehr umfangreiche Schaltungslayouts eher ungeeignet.

Der Widerstandswert kann auch als ein geometrischer Faktor mit einer technologie-abhängigen Konstante aufgefasst werden. Eine Skalierung der Abmessungen eines Widerstands um einen einheitlichen Faktor führt nicht zu einer Veränderung des ohmschen Widerstands. Daher ist es zweckdienlich, die Länge eines Widerstands im Hinblick auf dessen Weite zu erfassen. Für einen rechteckigen Widerstand gilt damit, dass dessen ohmscher Wert sich durch die Anzahl der quadratischen Gebiete ausdrücken lässt. Dieser Sachverhalt kann auf beliebige Formen verallgemeinert werden, indem das zu untersuchende komplexe Polygon in einfachere Formen unterteilt wird. Es bleibt dann lediglich der Widerstandswert anhand der ermittelten Bereiche zu berechnen. Die Idee ist beispielsweise in [Horo83, Ozak80, Yosh79] zur Extraktion von Widerständen aus einem Layout aufgegriffen worden. Das Problem ist die Wahl der jeweiligen Schnittlinien. So sollte der Verlauf der Stromdichten nicht nachhaltig durch die Unterteilungen beeinflusst werden. Dazu wird in [Horo83] eine Heuristik, basierend auf dem Stromfluss, verwendet. In [Ozal80, Yosh79] beschränken sich die Betrachtungen auf einfache Längen/Weiten-Berechnungen.

Ein weiterer Ansatz zur Extraktion von Widerständen ist in [Lada93] präsentiert worden. Die Berechnung der Widerstandswerte von beliebig geformten Polygonen erfordert in dem Fall keine Zerlegung des Gebietes in einfachere Formen. Stattdessen wird der Stromfluss mit Hilfe eines Verdrahtungsalgorithmus ermittelt.

Wie bereits erwähnt wurde, ist in früheren Publikationen der Widerstandswert von beliebig geformten Strukturen, wie beispielsweise den 90°- bzw. 135°-Leitungsknicken, durch Lösen der Poisson-Gleichung berechnet worden [Horo83, Meij95, Sun97]. Die Berechnungen erfolgen dabei unter Verwendung der Finiten-Element-Methode bzw. der Randelementmethode. Das hier präsentierte Werkzeug verfolgt einen modifizierten Ansatz. Anstatt die Potentialverteilung direkt zu berechnen, wird eine einfache Methode zur Unterteilung eines beliebigen Polygons in gleich große Quadrate angewandt. Die grundlegende Idee ist in Abbildung A-7 veranschaulicht. Jedes Quadrat in der jeweiligen Struktur wird durch vier Widerstände modelliert. Die Widerstände sollen den Stromverlauf repräsentieren.

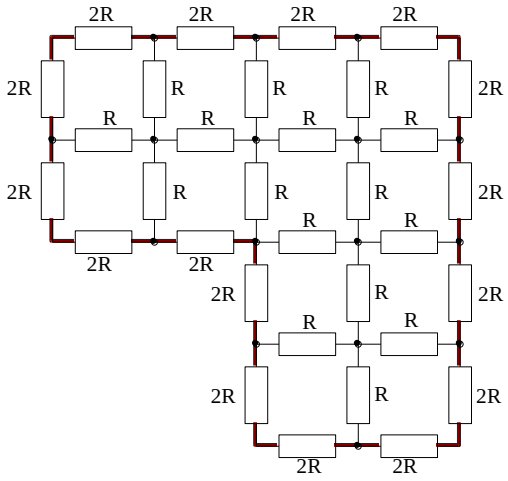


Abbildung A-7: Unterteilung eines 90°- Leitungsknicks in gleich große Quadrate

Eine Seite eines Quadrats entspricht dabei einem Widerstand. Der Wert eines Widerstands ist durch den zweifachen Flächenwiderstandswert des zugrundeliegenden Layers des Polygons gegeben. Auf diese Weise ergibt sich ein Widerstandsnetzwerk, mit dessen Hilfe die Verteilung der Stromdichte ermittelt werden kann. Nach Angabe der Bereiche, in denen der Strom in die ausgewählte Struktur hinein- bzw. herausfließt, kann das Netzwerk berechnet werden. Die Ergebnisse liefern näherungsweise eine Verteilung der Stromdichte in der zu untersuchenden Struktur. Die Abmessungen der Quadrate können beliebig angepasst werden, wodurch ein Einfluss auf die Genauigkeit und den Zeitbedarf einer Simulation genommen werden kann.

Im nachfolgenden Algorithmus A-1 ist der Ablauf für eine Stromdichte-Simulation unter Ausnutzung des genannten Modells skizziert.

Im ersten Schritt des Verfahrens werden die Daten, die die zu untersuchende Struktur beschreiben, eingelesen. Die Geometrie-Daten liegen im SKILL-Format vor. Es handelt sich dabei um die Interpretersprache von Cadence. Dieses Format wird bei der Ausgabe durch die Modulgeneratoren von MOGLAN verwendet. Es ist jedoch mit einem geringen Aufwand möglich, die Eingabe an andere Formate anzupassen. Bei Verwendung der Geometrie-Daten, die mit Hilfe von MOGLAN erzeugt wurden, kann eine Gruppierung der Objekte in zu untersuchende Netze erfolgen. Durch MOGLAN wurde jedem Objekte eine Netzzugehörigkeit zugewiesen. Andernfalls werden sich überlappende bzw. berührende geometrische Elemente der gleichen Maskenebene zu jeweils einem Netz zusammengefasst. Eine mögliche Überlappung der Objekte wird anschließend entfernt. Im nachfolgenden Schritt wird über alle ermittelten Netze iteriert. Wenn die Elemente eines Netzes nicht bereits als ein Polygon vorliegen, so werden diese durch geeignete booleschen Funktionen zu

einem bzw. mehreren Polygonen zusammengefasst. Es wird dabei auf Algorithmen zurückgegriffen, die in [Grei98, Liu03] erstmals vorgestellt wurden.

Algorithmus A-1: Ablauf einer Stromdichte-Simulation

```
Einlesen und Aufbereiten der geometrischen Daten;
Ermitteln der möglichen Netze aus den eingelesenen Daten;
für alle zu untersuchenden Netze
   alle zusammenhängenden Objekte des aktuellen Netzes mit dem gleichen Layer zu
   einem Polygon zusammenfassen;
   wenn keine Vorgabe für Kantenlänge der Quadrate existiert dann
      minimale Kantenlänge der Quadrate für das jeweilige Polygon berechnen;
   wenn_ende
   Widerstandsnetzwerk für das jeweilige Polygon erstellen;
für_ende
suche nach Kontakten, die Polygone des gleichen Netzes verbinden;
wenn Verbindungen existieren dann
   verbinde korrespondierende Widerstandsnetzwerke;
wenn_ende
Positionen der Ein- und Ausgänge einlesen;
Knotenadmittanzmatrizen und Eingangsvektoren erstellen;
Berechnungen der linearen Gleichungssysteme durchführen;
Ergebnisse der Berechnungen in eine Datei schreiben;
```

Wenn keine Vorgabe, hinsichtlich der Kantenlänge der Quadrate, durch den Benutzer getroffen wurde, so wird diese automatisch ermittelt. Dazu wird für das jeweilige Polygon der größte gemeinsame Teiler der horizontal, vertikal und ggf. diagonal verlaufenden Seiten ermittelt. Aus diesen drei Werten wird ebenfalls der größte gemeinsame Teiler berechnet. Der resultierende Wert stellt die Kantenlänge der Quadrate dar, in das Polygon gegliedert werden soll. Mit Hilfe eines Scanline-Verfahrens wird das Widerstandsnetzwerk für das aktuelle Polygon nach dem eingangs vorgestellten Modell erstellt. Nachfolgend wird nach Kontakten gesucht, die die Polygone des gleichen Netzes, jedoch mit unterschiedlichen Layern, miteinander verbinden. Falls derartige Verbindungen gefunden werden, so erfolgt eine Fusionierung der entsprechenden Widerstandsnetzwerke. Der Widerstandswert für den Kontakt wird dabei aus der zu verwendenden Technologie entnommen.

Nachdem die Netzwerke erstellt worden sind, werden die Positionen der Ein- bzw. Ausgänge eingelesen. Mit Hilfe der bis zu diesem Schritt gewonnenen Daten werden die Knotenadmittanzmatrizen und Eingangsvektoren erzeugt. Anschließend werden die resultierenden linearen Gleichungssysteme mittels numerischer Verfahren berechnet. Die Ergebnisse werden in eine Datei geschrieben und stehen damit für eine folgende Auswertung durch den Benutzer zur Verfügung.

Das Programm zur Simulation der Stromdichte ist in C++ geschrieben worden. Für die zugehörige graphische Benutzeroberfläche wurde die Sprach TCL/Tk verwendet. Mit Hilfe der Oberfläche können das Simulationswerkzeug bedient und die Ergebnisse ausgewertet werden. Beide Programmteile sind in das Designpaket ALADIN integriert, können jedoch unabhängig von Cadence oder einer anderen kommerziellen CAD Software genutzt werden.

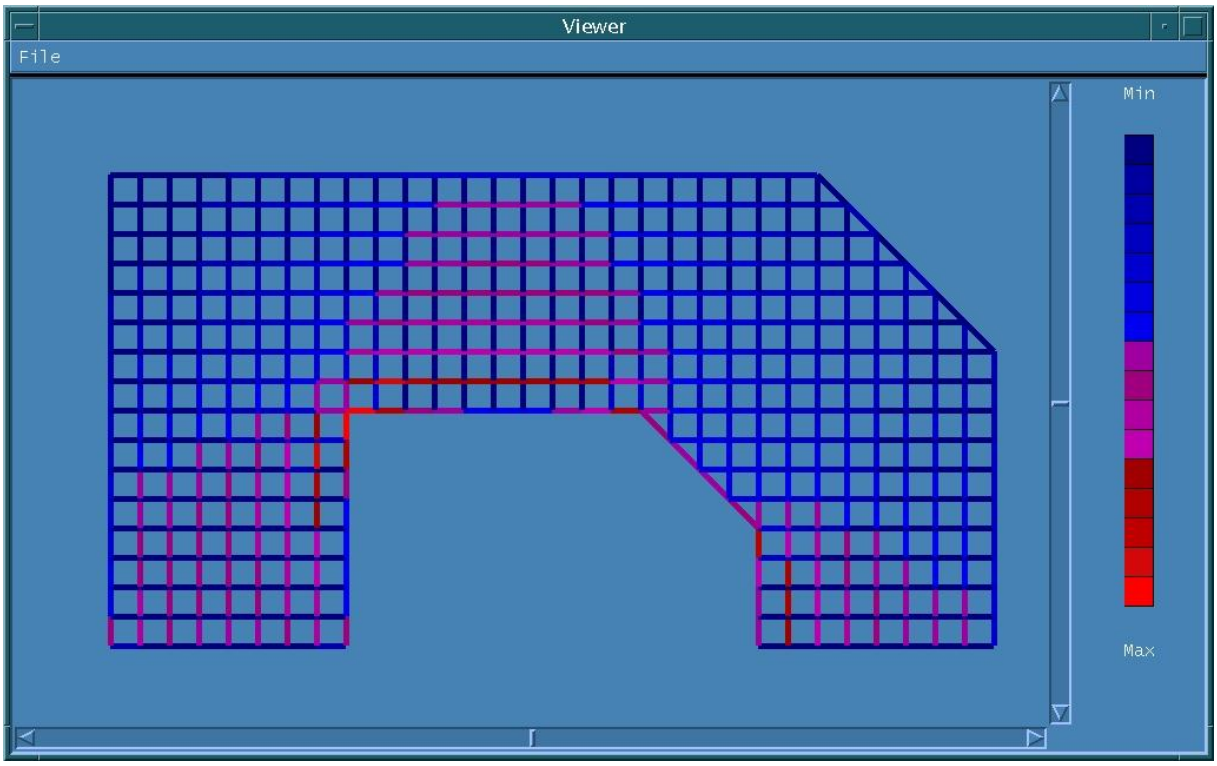


Abbildung A-8: Simulierte Stromdichte einer aus einem 90° und einem 135°-Knick zusammengesetzten Form

Der Abbildung A-8 sind die Ergebnisse einer erfolgreich durchgeführten Simulation für eine einfache Struktur zu entnehmen. Die linke Hälfte des Polygons besteht aus einer rechtwinklig verlaufenden Leitung, während der rechte Bereich von 45° Ecken Gebrauch macht. Der zu simulierende Stromfluss verläuft von der linken unteren Seite über den Bogen zum rechten unteren Rand. Wie aus der Abbildung ersichtlich ist, weist der innere Eckpunkt des 90°-Bogens die höchste Stromdichte auf. Demgegenüber ist im umliegenden Gebiet und auch bei den 45° Ecken eine deutlich geringere Belastung zu beobachten. Neben diesem einfachen Beispiel können mit Hilfe des Werkzeugs auch komplexere Strukturen im Hinblick auf die Stromdichte untersucht werden.

A.3 Wärmesimulation

Die zunehmende Miniaturisierung integrierter Schaltungen erfordert immer mehr Aufmerksamkeit im Hinblick auf die elektrothermischen Wechselwirkungen. Eine große Anzahl an Vorgängen in einem IC sind von der Betriebstemperatur abhängig, wie beispielsweise Offset Spannungen, elektrische Überlastungen oder der Effekt der Elektromigration. Aus diesem Grund benötigen die Designer von integrierten Schaltungen, die ein kritisches thermoelektrisches Verhalten aufweisen, weitere Unterstützung während der Entwurfsphase. In [Chen96, Tsai00, Lee93] sind Ansätze zur Lösung der elektrothermischen Probleme in elektronischen Schaltungen dargelegt.

Die Eingabedaten für die Wärmesimulation basieren auf den Simulationsergebnissen einer DC-Analyse des Analog Extrakts der zu untersuchenden Schaltung. Genauer gesagt wird die Verlustleistung der Transistoren verwendet und vom Tool automatisch extrahiert. Für die weitere Berechnung wird davon ausgegangen, dass die Wärmeleitfähigkeit anisotherm, die Wärmeverteilung isotropisch ist und die Simulation nur für den stationären Betrieb durchgeführt werden soll. Wenn die genannten Bedingungen erfüllt sind, ergibt sich für die Wärmeleitung die nachstehende Gleichung

$$\lambda(x,y,z)\cdot\left[\frac{\partial^2}{\partial x^2}T(x,y,z)+\frac{\partial^2}{\partial y^2}T(x,y,z)+\frac{\partial^2}{\partial z^2}T(x,y,z)\right]=-g(x,y,z), \tag{A-1}$$

wobei T die Temperatur, g die Leistungsdichte der Wärmequelle und λ die thermale Leitfähigkeit darstellen. Die Lösung der Differentialgleichung mit Hilfe von numerischen Lösungsverfahren liefert das folgende, umfangreiche lineare Gleichungssystem:

$$A\cdot\vec{T}=\vec{G}. \tag{A-2}$$

wobei T den Vektor der zu berechnenden Temperaturen beschreibt, A die Matrix der thermischen Leitwerte enthält und G den Vektor der Eingabe-Wärmequellen beinhaltet. Die Größe des Gleichungssystems in Gl. (A-2) ist abhängig von Abmessungen der zu untersuchenden Schaltung und der Auflösung des Gitters. Zur Lösung des linearen Gleichungssystems ist das Verfahren der konjugierten Gradienten implementiert worden [Brae07, Hest52, Meis99, Plat06]. Es ist eine effiziente Methode zur Lösung von großen, symmetrischen, positiv definiten Gleichungssystemen der Form $A\vec{x}=\vec{b}$. Der Algorithmus konvergiert nach spätestens m Schritten, wobei m der Dimension der quadratischen Koeffizientenmatrix A entspricht. Für das konjugierte Gradientenverfahren wird die folgende Gleichung betrachtet

$$F(\vec{x})=\frac{1}{2}\langle A\vec{x},\vec{x}\rangle-\langle\vec{b},\vec{x}\rangle+c, \tag{A-3}$$

wobei $\langle\cdot,\cdot\rangle$ das euklidsche Skalarprodukt bezeichnet und $A\in\mathbb{R}^{m\times m}$ eine symmetrische, positiv definite Matrix darstellt. Der Gradient der partiell differenzierbaren Funktion F lautet damit:

$$\nabla F(\vec{x})=A\vec{x}-\vec{b}=G(\vec{x}). \tag{A-4}$$

Der Ansatz $\nabla F(\vec{x})=0$ zur Berechnung des Extremums führt zu dem linearen Gleichungssystem $A\vec{x}=\vec{b}$. Es wird somit derjenige Vektor x gesucht, für den die Funktion F minimal wird. Nachfolgend wird der Ablauf des Verfahrens der konjugierten Gradienten kurz skizziert. Die Voraussetzungen sind durch Gl. (A-3) und Gl. (A-4) gegeben. Es ist zu Beginn ein beliebiger Startvektor $x\in\mathbb{R}^n$ zu wählen und der erste Residuenvektor r_0 zu berechnen

$$G_0=G(\vec{x}_0),\overrightarrow{r_0}=-G_0. \tag{A-5}$$

In jedem anschließenden Iterationsschritt k wird der Vektor x_{k+1} durch die Korrektur des Vektors x_k in die Richtung r_k berechnet. Damit gilt

$$\vec{x}_{k+1}=\vec{x}_k+\alpha_k\cdot\vec{r}_k. \tag{A-6}$$

wobei der Koeffizient α_k sich wie folgt berechnet

$$\alpha_k=-\left(A\vec{x}_k-\vec{b}\right)^T\cdot\frac{\vec{r}_k}{\|\vec{r}_k\|_A^2}=\frac{\|\nabla F(\vec{x}_k)\|^2}{\|\vec{r}_k\|_A^2}. \tag{A-7}$$

Als Vorbereitung für den nächsten Iterationsschritt ist ein neuer Residuenvektor r_{k+1} zu berechnen.

$$\vec{r}_{k+1}=-\nabla F(\vec{x}_{k+1})+\beta_k\vec{r}_k, \tag{A-8}$$

wobei für β_k gilt

$$\beta_k=\frac{\nabla F(\vec{x}_{k+1})\cdot A\cdot\vec{r}_k}{\|\vec{r}_k\|_A^2}=\frac{\|\nabla F(\vec{x}_{k+1})\|^2}{\|\nabla F(\vec{x}_k)\|^2}. \tag{A-9}$$

Wenn der Fall $G_k = G(\vec{x}_k) = 0$ vorliegt, dann kann das Verfahren beendet werden und der Vektor $\vec{x}_k$ ist das Minimum von F. Basierend auf den präsentierten Gleichungen wird das Gleichungssystem zur Temperaturverteilung berechnet. Insgesamt stellen die Ergebnisse, die aus der Lösung des linearen Gleichungssystems hervorgehen, lediglich eine grobe Abschätzung der Temperaturverteilung zwischen den Transistoren dar.

Algorithmus A-2: Ablauf einer Wärmesimulation

```
Einlesen der Grafikdaten;
Einlesen der Transistordaten;
Grafikdaten nach Layern gruppieren;
Bereinigung der Grafikdaten durchführen;
zusammenhängende Gebiete erkennen und extrahieren;
nach Überschreitungen zwischen POLY und DIFFUSION-Gebieten suchen;
für alle gefundenen Schnittgebiete
   aktuellem Schnittgebiet den Transistornamen und die Wärmeleistung zuweisen;
für_ende
Widerstandsnetzwerk erstellen;
Aufstellen der Knotenadmittanzmatrix und des Eingabevektors;
Lösen des linearen Gleichungssystems;
Ergebnisse in einer Datei speichern;
```

Der Ablauf einer Wärmesimulation ist in Algorithmus A-2 zusammenfassend dargestellt. Im Designframework von Cadence werden mit Hilfe des erweiterten Designassistenten die geometrischen Elemente, aus denen sich die zu untersuchende Schaltung zusammensetzt, sowie die Simulationsdaten der Transistoren in entsprechende Dateien geschrieben. Diese Dateien werden im ersten Schritt der Wärmesimulation eingelesen. Dabei werden die graphischen Objekte nach Layern gruppiert. Anschließend erfolgt eine Untersuchung der Grafikdaten der jeweils gleichen Maskenschicht auf eine mögliche Überlappung, die zu entfernen ist. Die aneinandergrenzenden Elemente, die den gleichen Layer aufweisen, werden danach zu einem Polygon vereinigt. Im nächsten Schritt werden die MOS Transistoren aus den geometrischen Daten extrahiert. Dazu wird nach Überschneidungen von Elementen mit den Layern POLY und DIFFUSION gesucht. Jedem auf diese Weise ermittelten Schnittgebiet werden die korrespondierenden Namen und Wärmeleistungen aus den zu Beginn eingelesenen Transistordaten zugewiesen.

Analog zur Stromdichte-Simulation wird das Layout der Schaltung anschließend in gleich große Quadrate unterteilt. Erneut repräsentieren die Kanten der Quadrate die Widerstände. Als Widerstandswert wird hingegen der spezifische Wärmewiderstands des jeweiligen Layers verwendet. Auf diese Weise kann mit dem Modell aus dem vorherigen Abschnitt ein Widerstandsnetzwerk aufgebaut werden. Um die Wärmeverteilung berechnen zu können, wird das resultierende Netzwerk in die Knotenadmittanz-Darstellung überführt. Als Eingabevektor werden die Wärmeleistungen der jeweiligen Transistoren verwendet. Nach Lösen des linearen Gleichungssystems mittels des beschriebenen Verfahrens der konjugierten Gradienten, werden die Ergebnisse der Simulation in eine Datei geschrieben. Diese Ergebnisse werden mittels des Designassistenten automatisch in das Analogextrakt der zu untersuchenden Schaltung eingefügt. Eine anschließende DC Simulation der Schaltung kann die möglichen Temperatureinflüsse verdeutlichen. Anhand der Resultate ist der Schaltungentwickler in der Lage, die Einflussnahme der Temperatur auf bestimmte Zellen der Schaltung zu beurteilen und ggf. eine andere Anordnung zu wählen.

Die Benutzerschnittstelle für das Werkzeug zur Wärmesimulation ist in SKILL, der Interpretersprache von Cadence, entworfen worden. Die Umsetzung der Algorithmen zur Berechnung der Wärmeverteilung erfolgt in C++.

A.4 Waffel-Transistor

Das Layout eines MOS-Transistors wird durch die einfache Überlappung zweier Rechtecke erreicht: Eines definiert das aktive Gebiet, das andere das Polysilizium-Gate. Diejenigen Bereiche des aktiven Bereichs, die nicht durch das Gate geschützt werden, bilden das Source- bzw. Drain-Gebiet. Bei der Layouterstellung von Transistoren sind weiterhin die folgenden Punkte zu berücksichtigen:

- parasitäre Widerstände am Source und Drain müssen so niedrig wie möglich gehalten werden,
- parasitäre Kapazitäten sollten minimiert werden,
- das Matching-Verhalten von zusammengehörenden Elementen ist zu berücksichtigen.

Eine Reduzierung der parasitären Kapazitäten kann durch die Verwendung eines gefalteten Transistors erreicht werden (Reduzierung um den Faktor 2). Mit Hilfe eines Waffel-Transistors kann diese Kapazität um den Faktor 4 gemindert werden, da an einem Source- bzw. Drain-Gebiet vier Gates angrenzen [Malo01].

Das Waffel-Layout wird bei MOS-Transistoren mit einer sehr großen Weite eingesetzt. Es ermöglicht einen größeren Drainstrom und damit auch eine höhere Treiberleistung bei einer kleineren Fläche im Vergleich zum herkömmlichen Layout [Gime09]. In Bezug auf das Matching-Verhalten ist in [Bast96] durch Messungen gezeigt worden, dass MOS-Transistoren im Waffel-Layout keinen systematischen Matching-Fehler aufweisen. Das Matching ist vielmehr linear abhängig vom Inversen der Kanalfläche der Transistoren. Ein großer Nachteil der Waffelstruktur liegt darin, dass die Weite nicht hinreichend genau festgelegt werden kann. Ebenso ist die Länge der Transistoren nicht exakt definiert. An den Kreuzungspunkten der einzelnen Gates verhält sich der Transistor nicht wie ein MOSFET. Das elektrische Feld ist an dieser Stelle gleich Null. Damit fließt im Zentrum dieses Gebiets kein Strom.

Der Einsatzbereich für Transistoren im Waffel-Layout ist bei Anwendungen zu finden, die einen großen Drainstrom benötigen, während das exakte Weiten-Längen-Verhältnis eine untergeordnete Bedeutung aufweist. Die Waffel-Struktur wird andernfalls nur selten genutzt, da neben dem Seitenverhältnis die Angaben für Länge und Weite eines Transistors nicht klar definiert werden können. Weiterführende Informationen bezüglich des Waffel-Layouts für MOS-Transistoren sind in [Zhan02] zu finden.

A.4.1 Modulgenerator zur Erstellung von Waffel-Transistoren

Im Nachfolgenden wird die Funktionsweise des Modulgenerators zur Erstellung von MOS-Transistoren im Waffel-Layout beschrieben. In Abbildung A-9 ist das Layout eines Waffel-Transistors nach erfolgreicher Ausführung des Generators dargestellt. Für die Erstellung des Transistorlayouts ist eine 0.35µm CMOS Technologie der Firma austriamicrosystems verwendet worden.

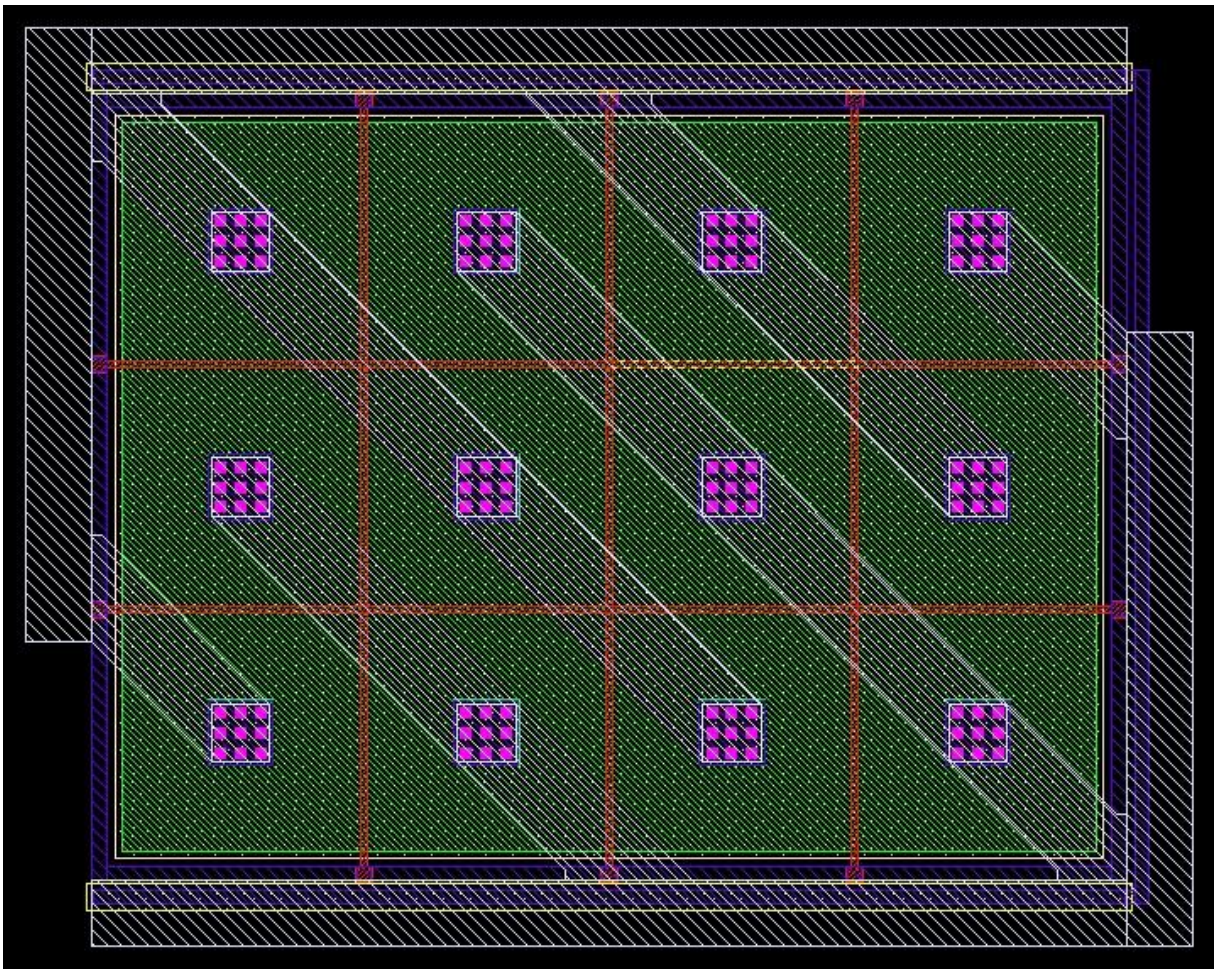


Abbildung A-9: Layout eines MOS-Transistors unter Verwendung der Waffel-Struktur

Als Parameter stehen die Gesamtweite und -länge des zu erstellenden Transistors zur Verfügung. Ebenso kann die Weite der Metallbahnen angegeben werden. Der Generator berechnet diese automatisch, wenn der Parameter nicht spezifiziert wird. In diesem Fall entspricht die Weite der Leitungen mindestens dem in den Entwurfsregeln gespeicherten Wert. Weitere Parameter sind die Anzahl der Felder aus den sich der Waffel-Transistor zusammensetzen soll. Der Wert wird durch die Anzahl der Spalten und Zeilen festgelegt. Die Anzahl der Einzeltransistoren, aus denen sich der Waffeltransistor zusammensetzt, berechnet sich damit wie folgt:

$$\#\text{Transistoren} = (2 \cdot \#\text{Zeilen} \cdot \#\text{Spalten}) - (\#\text{Zeilen} + \#\text{Spalten}). \quad \text{(A-10)}$$

Mit Hilfe des letzten Parameters schließlich kann angegeben werden, ob ein n-Kanal oder p-Kanal MOSFET erstellt werden soll. Der generelle Ablauf der Erstellung eines Waffel-Transistors ist in Abbildung A-10 skizziert.

Eine einzelne Zeile setzt sich aus den Source- bzw. Drain-Anschlüssen der einzelnen Transistoren zusammen (siehe Abbildung A-10 (a)). Dabei wird aus je zwei POLY Rechtecken eine L-Form erstellt und an einen Kontakt herankompaktiert. Den Abschluss einer Zeile bilden ein Kontakt und ein horizontal verlaufendes POLY Rechteck. Die resultierenden Reihen werden aneinander kompaktiert (siehe Abbildung A-10 (b)). Der Vorgang wird solange wiederholt, bis die um Eins reduzierte Anzahl an Zeilen vorliegt. Für die letzte Zeile wird eine Reihe, bestehend aus der wechselnden Abfolge Kontakt - vertikales POLY Rechteck, an die bereits vorhandene Struktur kompaktiert. Die Anzahl der

Kontakte entspricht der Vorgabe der Spaltenzahl (siehe Abbildung A-10 (c)). Im nächsten Schritt wird um die Struktur ein Rechteck mit der Maske DIFFUSION gelegt. An die Gate-Anschlüsse, die sich an den Rändern der Struktur befinden, werden Kontakte herankompaktiert (siehe Abbildung A-10 (d)). Auf diese Weise ist es möglich, die Gates der einzelnen Transistoren im nächsten Schritt mit einem Metall-Ring zu verbinden (siehe Abbildung A-10 (e)). Es werden ferner die jeweiligen Source- bzw.-Drain-Anschlüsse mit einer METALL2 Maske diagonal angeschlossen. Abschließend bleiben die Wannenkontakte zu erstellen. Als Ergebnis ergibt sich beispielsweise das in Abbildung A-9 gezeigte Layout eines MOS-Transistors unter Verwendung einer Waffel-Struktur.

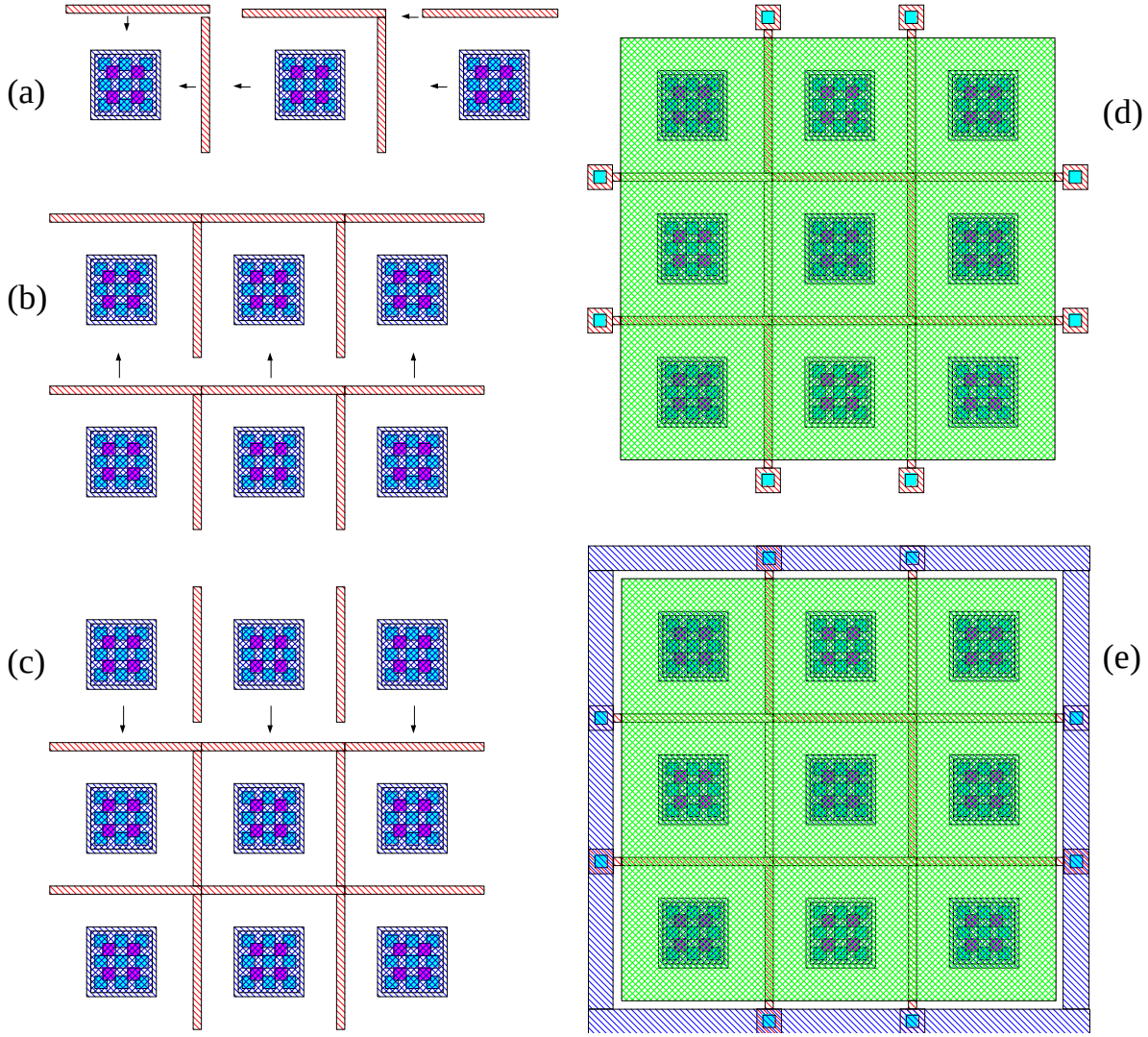


Abbildung A-10: Abfolge zur Erstellung eines Waffel-Transistors

B Beschreibung von Verfahrensabläufen

Zur Beschreibung der Algorithmen wird eine um natürlich-sprachliche Elemente pascalähnliche Schreibweise verwendet. Ein mit Hilfe dieses Pseudo-Codes beschriebenes Verfahren besteht aus einer Folge von Anweisungen. Einige Anweisungen enthalten wiederum Anweisungsfolgen, die durch die Schlüsselworte **`begin`** und **`end`** eingeschlossen sind. Es werden folgende Anweisungen unterschieden (Schlüsselworte sind fett, Platzhalter in spitze Klammern eingeschlossen):

- Wertzuweisung:

  <Variable> := <Ausdruck>

  Der Wert einer Variablen ergibt sich beispielsweise durch die Auswertung eines arithmetischen Ausdrucks.

- Bedingte Anweisung:

  `wenn` <Ausdruck> **`dann`**

  <Anweisung(en)>

  `sonst`

  <Anweisung(en)>

  `wenn_ende`

  Ein Boolescher Ausdruck besteht entweder aus einem einzelnem Vergleich oder einer Folge von logisch durch die Schlüsselworte und/oder verknüpften Vergleichen.

- Laufanweisung:

  `für` <Variable> := <Startwert> **`bis`** <Endwert>

  <Anweisung(en)>

  `für_ende`

  Die Laufanweisung kann aus Gründen der einfacheren Darstellung auch folgende Form aufweisen:

  `für` <Variable> $\in$ <Menge>

  <Anweisung(en)>

  `für_ende`

Dies bedeutet, dass die Anweisung für jedes Mengenelement ausgeführt werden soll.

- Schleifen:

solange <Ausdruck>

<Anweisung(en)>

solange_ende

Die Anweisung wird so lange ausgeführt wiederholt ausgeführt, bis der (Boolesche) Ausdruck den Wahrheitswert falsch aufweist.

wiederhole

<Anweisung(en)>

bis <Ausdruck> **wiederhole_ende**

Die Anweisung wird so lange (mindestens einmal) ausgeführt, bis der (Boolesche) Ausdruck den Wahrheitswert wahr liefert.

Mit Hilfe der Schlüsselworte Fortfahren bzw. Unterbrechen kann der aktuelle Schleifendurchlauf beendet und die Schleife mit der nächsten Iteration fortgesetzt bzw. die gesamte Schleife abgebrochen werden.

Darüber hinaus werden zur Vereinfachung natürlich-sprachliche Anweisungen verwendet, wie beispielsweise: füge das Elemente x an das Ende der Liste L an.

C Liste der verwendeten Symbole

Symbol	Bedeutung	Einheit
Δ_{cnt}	Mindestabstand zwischen Kontakten	m
Δ_{W}	Leitungsbreite	m
$\Pi = (\pi_{ij})$	Vorgängermatrix	
α_A^{-1}	Position der Zelle A in der Folge α	
α_{cap}	Gewichtung der kapazitiven Einkopplungen	
α_{i}	Gewichtungsfaktor	
α_{L}	Gewichtung der Weglänge	
$\alpha_{\mathrm{net(k)}}$	Priorität des k-ten Netzes	
γ	Faktor, abhängig vom Leitungstyp eines Pfades	
δ	Eindringtiefe	m
δ_{ij}	Gewicht der Kante zwischen Knoten i und j	
$\delta_{K,j}$	Gewicht der Kanten vom Knoten K zum j-ten Knoten der kompaktierten Zelle	
$\delta_{Z,i}$	Gewicht der Kanten vom Knoten Z zum i-ten Knoten der Zielzelle	
ε_0	Dielektrizitätskonstante Vakuum	$8.854 \cdot 10^{-12} Fm^{-1}$
ε_{ox}	Dielektrizitätskonstante Siliziumoxid	3.9
ε_r	Dielektrizitätskonstante Substrat	
η_i	Kapazitätsverhältnis	
μ	magnetische Permeabilität	$VsA^{-1}m^{-1}$
ρ	spezifischer Widerstand	Ωm
$\bar{\rho}$	mittlerer spezifischer Widerstand	Ωm
$\pi_{ij}^{(k)}$	Vorgänger des Knoten v_j auf dem kürzesten Weg vom Knoten v_i mit k inneren Knoten	
A	Fläche	m²
a, d	Leitungsabstand	m
a_i	Alternative	
$C = (c_{ij})$	Kostenmatrix	
C	Kapazität	F
C_i	Kapazitätsgrößen	F
C_L	Lastkapazität	F
C_{min}	minimale Kapazität	F
C_{ox}	Kapazität zwischen Leitungsebene und Substrat	F
C_{si}	parasitäre Kapazität des Substrats	F
C_{sheet}	Kapazitätsbelag	$fF/\mu m^2$
$C_{substrat}$	Kapazitätsbelag des Substrats	$fF/\mu m^2$
c_{cap}(W)	Kosten der kapazitiven Einkopplung eines Pfades W	
c_{ij}	Platzierungskosten	
$c_{ij}^{(k)}$	Kosten des kürzesten Pfades von v_i nach v_j mit k inneren Knoten	
c_L(W)	Kosten der Weglänge eines Pfades W	
Dir_i	Kompaktierungsrichtung	
E_{conn}	Kantenmenge des Verbindungsgraphen	

E_K	Menge der Layoutkanten des kompaktierten Objekts	
E_Z	Menge der Layoutkanten des Zielobjekts	
e, e_1, e_V, e_H	Kante	
err_K	relativer Fehler	
f	Frequenz	Hz
f_{res}	Eigenresonanzfrequenz	Hz
g_i, g_{ds}, g_{dsi}	Ausgangsleitwert MOS Transistor	Ω^{-1}
g_m, g_{mi}, G_{mi}	Steilheit MOS Transistor	Ω^{-1}
G_{conn}	Verbindungsgraph	
GBW	Transitfrequenz	Hz
GMD	mittlerer geometrischer Abstand	m
I_{max}	größter zu erwartender Strom	A
$J_{max}(T_{ref})$	maximal zulässige Stromdichte for eine bestimmte Leitungstemperatur T_{ref}	A
K	Bezugspunkt der zu kompaktierenden Zelle	
K_d	definiertes Kapazitätsverhältnis	
K_r	reales Kapazitätsverhältnis	
L_{cnt}, W_{cnt}	minimale Abmessungen eines Kontaktes	m
l, l_i, L, L_i	Länge	m
L_{min}	minimale Länge	m
l_{inner}	Innenweite der Induktivität	m
L(e)	Layername der Kante	
M^+,M^-,M^0,M	Induktivität	H
m_{Ai}, m_{Bi}	Modultopologien	
min_d	minimaler berechneter Abstand	m
minDist(Layer)	Mindestabstand für Elemente der Maskenebene Layer	m
ovlp	minimaler Überlappungswert zweier Elemente	m
R	Widerstand	Ω
$R_□$	Flächenwiderstand	Ω/□
R_S	parasitärer Leitungswiderstand	Ω
R_{si}	parasitäter Widerstand des Substrats	Ω
$R_{substrat}$	Widerstandsbelag des Substrats	Ω/□
S	Menge der Zustände	
s_j	Zustand	
t_W	Leitungsdicke	m
T_A	Terminal A	
t_{Layer}	Dicke der Leiterbahn Layer	m
t_{ox}	Dicke des Siliziumdioxids	m
$U = (u_{ij})$	Nutzenmatrix	
u_{ij}	Nutzwerte	
V_{conn}	Knotenmenge des Verbindungsgraphen	
V_i	Eingangsspannung	V
V_o	Ausgangsspannung	V
v_i	Wegpunkt	
W_{ij}	Pfad in Form einer Folge von Wegpunkten	
w, w_i, W, W_i	Weite	m
W_{min}	minimale Weite	m
W_{Path}	Breite einer Leitung	m
$W_{Path,ref}$	Leitungsbreite des Segments $Element_{ref}$ des Pfades Path	m
w_k	Netzgewicht der Verbindung k	
w_{wire}	Leitungsbreite	m
x_1, x_2, x_3	Länge	m

$X(p_i)$	x-Koordinate des Punktes p_i	
$X_l(e)$, $X_r(e)$	linke bzw. rechte x-Koordinate der Kante e	
$\bar{x}_j$	mittlere Schichtdicke des Widerstands	m
$\vec{x}_i$	Position der Zelle i	
x_{TA}	x-Koordinate des Terminals A	
$Y(p_i)$	y-Koordinate des Punkte p_i	
$Y_l(e)$, $Y_r(e)$	linke bzw. rechte y-Koordinate der Kante e	
y_{TA}	y-Koordinate des Terminals A	
Z	Bezugspunkt der Zielzelle	

Literatur

[Ackl83] B. Ackland, N. Weste; *An automatic assembly tool for virtual grid symbolic layout*; VLSI ,83, S. 457-466, 1983.

[Adle01] T. Adler; *Algorithmen zur optimierten Verdrahtung integrierter Analogschaltungen*; Dissertation, Fortschrittsberichte VDI, Reihe 20, Nr. 336, 2001.

[Ahuj83] B.K. Ahuja; *An Improved Frequency Compensation Technique for CMOS Operational Amplifiers*; IEEE Journal Solid-State Circuits, Vol. 18, No. 6, S. 629-633, Dec. 1983.

[Aker70] S.B. Akers, J.M. Geyer, D.L. Roberts; *IC mask layout with a single conducting layer*; Proc. of the 7th Design Automation Workshop, S. 7-16, Juni 1970.

[Alle87] P.E. Allen und D.R. Holberg; *CMOS Analog Circuit Design*; ed. Bg. Holt, Rinehart and Winston, 1987.

[Alls82] D.J. Allstot; *A precision variable-supply CMOS comparator*; IEEE Journal of Solid-State Circuits, Vol. SC-17, S. 1080-1087, December 1982.

[Anzi93] F.S. Al-Anzi, S. Kim; *Automata Algorithms for Homotopic Compaction and Expansion*; Technical Report No. 93-17, Computer Science Department, Rensselaer Polytechnic Institute, 1993.

[Anzi03] F.S. Al-Anzi; *Efficient Cellular Automata Algorithms for Planar Graph and VLSI Layout Homotopic Compaction*; International Journal of Computing and Information Sciences, Vol. 1, No. 1, S. 1-17, Dez. 2003.

[Ashb94] K.B. Ashby, W.C. Finley, J.J. Bastek, S. Moinian und I.A. Koullias; *High Q inductors for wireless applications in a complementary silicon bipolar process*; Proc. Bipolar and BiCMOS Circuits and Technology Meeting, S. 179-182, Minneapolis, 1994.

[Bair75] H.S. Baird, Y.E. Cho; *An Artwork Design Verifikation System*; Proc. of the 12th ACM/IEEE Design Automation Conference, S. 414-420, Juni 1975.

[Bair78] H.S. Baird; *Fast Algorithms for LSI Artwork Analysis*; Journal of Design Automation and Fault-Tolerant Computing, S. 179-209, 1978.

[Bala99] F. Bala, K. Lampaert; *Module Placement for Analog Layout Using the Sequence-Pair Representation*; Proc. 36th ACM/IEEE Design Automation Conference DAC, S. 274-279, 1999.

[Bala00] F. Balasa, K. Lampaert; *Symmetry with sequence-pair representation in the context of placement for analog design*; IEEE Trans. on Computer-Aided Design, Vol. 19, S. 721-731, Juli 2000.

[Bark85] E. Barke; *Resistance Calculation from Mask Artwork Data by Finite Element Method*; Proc. of the 22nd Design Automation Conference, S. 305311, 1985.

[Bark88] E. Barke; *Line-to-Ground Capacitance Calculation for VLSI: A Comparison*; IEEE Transactions on Computer-Aided Design, Vol. 7, No. 2, S. 293298, Feb. 1988.

[Bast96] J. Bastos, M. Steyaert, B. Graindourze, W. Sansen; *Matching of MOS Transistors with Different Layout Styles*; Proc. IEEE International Conference on Microelectronic Test Structures, Vol. 9, S. 17-18, März 1996.

[Bell58] R. Bellman; *On a Routing Problem*; Quarterly of Applied Mathematics 16, S. 87 -90, 1958.

[Bent80] J.L. Bentley, D. Haken, R.W: Hon; *Fast Geometric Algorithms for VLSI Tasks*; Proc. of IEEE Computer Conference, S. 88-92, 1980.

[Boot92] H.D. Booth; *An Overview over Red-Black and Finger Trees*; Disseration, Department of Computer Science, University Tenessee, USA, 1992.

[Boye83] D.G. Boyer, N. Weste; *Virtual Grid Compaction Using the Most Recent Layers Algorithm*; Proc. of ICCAD, S. 92-93, 1983.

[Boye88] D.G. Boyer; *Symbolic layout compaction review*; Proc. of the 25th ACM/ IEEE Design Automation Conference, S. 383-389, 1988.

[Brae07] D. Braess; *Finite Elemente*; Springer Verlag Berlin Heidelberg, 2007.

[Brei84] C.J. Breiman, L. Friedman, J.H. Ohlson, R.A. Stone; *Classification and Regression Trees*; Pacific Grove: Wadsworth International Group, 1984.

[Breu72] M.A. Breuer, ed.; *Design Automation of Digital Systems*; Prentice-Hall, Englewood Cliffs, New Jersey, 1972.

[Breu77] M.A. Breuer; *A Class of Min-Cut Placement Algorithms*; 14th ACM/IEEE Design Automation Conference, S. 284-290, 1977.

[Bron99] I.N. Bronstein, K.A. Semendjajew, G. Musiol, H. Mühlig; *Taschenbuch der Mathematik*; Verlag Harri Deutsch, 4. überarb. und erw. Auflage, 1999.

[Bruc96] J.D. Bruce, H.W. Li, M.J. Dallabetta und R.J. Baker; *Analog Layout Using ALAS*; IEEE Journal of Solid-State Circuits, Vol. 31, No. 2,S. 271-274, Feb 1996.

[Brüc93] R. Brück; *Entwurfswerkzeuge für VLSI-Layout*; Hanser Verlag, 1993.

[Bui89] T. Bui, C. Heigham, C. Jones und T. Leighton; *Improving the Performance of the Kerninghan-Lin and Simulated Annealing Graph Bisection Algorithms*; 26th ACM/IEEE Design Automation Conference, S. 755-778, 1989.

[Cade00] *Diva Reference*; Product Version 4.4.6, Cadence Design Systems, 2000.

[Chan91] H. Chan, P. Mazamder, K. Shahookar; *Macro-cell and module placement by genetic adaptive search with bitmap-represented chromosome*; Integration, the VLSI Journal, 12(1), S. 49-77, 1991.

[Caso87] A. Casotto, F. Romeo, A. Sangiovanni-Vincentelli; *A Parallel Simulated Annealing Algorithm for the Placement of Macro-Cells*; IEEE Trans. on Computer-Aided Design, Vol. CAD-6, No. 5, S. 838-847, September 1987.

[Char94] E. Charbon, E. Malavasi, D. Pandini, A. Sangiovanni-Vincentelli; *Simultaneous Placement and Module Optimization of Analog IC's*; 31st ACM/ IEEE Design Automation Conference DAC, S. 31-35, 1994.

[Chen96] Y.-K. Cheng, E. Rosenbaum und S.-M. Kang; ETS-A: *A New Electrothermal Simulator for CMOS VLSI Circuits*; Proc. The European Design & Test Conference, S. 560-570, März 1996.

[Chia90] C. Chiang, M. Sarrafzadeh und C.K. Wong; *Global routing based on Steiner min-max trees*; IEEE Transactions on Computer-Aided Design, Vol. 9, No. 12, S. 1318-1325, 1990.

[Cho77] Y.E. Cho, A.J. Korenjak, D.E. Stockton; FLOSS: *An approach to automated layout of High-Volume designs*; Proc. of the 14th ACM/IEEE Design Automation Conference, S.138-141, Juni 1977.

[Cho85] Y.E. Cho; *A Subjective review of Compaction*; Proc. of the 22nd ACM/ IEEE Design Automation Conference, S. 396-404, 1985.

[Chou01] Y.-C. Chou, Y.-L. Lin; *A Performance-Driven Standard-Cell Placer Based on a Modified Force-Directed Algorithm*; Proc. International Symposium on Physical Design, S. 24-29, 2001.

[Cohn91] J.M. Cohn, et al.; *KOAN/ANAGRAM II: New Tools for Device-Level Analog Placement and Routing*; IEEE J. Solid-State Circuits, Vol. 26, S. 330342, März 1991.

[Cohn94] J. Cohn, D. Garrod, R. Rutenbar, L. Carley; *Analog Device-Level Automation*; Kluwer Academic Publishers, 1994.

[Coho87] J. Cohoon, W.Paris; *Genetic Placement*; IEEE Trans. on Computer-Aided Design, Vol. CAD-6, S. 956-964, Nov. 1987.

[Coho03] J. Cohoon, J. Karro, J. Lienig; *Evolutionary algorithms for the physical design of VLSI circuits*; Advances in evolutionary computing: theory and applications, Natural Computing Science, S. 683-711, 2003.

[Conw92] J.D. Conway, G.G. Schrooten; *An Automatic Layout Generator for Analog Circuits*; European Design Automation Conference EDAC, S. 513-519, 1992.

[Corm89] T.H. Cormen, C.E. Leiserson, R.L. Rivest; *Introduction to Algorithms*; The MIT Press, 1989.

[Corm01] T.H. Cormen, C.E. Leiserson, R.L. Rivest und C. Stein; *Introduction to Algorithms*; Second Edition, MIT Press and McGraw.Hill, Abschnitt 24.3: Dijkstra‘s algorithm, S. 595-601, 2001.

[Croc87] W.H. Crocker, C.Y. Lo, R. Varadarahan; *MACS: a Module Assembly and Compaction System*; Proc. of the IEEE International Conference on Computer Design, S. 205.208, Okt. 1987.

[Darw80] C. Darwin; *Die Entstehung der Arten durch natürliche Zuchtwahl*; Reclam, Stuttgart, 1980.

[Dijk59] E.W. Dijkstra; *A Note on two Problems in Connexion with Graphs*; Numerische Mathematik 1, S. 269-271, 1959.

[Dunl78] A.E. Dunlop; *SLIP: Symbolic Layout of Integrated Circuits with Compaction*; Computer-Aided Design, S. 387-391, Nov. 1978.

[Dunl80] A.E. Dunlop; *SLIM - The translation of symbolic layouts into mask data*; Proc. of the 17th ACM/IEE Design Automation Conference, S. 595-602, Juni 1980.

[Eise98] H. Eisenmann und F.M. Johannes; *Generic Global Placement and Floorplanning*; Proc. Design Automation Conference, S. 269-274, ACM Press, 1998.

[Esbe92] H. Esbensen; *A genetic algorithm for macro cell placement*; Proc. of the conference on European Design Automation, S. 52-57, 1992.

[Esbe97] H. Ebsensen, E.S. Kuh; *A performance-driven IC/MCM placement algorithm featuring explicit design space exploration*; ACM Trans. on Design Automation of Electronic Systems, Vol. 2, Issue 1, S. 62-80, 1997.

[Esch95] R.G.H. Eschauzier und J.H. Huijsing; *Frequency Compensation Techniques for Low-Power Operational Amplifiers*; Kluwer Academic Publishers, 1995.

[Fang91] J. Fang, J.S.L. Wong, K. Zhang, P. Tang; *A New Fast Constraint Graph Generation Algorithm for VLSI Layout Compaction*; Proc. IEEE International Symposium on Circuits and Systems, S. 2858-2861, Okt. 1991.

[Fidu82] C.M. Fiduccia und R.M. Mattheyses; *A Linear-Time Heuristic for Improving Network Partitions*; Proc. 19th Design Automation Conference IEEE/ ACM, S. 175-181, 1982.

[Fish85] J.A. Fisher; *A High-Performance CMOS Power Amplifier*; IEEE J. Solid-State Circuits, Vol. SC-20, S. 1200-1205, Dez. 1985.

[Floy62] R.W. Floyd; *Algorithm 97: Shortest Path*; Communications of the ACM, 5, 6, 345, 1962.

[Fran94] J.E. Franca und Y. Tsividis; *Design of Analog-Digital VLSI Circuits for Telecommunications and Signal Processing*; Prentice-Hall, Februar 1994.

[Fred87] M.L. Fredman, R.E. Tarjan; *Fibonacci heaps and their uses in improved network optimization algorithms*; Journal of ACM 34, S. 596-615, 1987.

[Fred90] M.L. Fredman, D.E. Willard; *Trans-dichotomus Algorithms for Minimum Spanning Trees and Shortest Paths*; Proc. of the 31th Annual IEEE Conference on Foundations of Computer Science, S. 719-725, 1990.

[Frie77] J.H. Friedman; *A recursive partitioning decision rule for nonparametric classification*; IEEE Transactions on Computers, C-26, S. 404-408, 1977.

[Gao89] S. Gao, M. Kaufmann, F.M. Maley; *Advances in Homotopic Layout Compaction*; ACM Symposium on Parallel Algorithms and Architecture, S. 273-282, 1989.

[Garr88] D.J. Garrod, R.A. Rutenbar und L.R. Carley; *Automatic Layout of Custom Analog Cells in ANAGRAM*; International Conference on Computer-Aided Design, S. 544-547, 1988.

[Garr91] D.J. Garrod; *Device-Level Routing for Analog Cells in ANAGRAM II*; Dissertation, Dept. of ECE, Carnegie Mellon University, August 1991.

[Giel95] G. Gielen, et. al.; *An Analog Module Generator for Mixed Analog/Digital ASIC Design*; John Wiley International Journal of Circuit Theory and Applications, Vol. 23, S. 269-283, Juli-Aug. 1995.

[Gime09] S.P. Gimenez; The Wave SOI MOSFET: *A New Accuracy Transistor Layout to Improve Drain Current and Reduce Die Area for Current Drivers Applications*; 215th ECS Meeting, Silicon-on-Insulator Technology and Devices 14, ECS Transactions, ECS, Vol. 19, S. 153-158, 2009.

[Glov85] F. Glover, D.D. Klingman, N.V. Phillips; *A New Polynomially Bounded Shortest Path Algorithm*; Operations Research 33(1), S. 65-73, 1985.

[Grah72] R.L. Graham; *An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set*; Information Processing Letters 1, S. 132-133, 1972.

[Gray80] P.R. Gray; *Basic MOS Operational Amplifier Design - An Overview*; Analog MOS Integrated Circuit, S. 28-49, 1980.

[Gray82] P.R. Gray und R.G. Meyer; *MOS operational amplifier design - A tutorial overview*; IEEE Journal Solid-State Circuits, Vol. SC-17, S. 969-982, Dez. 1982.

[Gray93] P.R. Gray und R.G. Meyer; *Analysis and Design of Analog Integrated Circutis*; John Wiley & Sons, Inc. New York, Chichester, Bisbaine, Toronto, Singapore, 1993.

[Gree74] H.M. Greenhouse; *Design of planar rectangular microelectronic inductors*; IEEE Transactions on Parts, Hybrids and Packing, Vol. PHP-10, No. 2, S. 101-109, Juni 1974.

[Grei98] G. Greiner, K. Hormann; *Efficient clipping of arbitrary polygons*; ACM Transactions on Graphics, 17(2), S. 71-83, 1998.

[Grov62] F.W. Grover; *Inductance Calculations*; Princeton New Jersey: Van Nostrand, 1946; Reprinted by New York, New York: Dover Publications, 1962.

[Hadl77] F.O. Hadlock; *A shortest path algorithm for grid graphs*; Netwroks, Vol. 7, No. 4, S. 323-334, 1977.

[Hana72] M. Hanan und J.M. Kurtzberg; *Placement techniques*; in: M.A. Breuer (Hrsg.): Design Automation of Digital Systems, Prentice Hall, S. 213-282, 1972.

[Hart68] P.E. Hart, N.J. Nilsson, B. Raphael; *A Formal Basis for the Heuristic Determination of Minimum Cost Paths*; IEEE Transactions on System Science and Cybernetics SSC4 4(2), S. 100-107, 1968.

[Hash71] A. Hashimoto, S. Stevens; *Wire routing by optimizing channel assignment with large apertures*; Proc. 8th Design Automation Conference, S.155-169, 1971.

[Hein04] H. Hein; *Untersuchungen zur vollständigen Integrierbarkeit von Empfängern in Standard CMOS Technologie*; Disseration, Friedrich Alexander Universität Erlangen-Nürnberg, 2004.

[Hest52] M.R. Hestens, E. Stiefel; *Methods of Conjugate Gradients for Solving Linear Systems*; J. Res. Natl. Bur- Stand. 49, S. 409-436, 1952.

[High69] D.W. Hightower; *A solution to line-routing problems on the continuous plane*; Proc. of the 6th annual conference on Design Automation, S.1-24, 1969.

[Hoer65] C. Hoer und C. Love; *Exact inductance equations for rectangular conductors with applications to more complicated geometries*; Journal of Research of the National Bureau of Standards, Vol. 69C, No. 2, S. 127-137, 1965.

[Horo83] M. Horowitz und R.W. Dutton; *Resistance Extraction from Mask Layout Data*; IEEE Transactions on CAD, Vol. CAD-2, S. 145-150, Juli 1983.

[Hsue79] M.Y. Hsueh; Symbolic *Layout and Compaction of Integrated Circuits*; U.C. Berkeley, UCB/ERL Report, M79/80, 1979.

[Hu02] B. Hu und M. Marek-Sadowska; *FAR: Fixed-points addtion & relaxation based placement*; Proc. of ISPD, S. 161-166, ACM Press, 2002.

[Hwan91] J.P. Hwang; *REX - A VLSI Parasitic Extraction Tool for Electromigration and Signal Analysis*; Proc. of the 28th Design Automation Conference, S. 717-722, 1991.

[Jarv73] R.A. Jarvis; *On the Identification of the Convex Hull of a Finite Set of Points in the Plane*; Information Processing Letters 2, S. 18-22, 1973.

[Jeps84] D.W. Jepsen, C.D. Gellat Jr.; *Macro placement by Monte Carlo Annealing*; Proc. IEEE International Conference on Computer Design, S. 495-498, November 1984.

[John97] D. Johns und K. Martin; *Analog Integrated Circuit Design*; John Wiley & Sons, 1997.

[Kast00] R. Kastner, E. Bozorgzadeh und M. Sarrafzadeh; *Predictable Routing*; Proc. International Conference on Computer-Aided Design, S. 110-113, 2000.

[Kaya88] M. Kayal, S. Piguet, M. Declercq und B. Hochet; *SALIM: A Layout Generation Tool For Analog ICs*; IEEE Custom Integrated Circuits Conference, S. 7.5.1-7.5.4, 1988.

[Kede84] G. Kedem, H. Watanabe; *Graph optimization techniques for IC layout and compaction*; IEEE Transactions on Computer-Aided Design, Vol. CAD-3, No.1, Jan. 1984.

[Kern70] B. Kerninghan und S. Lin; *An efficient heuristic procedure for partitioning graphs*; Bell System Technical Journal 29, S. 291-307, 1970.

[King84] C. Kingsley; *A Hierarchical, Error-Tolerant Compactor*; Proc. of the 21st ACM/IEEE Design Automation Conference, S. 126-132, Juni 1984.

[Klei88] J.M. Kleinhans, G. Sigl, F.M. Johannes; *GORDIAN: A New Global Optimization/Rectangle Dissection Method for Cell Placement*; Proc. International Conference on Computer-Aided Design, S. 506-509, 1988.

[Knut98] D. Knuth; *The Art of Computer Programming*; Addison-Wesley Verlag, 1998.

[Kong02] T. Kong; *A Novel Net Weighting Algorithm for Timing-Driven Placement*; Proc. International Conference on Computer-Aided Design, S. 172-176, 2002.

[Korn82] R.K. Korn; *An Efficient Variable-Cost Maze Router*; IEEE 19th Design Automation Conference, S. 425-431, 1982.

[Koud06] S. Kouda, C. Kodama, K. Fujiyoshi; *Improved method of cell placement with symmetry constraints for analog IC layout design*; Proc. International Symposium on Physical Design, S. 192-199, 2006.

[Koza84] T. Kozawa, C. Miura, H. Terai; *Combine and Top Down Block Placement Algorithm for Hierarchical Logic VLSI Layout*; 21th ACM/IEEE Design Automation Conference DAC, S. 667-669, 1984.

[Kram82] M.R. Kramer und J. van Leeuwen; *Wire-routing is NP-complete*; Technical Report, Computer Science Dept., University of Utrecht, Netherlands, 1982.

[Lada93] L. Ladage und R. Leupers; *Resistance Extraction using a Routing Algorithm*; DAC'93: Proc of the 30th Design Automation Conference, S. 38-42, 1993.

[Lake94] K. R. Laker und W.M.C. Sansen; *Design of Analog Integrated Circuits and Systems*; McGraw-Hill Inc., 1994.

[Lamp95] K. Lampaert, G. Gielen und W. Sansen; *A performance-driven placment tool for analog integrated circuits*; IEEE Journal of Solid-State Circuits, Vol. SC-30, No. 3, S. 327-330, März 1995.

[Lamp96a] K. Lampaert, G. Gielen und W. Sansen; *Analog routing for manufacturablility*; CICC: Proc. Custom Integrated Circuits Conference, S. 175-178, Mai 1996.

[Lamp96b] K. Lampaert, G. Gielen und W. Sansen; *Thermally Constrained Placement of Analog and Smart Power Integrated Circuits*; ESSCIRC'96: Proc. of the 22nd European Solid-State Circuits Conference, S. 160-163, Sep. 1996.

[Lang88] H.W. Lang; *Transitive Closure on the Instruction Systolic Array*; Proc. of the International Conference on Systolic Arrays, S. 295-304, San Diego, 1988.

[Laur87] P.J.M. van Laurhoven und E.H.L. Aarts; *Simulated Annealing: Theory and Applications*; D. Riedel, Dortrecht, 1987.

[Laut87] U. Lauther; *An Overview of Placement and Routing Techniques*; Proc. COMPEURO 87, S. 615-620, 1987.

[Lawl76] E.L. Lawler; *Combinatorial Optimization: Networks and Matroids*; Holt, Rinehart & Winston, New York, 1976.

[Lee61] C. Lee; *An Algorithm for Path Connections and its Applications*; IRE Trans. on Electronic Computers, VEC-10, S. 346-365, 1961.

[Lee93] S.S. Lee und D.J. Allstot; *Electrothermal Simulation of Integrated Circuits*; IEEE J. Solid-State Circuits, S. 1283-1293, Dez. 1993.

[Leng90] T. Lengauer; *Combinatorial Algorithms for Integrated Circuit Layout*; B.G. Teubner Verlag, Stuttgart, John Wiley and Sons Inc., Chichester, New York, 1990.

[Liao83] Y. Liao, C.K. Wong, *An Algorithm to Compact VLSI Symbolic Layout with Mixed Constraints*; IEEE Trans. on Computer-Aided Design if Circuits and Systems, S. 62-69, April 1983.

[Lien03] J. Lienig, G. Jerke; *Elektromigration - Eine neue Herausforderung beim Entwurf elektronischer Baugruppen, Teil 2: Stromabhängige Verdrahtung von Leiterbahnen*; F & M Feinwerktechnik, Mikrotechnik, Mikroelektronik, S. 26-28, Januar/Februar 2003.

[Lien05] J. Lienig und G. Jerke; *Electromigration - Aware Physical Design of Integrated Circuits*; VLSID‘05: Proc. Conference On Embedded Systems Design, S. 77-82, Jan. 2005.

[Lien06] J. Lienig; *Invited Talk: Introduction to Electromigration-Aware Physical Design*; ISPD ´06, S. 39-46, April 2006.

[Lipk06] B. Lipka, J. Lindroos und U. Kleine; *Design of an Analog Power OpAmp with ±1.5V and 45° Structures to Reduce the Effect of Electromigration*; Kleinheubacher Tagung 2006, Miltenberg, Sep. 2006.

[Lipk07] B. Lipka und U. Kleine; *Design of a Linear Power Amplifier with ±1.5V Power Supply Using ALADIN*; 17th International Workshop, PATMOS 2007, S. 97-106, Sep. 2007.

[Lips83] W. Lipski; *Finding a Manhattan Path and Related Problems*; Networks, John Wiley and Sons Inc., Chichester, New York, Vol. 13, S. 399-409, 1983.

[Liu03] Y.K. Liu, Y. Gao, Y.Q. Huang; *An efficient algorithm for polygon clipping*; Journal of Software, 14(4), S. 845-856, 2003.

[Long97] J.R. Long und M.A. Copeland; *The Modeling, Characterization and Design of Monolithic Inductors for Silicon RF ICs*; IEEE Journal Solid-State Circuits, Vol. 32, No. 3, S. 357-369, März 1997.

[Lou97] J. Lou, A.H. Salek, M. Pedram; *An Exact Solution to Simultaneous Technology Mapping and Linear Placement Problem*; Proc. International Conference on Computer-Aided Design, S. 671-675, 1997.

[Lovr91] T. Lovric; Platzierung bei der Layoutsynthese für analoge Schaltungen; Diplomarbeit Universität Dortmund, 1991.

[Luk85] W.K. Luk; *A Greedy Switchbox Router*; Integration, The VLSI Journal, Vol. 3, S. 129-149, 1985.

[Mala90] E. Malavasi, et. al.; *A Routing Methodology For Analog Integrated Circuits*; Proc. IEEE ICCAD, S.202-205, Santa Clara, November 1990.

[Mala91] E. Malavasi, E. Charbon, G. Jusuf, A. Sangiovanni-Vincentelli; *Virtual symmetry axes for the layout of analog IC‘s*; Proc. 2nd ICVC, S. 195-198, Oktober 1991.

[Mala96] E. Malavasi, E. Charbon, E. Felt, A. Sangiovanni-Vincentelli; *Automation of IC layout with analog constraints*; IEEE Trans. on Computer-Aided Design of IC‘s and Systems, Vol. 15, No. 8, S. 923-942, August 1996.

[Mali00] S.Q. Malik und R.L. Geiger; *Minimization of area in low-resistance MOS switches*; Proc. of 2000 Midwest Symposium on Circuits and Systems, S. 1392-1395, Aug. 2000.

[Malo01] F. Maloberti; *Analog Design for CMOS VLSI Systems*; Kluwer Academic Publishers, Boston 2001.

[Malo04] C. Malonnek; *Ein leitbahnorientiertes Verfahren für den physikalischen Entwurf integrierter digitaler Schaltungen*; Dissertation, Universität Hannover, 2004.

[Marw93] P. Marwedel; *Synthese und Simulation von VLSI Systemen*; Carl-Hanser Verlag, München, 1993.

[McCr81] J.L. McCreary; *Matching properties, Voltage and Temperature Dependence of MOS Capacitors*; IEEE Journal of Solid-State Circuits, Vol. SC16, S. 608-616, Dezember 1981.

[Mehr91] S.W. Mehranfar; *A Technology-Independent Approach to Custom Analog Cell Generation*; IEEE Journal of Solid-State Circuits, Vol. 26, No. 3, S. 386-393, März 1991.

[Meij95] N.P. van der Meijs, A.J. van Genderen; *Delayed Frontal Solution for Finite-Element based Resistance Extraction*; Proc. 32nd Design Automation Conference, S. 273-278, 1995.

[Meis99] A. Meister; *Numerik linearer Gleichungssysteme*; Vieweg Verlag, 1999.

[Meye93] V. Meyer zu Bexten, C Moraga, R. Klinke und W. Brockherde; *ALSYN: Flexible Rule-Based Layout Synthesis for Analog ICs*; IEEE Custom Integrated Circuits Conference, pp. 11.6.1-11.6.4, 1993.

[Meye94] V. Meyer zu Bexten; *User-Controlled Layout Synthesis for Analog Integrated Circuits*; Dissertation Universität Dortmund, Shaker Verlag, 1994.

[Mich94] Z. Michalewitz; *Genetic Algorithms + Data Structures = Evolution Programs*; 2nd Edition Springer Verlag, 1994.

[Mika68] K. Mikami, K. Tabuchi; *A Computer Program for Optimal Routing of Printed Circuit Conductors*; Proc. of the IFIP Congress, Vol. 2, S. 14751478, 1968.

[Mo01] F. Mo, A. Tabbara und R.K. Brayton; *A Force-Directed Mace Router*; Proc. International Conference on Computer-Aided Design, S. 404-407, 2001.

[Moor59] E.F. Moore; *Shortest Path Through a Maze*; Annals of the Harvard Computation Laboratory, Vol. 30, part II, Cambridge, Mass., Harvard University Press, S. 285-292, 1959.

[Mura96] H. Murata, K. Fujiyoshi, S. Nakatake, Y. Kajitani; *VLSI module placement based on rectangle-packing by the sequence-pair*; IEEE Trans. on Computer-Aided Design of IC's and Systems, Vol. 15, No. 12, S. 1518-1524, Dezember 1996.

[Naha86] S. Nahar, S. Sahni und E. Shragowitz; *Simulated Annealing and Combinatorial Optimization*; 23rd ACM/IEEE Design Automation Conference, S. 293-299, 1986.

[Onod90] H. Onodera, V. Taniguchi, K. Tamaru; *Operational-Amplifier Compilation with Performance Optimization*; IEEE Journal of Solid-State Circuits, Vol. 25, No. 2, S. 466-473, 1990.

[Onod92] H. Onodera, K. Tamaru; *Analog Circuit Placement - Branch-and-Bound Placement with Shape Optimization*; IEEE Custom Integrated Circuits Conference, S. 11.5.1-11.5.6, 1992.

[Oust84] J. Ousterhout; *Corner Stitching: A Data-Structuring technique for VLSI Layout Tools*; IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, CAD-3(1), S. 87-100, 1984.

[Otte80] R. Otten; *Complexity and diversity in IC layout design*; Proc. IEEE International Symposium Circuits and Computers, 1980.

[Ozak80] T. Ozaki, J. Yoshida, M. Kosaka; *PANAMAP-1: A mask pattern analysis program for IC/LSI*; Proc. Int. Symposium Circuits and Systems, S. 1020-1026, 1980.

[Palo03] L. Palo; *Elektronik für Ingenieure: Analoge und digitale integrierte Schaltungen*; Vieweg Verlag, 2003.

[Pete03] M. Peter; *Optimierte induktive Bauelemente in monolithisch integrierten Schaltungen für Radiofrequenzen*; Dissertation, Friedrich Alexander Universität Erlangen-Nürnberg, 2003.

[Plat06] R. Plato; *Numerische Mathematik kompakt*; Vieweg Verlag, 2006.

[Prep85] F.P. Preparata, M.I. Shamos; *Computational Geometry - An Introduction*; Springer Verlag, 1985.

[Puwa95] V. Puwada, S. Potla, T. Selvam und P. R. Suresh; *A simulation study on the effectiveness of n-guardring/p-guardring on latchup in 0.8 µm CMOS technology*; Proc. of the 8th Internaltional Conference on VLSI Design, S. 192, Januar 1995.

[Quin86] J.R. Quinlan; *Induction of decision trees*; Machine Learning, 1, S. 81-106, 1986.

[Quin93] J.R. Quinlan; *C4.5: Programs for Machine Learning*; Morgan Kaufmann Publishers, 1993.

[Ramo65] S. Ramo, J. Whinnery, T. Van Duzer; *Fields and Waves in Communication Electronics*; New York Wiley, 1965.

[Raza92] B. Razavi und B.A. Wooley; *Design Techniques for High-Speed, High Resolution Comparators*; IEEE Journal of Solid-State Circuits, Vol. 27, No. 12, S. 1916-1926, Dezember 1992.

[Renc03] M. Rencz, V. Szekely und A. Poppe; *A fast algorithm for the layout based electro-thermal simulation*; DATE'03: Proc. of the conference on Design, Automation and Test in Europe, S. 11032, 2003.

[Ribn84] D.B. Ribner und M.A. Copeland; *Design Techniques for Cascoded CMOS Op Amps with Improved PSRR and Common-Mode Input Range*; IEEE Journal Solid-State Circuits, Vol. SC-19, S. 919-925, Dez. 1984.

[Rijm89] J. Rijmenants, J.B. Litsois, T.R. Schwarz und M.G.R. Degrauwe; *ILAC: An Automated Layout Tool for Analog CMOS Circuits*; IEEE Journal of Solid-State Circuits, Vol. SC-24, No. 2, pp. 417-425, 1989.

[Rive82] R.L. Riveest, C. Fiducia; *A greedy channel router*; Proc. of the 19th Design Automation Conference, S.418-424, 1982.

[Roew02] F. Roewer und U. Kleine; *A Novel Class of Complementary Folded-Cascoded Opamps for Low Voltage*; IEEE Journal Solid-State Circuits, Vol. 37, No. 8, S. 1080-1083, Aug. 2002.

[Rose96] M. Rose; *Kritische Analyse von Routensuchverfahren*; Institut für Verkehrswirtschaft, Straßenwesen und Städtebau, Universität Hannover, 1996.

[Sadi93] S.M. Sadiq, H. Youssef, S. Tanvir, M.S.T. Benten; *Timinig influenced general-cell floorplaner*; Proc. of the Asia and South Pacific Design Automation Conference, S.135-140, 1995.

[Sale98] A.H. Salek, J. Lou, M. Pedram; *A DSM Design Flow: Putting Floorplanning, Technology-Mapping and Gate-Placement Together*; Proc. International Conference on Computer-Aided Design, S. 128-133, 1998.

[Sarr92] M. Sarrafzadeh, C.K. Wong; *Hierarchical Steiner Tree Construction in Uniform Operations*; IEEE Trans. On Computer-Aided Design, Vol. 11, No. 9, S. 1095-1103, September 1992.

[Sarr97] M. Sarrafzadeh, M. Wang; *NRG: Global and Detailed Placement*; International Conference on Computer-Aided Design, S. 532-537, 1997.

[Sast82] S. Sastry, A. Parker; *The Complexity of Two-Dimensional Compaction of VLSI Layout*; Proc. of ICCC-82, S. 402-406, 1982.

[Saye02] D.E. Sayed und M. Dessouky; *Automatic Generation of Common-Centroid Capacitor Arrays with Arbitrary Capacitor Ratio*; DATE'02: Proc. of the conference on Design, Automation and Test in Europe, S. 576-580, 2002.

[Schi83] W.L. Schiele; *Improved Compaction by Minimized Length of Wires*; Proc. of the 20th ACM/IEEE Design Automation Conference, S. 121-127, Juni 1983.

[Schl83] M. Schlag, Y.-Z. Liao, C.K. Wong; *An algorithm for optimal two-dimensional compaction of VLSI layouts*; Integration, the VLSI Journal, Vol. 1, Issues 2-3, S. 179-209, 1983.

[Schr05] L. Schreiner, M. Olbrich, E. Barke und V. Meyer zu Bexten; *Routing of Analog Busses with Parasitic Symmetry*; ISPD'05, S. 14-19, San Francisco, 2005.

[Sech86] C. Sechen und A. Sangiovanni-Vincentelli; *TimberWolf3.2: A New Standard Cell Placement and Global Routing Package*; 23rd ACM/IEEE Design Automation Conference, S. 432-439, 1986.

[Sech88] C. Sechen; *Chip-Planning, Placement and Global Routing of Macro/Custom Cell Integrated Circuits Using Simulated Annealing*; 25th ACM/IEEE Design Automation Conference DAC, S. 73-80, 1988.

[Sedg86] R. Sedgewick, J. Vitter; *Shortest path in euclidean graphs*; Algorithmica 1, S. 31-48, 1986.

[Shah91] K. Shahookar, P. Mazumder; *VLSI cell placement techniques*; ACM computing surveys, Vol. 23, S. 143-221, 1991.

[Ship88] T. Shiple, P. Kollaritsch, J. Allen; *Area Evaluation Metrics for Transistor Placement*; IEEE International Conference on Computer Design ICCD, S. 428-433, 1988.

[Sigl91] G. Sigl, K. Doll, F.M. Johannes; *Analytical Placement: A Linear or a Quadratic Objective Function?*; Proc. Design Automation Conference, S. 427-431, 1991.

[Soin91] R.S. Soin, F. Maloberti und J. Franca; *Analogue-Digital ASICs: circuit techniques, design tools and applications*; Peter Peregrinus Ltd. London, 1991.

[Souk78] J. Soukup; *A fast maze router*; Proc. 15th Design Automation Conference, S. 100-101, 1978.

[Souk92] J. Soukup; *Maze router without a grid map*; Proc. of the 1992 IEEE/ACM International Conference on Computer-Aided Design, Santa Clara, S. 382-385, 1992.

[Spir73] P.M. Spira; *A new Algorithm for Finding all Shortest Paths in a Graph of Positive Arcs in Average Time O($n^2 \log^2 n$)*; SIAM Journal of Computing 2, S. 28-32, 1973.

[Spir90] H. Spiro; *Simulation integrierter Schaltungen*; Oldenbourg Verlag, 1990.

[Sten97] G. Stenz, B.M. Riess, B. Rohfleisch, F.M. Johannes; *Timinig Driven Placement in Interaction with Netlist Transformations*; International Symposium on Physical Design, S. 36-41, 1997.

[Su93] D.K. Su, M.J. Loinaz, S. Masui und B.A. Wooley; *Experimental Results and Modeling Techniques for Substrate Noise in Mixed-Signal Integrated Circuits*; IEEE Journal of Solid-State Circuits, Vol. SC-28, No. 4, S. 420430, April 1993.

[Sun95] W.-J. Sun, C. Sechen; *Efficient and effective placement for very large circuits*; IEEE Trans. on Computer-Aided Design of IC's and Systems, Vol. 14, No. 3 S. 349-359, März 1995.

[Sun97] W.Sun, W. Hong, W. Dai; *Resistance Extraction using Superconvergence Accelerated Boundary Element Method*; Proc. Asia Pacific Microwave Conference, S. 1061-1064, 1997.

[Swar95] W. Swartz, C. Sechen; *Timing Driven Placement for Large Standard Cell Circuits*; Proc. of the 32nd ACM/IEEE Design Automation Conference DAC, S. 211-215, 1995.

[Sysl83] M.M. Syslo, N. Deo, J.S. Kowalik; *Discrete Optimization Algorithms with PASCAL programs*; Prentice Hall, Englewood Cliffs, N.J., 1983.

[Szym83] T.G. Szymanski, C.J. Van Wyk; *Space Efficient Algorithms for VLSI Artwork Analysis*; Proc. of the 20th ACM/IEEE Design Automation Conference, S. 734-739, Juni 1983.

[Tam06] Y. Tam, E. Young, C. Chu; *Analog placement with symmetry and other placement constraints*; Proc. ACM/IEEE International Conference on Computer-Aided Design, S. 349-354, 2006.

[Tsai98] T. Li, C.-H. Tsai und S.-M. Steve; *Efficent Transient Electrothermal Simualtion of CMOS VLSI Ciruits under Electrical Overstress*; ICCAD'98: Proc. of the 1998 IEEE/ACM International conference on Computer-aided design, 1998.

[Tsai00] C.-H. Tsai und S.-M. Kang; *Fast Temperature Calculation for Transient Electrothermal Simulation by Mixed Frequency/Time Domain Thermal Model Reduction*; DAC'00: Proc. On Design Automation, S. 750-755, Juni 2000.

[Tsiv87] Y.P. Tsividis; *Operation and modeling of the MOS transistor*; McGraw-Hill, 1987.

[Ullm84] J.D. Ullman; *Computational Aspects of VLSI*; Computer Science Press, 1984.

[Vahi97] F. Vahid und T.D. Lee; *Extending the Kerninghan/Lee Heuristic for Hardware and Software Functional Partitioning*; Design Automation for Embedded Systems 2, S. 237-261, 1997.

[Vall94] R.E. Vallee und E.I. El-Masry; *A very high-frequency CMOS complementary folded-cascode amplifier*; IEEE Journal Solid-State Circuits, Vol. 29, S. 130-133, Februar 1994.

[Verg93] N.K. Verghese, S.-S. Lee und D.J. Allstot; *A Unified Approach to Simulating Electrical and Thermal Substrate Coupling Interactions in ICs*; ICCAD'93: Proc. of the 1993 IEEE/ACM international conference on Computer-aided design, S. 422-426, 1993.

[Vorw04] K. Vorwerk, A Kennings und A. Vannelli; *Engineering Details of a Stable Force-Directed Placer*; Proc. of the 2004 IEEE/ACM Int. Conference on Computer-aided design, S. 573-580, 2004.

[Wald50] A. Wald; *Statistical decision functions*; New York Wiley, 1950.

[Wang00] M. Wang, X. Yang, M. Sarrafzadeh; *DRAGON2000: Standard-Cell Placement Tool for Large Industry Circuits*; International Conference on Computer-Aided Design, S. 260-263, 2000.

[Wars62] S. Warshall; *A Theorem on Boolean Matrices*; Journal of the ACM, Vol. 9 S. 11-12, 1962.

[West81a] N. Weste; *Virtual Grid Symbolic Layout*; Proc. of the 18th ACM/IEEE Design Automation Conference, S. 225-233, Juni 1981.

[West81b] N.H.E. Weste; *MULGA - An interactive symbolic system for the design of integrated circuits*; The Bell System Technical Journal, S. 823-857, 1981.

[Will99] J. Willemen, G. Digele; *Elektrothermische Schaltungssimulation mit Saber unter Einbeziehung des Gehäuseeinflusses*; Analog‘99, 5. ITG/GMM-Diskussionssitzung, S. 87-94, 1999.

[Wolf83] W. Wolf, R. Mathews, J. Newkirk, R. Dutton; *Two-Dimensional Compaction Strategies*; Proc. of ICCAD, S. 90-91, 1983.

[Wolf84] W.H. Wolf; *Two-Dimensional compaction strategies*; Dissertation, Standford University, 1984.

[Wolf85] W. Wolf; *An experimental comparison of 1-D compaction algorithms*; Chapel Hill VLSI Conference, 1985.

[Wolf96] M. Wolf, U. Kleine und B. Hosticka; *A Novel Analog Module Generator Environment*; Proc. The European Design & Test Conference, S. 388-392, März 1996

[Wolf97] M. Wolf, U. Kleine; *Application Independent Module Generation in Analog Layouts*; Proc. European Design and Test Conference 1997, S. 624, März 1997.

[Wolf98a] M. Wolf und U. Kleine; *A Novel Design Assistant for Analog Circuits*; Proc. Asia and South Pacific Design Automation Conference, S. 495-500, Feb. 1998.

[Wolf98b] M. Wolf, U. Kleine und J. Schulze; *New Description Language and Graphical User Interface for Module Generation in Analog Layouts*; Proc. International Symposium of Circuit and Systems, Vol. VI, S. 290-293, Juni 1998.

[Wolf99a] M. Wolf und U. Kleine; *Reliability Driven Module Generation for Analog Layouts*; ISCAS 99, Vol. 6, S. 412-415, June 1999.

[Wolf99b] M. Wolf; *Konstruktive Layoutgenerierung mit automatischer Neugenerierung unter geänderten Randbedingungen*; Dissertation, Otto-von-Guericke Universität, Magdeburg, 1999.

[Wolf01] M. Wolf und L. Zhang; *Automatische Layoutgenerierung für zuverlässige analoge integrierte Schaltungen*; Proc. Fachtagung Informationstechnik, S. 23-28, März 2001.

[Wong86] D.F. Wong, C.L. Liu; *A new algorithm for floorplan design*; Proc. 23rd ACM/IEEE Design Automation Conference DAC, S. 101-107, 1986.

[Wong88] D.F. Wong, H.W. Leong und C.L. Liu; *Simulated Annealing for VLSI Design*; Kluwer Academic Publishers, 1988.

[Xion90] X. Xiong; *Geometric Approach to VLSI Layout Compaction*; International Journal of Circuit theory and Applications, Vol. 8, S. 411-430, 1990.

[Yosh79] H. Yoshimura, K. Tansho, N. Ohwada, T. Nishide; *An algorithm for resistance calculation from IC mask pattern information*; Proc. International Symposium Circuits and Systems, S. 478-481, 1979.

[Yue99] C.P. Yue und S.S. Wong; *Design Strategy of On-Chip Inductors for Highly Integrated RF Systems*; DAC‘99: Proc. of the 36th annual ACM/IEEE Design Automation Conference, S. 982-987, 1999.

[Zhan00] L. Zhang, U. Kleine, T. Rudolph und M. Wolf; *A New Design Rule Description for Automated Layout Tools*; Proc. ICECS, S. 988-992, Dec. 2000.

[Zhan01] L. Zhang, U. Kleine, M. Wolf; *Automatic Inner Wiring for Integrated Analog Modules*; Proc. 8th International Conference on Mixed Design of Integrated Circuits and Systems, S. 109-114, Juni 2001.

[Zhan02] X. Zhang, S. Lam, R.K. Ko, M. Chan; *High-speed mixed signal and RF circuit design with compact waffle MOSFET*; Proc. IEEE Hong Kong Electrom Devices Meeting, S. 103-106, 2002.

[Zhan03] L. Zhang; *Layout Synthesis of Analog Integrated Circuits*; Dissertation, Otto-von-Guericke Universität Magdeburg, 2003.

[Zhan04] Y. Zhan und S.S. Sapatnekar; *Optimization of Integrated Spiral Inductors Using Sequential Quadratic Programming*; DATE'04: Proc. of the conference on Design, automation and test in Europe, Februar 2004.

[Zhan06a] L. Zhang, U. Kleine und Y. Jiang; *An Automated Design Tool for Analog Layouts*; IEEE Trans. On VLSI, Vol. 14, No. 8, S. 881-894, Aug. 2006.

[Zhan06b] L. Zhang, R. Raut, Y. Jiang und U. Kleine; *Placement Algorithm in Analog-Layout Design*; IEEE Trans. On CAD, Vol. 25, S. 1889-1903, No. 10, Oktober 2006.

[Zhan08] Q. Zhang; *Design von CMOS-Operationsverstärkern und -Komparatoren*; Diplomarbeit, Otto-von-Guericke-Universität Magdeburg, Juni 2008.

[Zhan08a] L. Zhang, C.-J.R. Shi, Y. Jiang; *Symmetry-Aware Placement with Transitive Closure Graphs for Analog Layout Design*; Proc. of the 13th Asia South Pacific Design Automation Conference ASP-DAC, S. 180-185, Januar 2008.

[Zhan09] Y. Zhang; *Entwurf von Modulgeneratoren für passive integrierte Bauelemente mit Hilfe der überarbeiteten Bibliothek MOGLAN*; Diplomarbeit, Otto-von-Guericke-Universität Magdeburg, Juli 2009.

Abbildungsverzeichnis

Tabellenverzeichnis

Lebenslauf

Name:	Lipka
Vorname:	Björn
akadem.Grad:	Dipl.-Math.
Geburtsdatum:	02.07.1978
Geburtsort:	Hannover
Geschlecht:	männlich
Staatsangehörigkeit:	Deutsch
Wohnort:	Hasselhorster Straße 21
	31515 Wunstorf
	Schulbildung
07/1990 – 06/1997	Hölty-Gymnasium Wunstorf
	Hochschulstudium
10/1998 – 11/2004	Leibniz Universität Hannover
	Mathematik mit der Studienrichtung Informatik
	Abschluss: Diplom-Mathematiker
	Berufliche Erfahrungen
06/2005 – 07/2005	Software Developer bei der Hannover Rückversicherung AG
08/2005 – 07/2010	Wissenschaftlicher Assistent an der Otto-von-Guericke Universität Magdeburg
09/2010 – 09/2011	Software-Architekt bei der ATLAS Elektronik GmbH, Bremen
seit 10/2011	Software System Engineer bei der OHB System AG, Bremen