Ein leitbahnorientiertes Verfahren für den physikalischen Entwurf integrierter digitaler Schaltungen

Vom Fachbereich Informatik
der Universität Hannover
zur Erlangung des akademischen Grades
Doktor-Ingenieur
genehmigte Dissertation

von Dipl.-Ing. Carsten Malonnek

geboren am 06.02.1973 in Hannover

2004

1. Referent: Prof. Dr.-Ing. E. Barke

2. Referent: Prof. Dr.-Ing. C.-E. Liedtke

Tag der Promotion: 26.03.2004

Für Nicole

Vorwort

Die vorliegende Arbeit entstand während meiner Zeit als wissenschaftlicher Mitarbeiter am Institut für Mikroelektronische Systeme der Universität Hannover.

An dieser Stelle bedanke ich mich ganz herzlich bei Herrn Prof. Dr.-Ing. E. Barke für die Betreuung meiner Arbeit sowie für die stetige Unterstützung und die wertvollen Hinweise während der Durchführung meiner Arbeit. Herrn Prof. Dr.-Ing. C.-E. Liedtke danke ich für das Interesse, das er mit der Übernahme des Korreferats dieser Arbeit entgegen gebracht hat und Herrn Prof. Dr. R. Parchmann für die Übernahme des Vorsitzes der Prüfungskommission.

Weiterhin danke ich allen Mitarbeiterinnen und Mitarbeitern des Instituts für Mikroelektronische Systeme für das gute Arbeitsklima und die vielen konstruktiven Diskussionen sowie den Mitgliedern der Layoutgruppe für die viele Anregungen und Impulse, die zum Gelingen dieser Arbeit beigetragen haben.

Mein ganz besonderer Dank gilt meiner Freundin Nicole, die mich bei der Durchführung der Arbeit stets unterstützt und motiviert hat. Mit ihrer unendlich entgegengebrachte Liebe und ihrem Verständnis war sie jederzeit für mich da.

Bad Münder, im März 2004

Carsten Malonnek

Inhaltsverzeichnis

Kurzfassung

Über viele Jahre und Jahrzehnte hinweg hat die minimale Strukturgröße integrierter Schaltungen kontinuierlich abgenommen, während gleichzeitig die maximale Taktfrequenz angestiegen ist. Durch die Verkleinerung der minimalen Strukturgrößen sinken die Verzögerungszeiten der Gatter, während die Verzögerungszeiten des Verbindungsnetzwerks pro Länge ansteigen. Die Modellierung der Leitungsverzögerungen erfolgte bisher bei vielen EDA-Werkzeugen nur mit statistischen Modellen. Für aktuelle Technologien, bei denen die Leitungsverzögerungen in der Größenordnung der Gatterverzögerungszeiten liegen, führen diese Modelle zu einem nicht mehr akzeptablen Fehler bei der Abschätzung von Pfad-Verzögerungszeiten.

In dieser Arbeit werden ein neues Platzierungsverfahren sowie ein neuartiges Verdrahtungsverfahren vorgestellt, die die Leitungsverzögerungen bereits in frühen Entwurfsphasen mit einer hohen Genauigkeit berücksichtigen. Das Platzierungsverfahren basiert auf kräftegesteuerter Platzierung und erweitert das bekannte Verfahren um zwei neue Aspekte: Zum einen kommt in jedem Platzierungsschritt eine Pfadkraft zum Einsatz, die die Differenz zwischen Verzögerungszeiten und Pfad-Constraints modelliert und zur iterativen Verbesserung der Platzierung verwendet wird. Zum anderen erfolgt die Globalverdrahtung gleichzeitig mit der Platzierung. Dadurch ist bereits während der Platzierung eine gute Modellierung der Detailverdrahtung möglich.

Das neue Detailverdrahtungsverfahren basiert auf einer Leitungselemente-Bibliothek. In dieser Bibliothek werden charakterisierte Leitungselemente mit unterschiedlicher Form und unterschiedlicher Anzahl von Vias gespeichert. Die Bibliothek enthält für jedes Element Informationen über seine Kapazität und seinen Widerstand. Die Detailverdrahtung erfolgt durch die Selektion und Platzierung der Bibliothekselemente. Während die Initialverdrahtung zufällig durchgeführt wird, erfolgt anschließend die Verbesserung der Verdrahtung mit einem auf Simulated-Annealing basierenden Optimierungsverfahren.

Die Ergebnisse zeigen, dass mit dem im Rahmen dieser Arbeit entwickelten Designflow eine Layouterstellung mit erheblichen Vorteilen gegenüber einem konventionellen Designflow möglich ist. Als Beispiele werden zur Vergleichbarkeit mit anderen Layoutwerkzeugen einige Schaltungen der MCNC-Benchmarksuite verwendet. Dabei hat sich gezeigt, dass die Länge der kritischen Pfade abnimmt, während gleichzeitig die Länge der unkritischen Pfade ansteigt. Durch die bibliotheksbasierte Verdrahtung wird die Abweichung zwischen dem nach der Platzierung geschätzten Laufzeitverhalten der Schaltung und dem tatsächlich erreichten Laufzeitverhalten nach der Verdrahtung kleiner. Dadurch ist bereits nach der Platzierung eine Vorhersage möglich, ob eine Verdrahtung ohne Verletzung von Laufzeitbedingungen möglich ist. Zeitaufwendige Iterationen zwischen Platzierung und Verdrahtung können so vermieden werden.

Abstract

Over many years transistor feature size and wire width of integrated circuits has decreased while the maximum clock frequency has increased at the same time. Due to the miniaturization of the structure sizes gate delays decrease whereas the interconnect delay per unit length increases. The modelling of interconnect delay has been based on statistical models in most EDA tools. For new technologies with gate and interconnect delays in the same magnitude these models lead to an unacceptable error in path delay calculation.

In this work, a new placement algorithm and a new detailed routing method are presented which take interconnect delay into account in early design phases with high accuracy. The placement algorithm is based on force-directed placement and extends the well-known approach by two new aspects: First, during the iterative improvement of the placement a path force, that models the slack, is used to improve the placement. Second, global routing is done simultaneously with placement. This allows a good modelling of the detailed routing during the placement phase.

The new detailed routing method is based on an interconnect library. Precharacterized interconnect elements with different shapes and a different number of vias are stored in this library. It also contains the resistance and capacitance to ground for every interconnect element. During detailed routing interconnect elements are selected and placed. While initial routing is done randomly, routing improvement uses an optimization method that is based on simulated annealing.

The results show that the new layout synthesis method has relevant advantages compared to a traditional design flow. The capability of the implemented prototype tool is demonstrated by experimental results using the MCNC benchmark suite. It can be shown, that the lengths of the critical paths are reduced, while the lengths of all non-critical paths increase. Due to the library based routing the difference between the estimated path delays after the placement and the path delays after the routing is getting smaller. This allows to predict already after the placement step, whether a routing without constraint violations can be done. Thus, time consuming iterations between placement and routing can be avoided.

Stichworte: Layout, Platzierung, Bibliotheksbasierte Verdrahtung, Timing-Closure-Problem

Keywords: Layout, Placement, Library-based Routing, Timing Closure Problem

1 Einleitung

Die kontinuierlich abnehmenden Strukturgrößen integrierter Schaltungen bei gleichzeitig steigenden Taktraten erfordern ständig neue Algorithmen für den Entwurfsprozess. Abbildung 1.1 zeigt den Anstieg der maximalen Taktrate auf einem Chip und gleichzeitig den Verlauf der minimalen Strukturgröße basierend auf einer Prognose der International Technology Roadmap for Semiconductors [ITRS02].

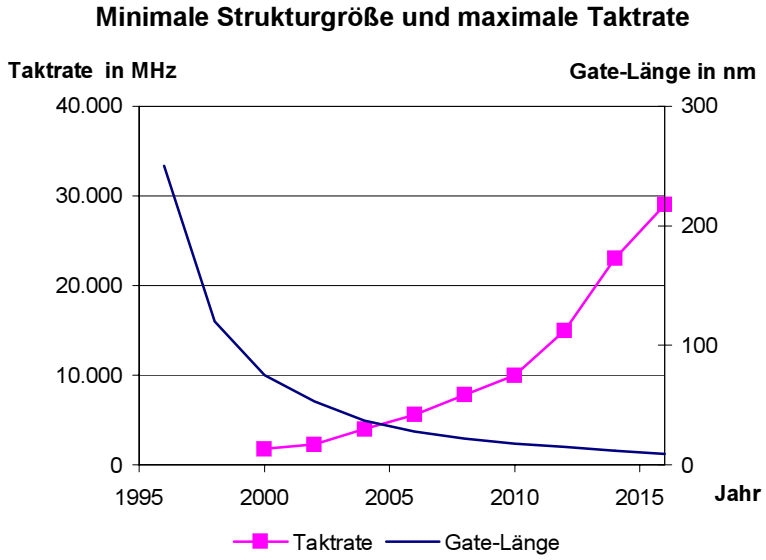

Abbildung 1.1: ITRS-Roadmap für Taktrate und minimale Strukturgröße

Im Zuge der Verkleinerung der minimalen Strukturgrößen auf einem Chip nimmt die Verzögerungszeit der einzelnen Gatter einer integrierten Schaltung ab, während gleichzeitig der Widerstand und die Kopplungskapazitäten pro Länge und damit auch die Verzögerungszeiten pro Länge des Verbindungsnetzwerks ansteigen. Dadurch ist eine stärkere Berücksichtigung der Verzögerungszeiten für die Entwicklung zukünftiger Designflows erforderlich. Viele der zurzeit im Einsatz befindlichen EDA-Werkzeuge sind deshalb für die in den nächsten Jahren erwarteten Technologien nicht mehr geeignet. Sie setzen in den meisten Fällen voraus, dass die Verzögerungszeiten aller Gatter in einem Pfad die Verzögerungszeiten durch das Verbindungsnetzwerk deutlich übersteigen. Diese Annahme ist in Zukunft nicht mehr zutreffend; für das Jahr 2006 sagt die Semiconductor Industry Association [ITRS02] sogar eine Delay-Verteilung zwischen Gattern und Leitungen von 20:80 voraus. Für den Entwurf solcher Schaltungen sind deshalb völlig neue Entwurfsverfahren erforderlich.

In den folgenden zwei Abschnitten soll zunächst ein Überblick über den Entwurfsprozess integrierter digitaler Schaltungen gegeben werden, um dann das Problem zu erläutern, welches sich aus der geänderten Verteilung der Verzögerungszeiten zwischen Gattern und Leitungen ergibt. Anschließend werden die Ziele dieser Arbeit vorgestellt.

1.1 Entwurf integrierter Schaltungen

Die Entwurfsschritte beim Design einer integrierten Schaltung lassen sich mit Hilfe des so genannten Y-Modells beschreiben. Das von [Gajski83] entwickelte Modell ist in Abbildung 1.2 dargestellt.

Im Y-Modell wird der Entwurfsraum in Ebenen und in Sichten unterteilt, wobei jede Achse im Diagramm eine Sicht und jeder Kreis eine Ebene repräsentiert. Die Ebenen dienen der Darstellung der unterschiedlichen Abstraktionsgrade bzw. der unterschiedlichen Detaillierungsgrade. Dabei nimmt der Grad der Detaillierung im Y-Modell von außen nach innen zu, d.h. die Abstraktionsebene ist im Zentrum am niedrigsten.

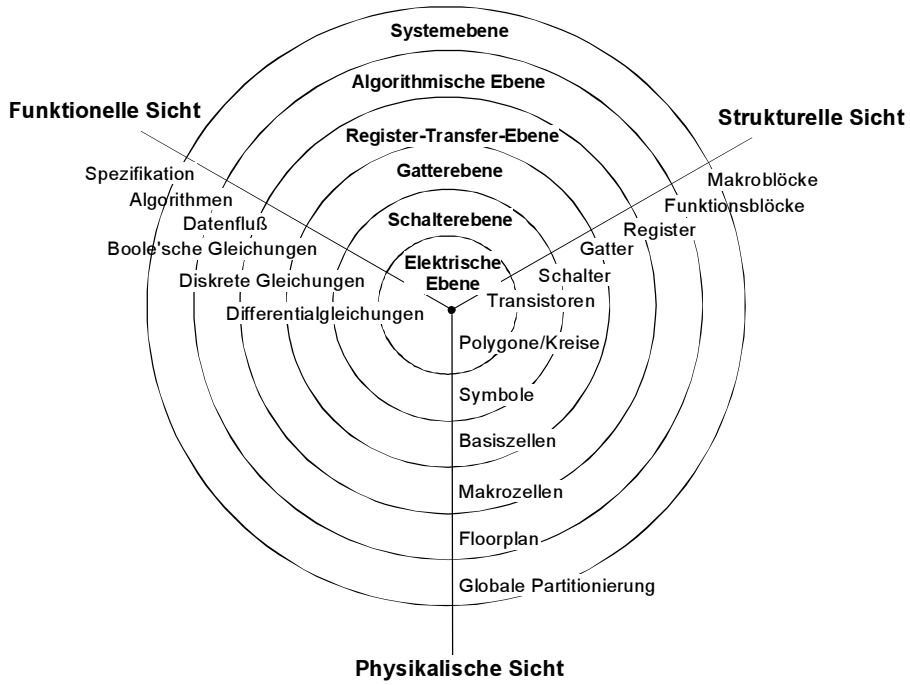

Abbildung 1.2: Y-Modell für den Entwurfsprozess

Die drei unterschiedlichen Sichten beschreiben unterschiedliche Repräsentationen einer Schaltung:

- Die funktionelle Sicht einer Schaltung beschreibt das Verhalten einer Schaltung. Die Schaltungsbeschreibung in der funktionellen Sicht erfolgt in den meisten Fällen mit Hilfe einer Hardware-Beschreibungssprache wie beispielsweise Verilog.
- Die strukturelle Sicht beschreibt die logische Struktur des Entwurfsgegenstands anhand von Komponenten und Verbindungen. Dabei verwendet man zur strukturellen Beschreibung im Entwurfsprozess in den meisten Fällen Netzlisten.

- Die physikalische Sicht beschreibt die konkrete Realisierung aller Elemente mit Hilfe realer Komponenten. In dieser Sicht erfolgt die Beschreibung der geometrischen Anordnung und Ausdehnung aller Elemente.

Ziel des Entwurfsprozesses einer integrierten Schaltung ist das geometrische Chiplayout. Dieses entspricht im Y-Diagramm der elektrischen Ebene in der physikalischen Sicht.

Während der Zielpunkt des Entwurfsprozesses fest definiert ist, kann der Entwurf an unterschiedlichen Positionen im Y-Diagramm beginnen. Bei modernen Schaltungen beginnt der Entwurf in den meisten Fällen in der funktionalen Sicht auf einer höheren Ebene. Der Weg von Start- zum Zielpunkt des Entwurfsprozesses kann in Abhängigkeit von der Technologie und dem Entwurfsstil unterschiedlich verlaufen.

Jeder konstruktive Entwurfsschritt, der den Entwickler einen Schritt näher zum Zielpunkt bringt, wird als Syntheseschritt bezeichnet. Dabei werden in jedem Syntheseschritt dem Entwurf Informationen hinzugefügt. Im Y-Diagramm entspricht einem Syntheseschritt ein Sicht- und/oder Ebenenwechsel. Einen Prüfschritt, der in umgekehrter Richtung zu einer abstrakteren Darstellung führt und sowohl einen Sicht- als auch einen Ebenenwechsel beinhalten kann, bezeichnet man als Analyseschritt. Analyseschritte dienen im Entwurfsprozess der Prüfung oder Validierung vorangegangener Entwurfsschritte.

1.2 Physikalischer Entwurf

Der physikalische Entwurf ist ein Teil des Entwurfsprozesses und entspricht im Y-Diagramm dem Übergang von der strukturellen Sicht auf die physikalische Sicht. Während dieses Übergangs wird das geometrische Layout einer Schaltung generiert.

Während des physikalischen Entwurfs sind folgende Syntheseschritte durchzuführen:

- In der Flurplanungsphase wird die gesamte Layoutfläche grob in Bereiche aufgeteilt, die bestimmten Schaltungsteilen für die Platzierung und Verdrahtung zugewiesen werden.
- In der Platzierungsphase werden alle Elemente der Netzliste auf der zur Verfügung stehenden Layoutfläche angeordnet.
- Die Verdrahtung bestimmt die geometrische Form der Verbindungen zwischen den bereits platzierten Zellen.

Zusätzlich zu diesen Syntheseschritten ist eine Vielzahl von Prüfschritten, wie Extraktion, Design-Rule-Check oder Layout-versus-Schematic erforderlich.

1.3 Timing-Closure-Problem

Während des Entwurfsprozesses müssen die Ergebnisse der später im Designflow folgenden Entwurfsschritte abgeschätzt, d.h. modelliert werden, um eine Bewertung des aktuellen Entwurfsschritts zu ermöglichen. Diese Modellierung ist umso ungenauer, je früher diese erfolgt.

Bereits während der Synthese der Schaltung, die dem Übergang zwischen funktioneller Beschreibung und struktureller Darstellung entspricht, müssen zur Abschätzung der Verzögerungszeiten die Ergebnisse des gesamten physikalischen Entwurfs modelliert werden, da eine Abschätzung der Verdrahtung zur Bewertung erforderlich ist.

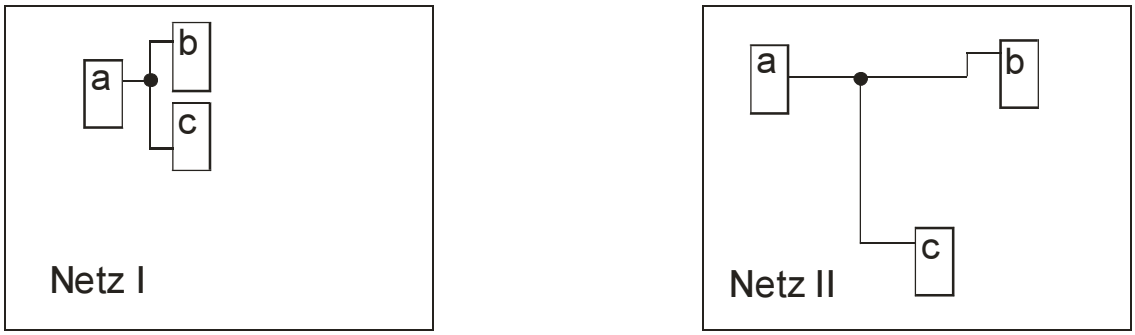

Abbildung 1.3: Modellierungsfehler beim Einsatz von Wire-Load-Modellen

In Abbildung 1.3 erkennt man zwei unterschiedliche Netze, die aus den gleichen Zellen a, b, c und einem Verbindungsnetzwerk bestehen. Aufgrund der längeren Leitungen der Anordnung II sind der Widerstand und die Eigenkapazität des Netzes II größer als die des Netzes I. Verwendet man zur Abschätzung der Verzögerungszeit vom Eingang zum Ausgang eines Netzes ein einfaches RC-Modell, so dass die Verzögerungszeit T proportional dem Produkt aus Widerstand und Lastkapazität ist, ergibt sich für die unterschiedlichen Platzierungen des gleichen Netzes eine unterschiedliche Verzögerungszeit. In den frühen Phasen des Entwurfs sind die zu einer genaueren Modellierung erforderlichen Informationen über die geometrische Anordnung der Zellen aber noch nicht verfügbar, so dass man statistische Modelle wie zum Beispiel die so genannten Wire-Load-Modelle einsetzt. Diese Modelle bestimmen die Signalverzögerungszeit aufgrund der Anzahl der getriebenen Gatter und schätzen im Beispiel von Abbildung 1.3 die Verzögerungszeit von Netz I und Netz II gleich ab.

Diese Genauigkeit war in den letzten Jahren ausreichend, da die Leitungsverzögerungszeit in der Regel nur maximal 20% der Gesamtverzögerungszeit eines Pfads ausgemacht hat. Ein Pfad ist dabei eine Reihe miteinander verbundener Gatter und beginnt und endet jeweils an einem Flipflop. Damit führte ein Modellierungsfehler bei einer Leitungsverzögerung von 50% bezogen auf die Gesamtverzögerungszeit eines Pfads

lediglich zu einem Fehler von 10%. Die Wire-Load-Modelle boten so für die meisten Schaltungen eine ausreichende Genauigkeit. Bei der bereits in den nächsten Jahren erwarteten Umkehr der Verzögerungszeit-Verhältnisse würde dieser Modellierungsfehler beim Pfad-Delay zu einem nicht tolerablen Fehler von 40% führen.

Der wesentliche Grund für die beschriebene Problematik liegt in der Teilung des Entwurfs in die voneinander unabhängigen Schritte Synthese, Platzierung und Verdrahtung. Sie führt bei neueren Technologien mit Strukturgrößen von weniger als 0,25 µm dazu, dass die abgeschätzten Verzögerungszeiten früherer Entwurfsschritte zu einem späteren Zeitpunkt im Designflow nicht erreicht werden können.

Die Unterschiede zwischen den in frühen Entwurfsschritten modellierten Verzögerungszeiten und den nach der Fertigstellung des Layouts berechneten Verzögerungszeiten einer Schaltung bezeichnet man als Timing-Closure-Problem.

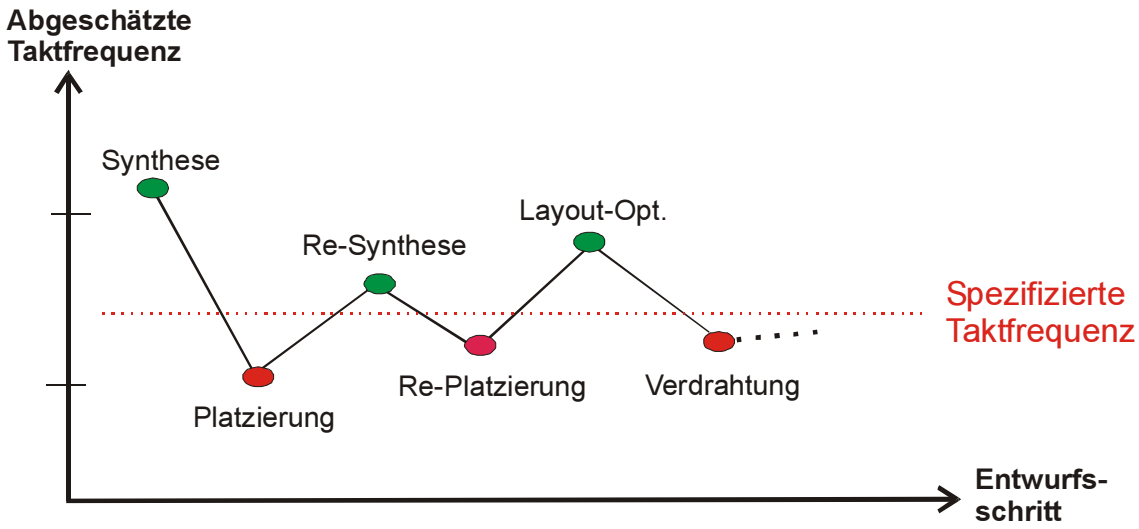

Abbildung 1.4: Maximale Taktrate in unterschiedlichen Entwurfsphasen

In Abbildung 1.4 ist die geschätzte maximale Taktfrequenz einer digitalen Schaltung während der verschiedenen Phasen des Entwurfs nach [Scheffer01] dargestellt. Man erkennt, dass es eine große Differenz der maximalen Taktrate zwischen Synthese und Platzierung gibt. Vergleichbar groß ist auch die Differenz der maximalen Taktrate zwischen der optimierten Platzierung und der Verdrahtung. Bei den Platzierungsschritten bzw. der Verdrahtung wird die Taktrate der Schaltung jeweils so reduziert, dass die spezifizierte Taktfrequenz nicht mehr eingehalten werden kann.

Die starken Schwankungen in der Abschätzung der maximalen Taktrate basieren fast ausschließlich auf den ungenauen Modellen, die bei der Synthese und der Platzierung verwendet werden. Aufgrund der Teilung des Entwurfsprozesses in voneinander unabhängige Phasen ist der Einsatz von Modellen aber nicht vermeidbar.

In den letzten Jahren hat man versucht, das Timing-Closure-Problem mit bestehenden Ansätzen und EDA-Werkzeugen zu lösen. Das hat dazu geführt, dass bei vielen Schaltungen nach der ersten Platzierung eine erneute Synthese (vgl. Abbildung 1.4) basierend auf den Platzierungsinformationen durchgeführt werden muss. Die Platzierungsinformationen führen dann zu einer höheren Genauigkeit der folgenden Modellierung im Resyntheseschritt. Dadurch kann eine Netzliste erzeugt werden, die zu weniger Laufzeitverletzungen im darauf folgenden Platzierungsschritt führt. Diese Schleife zwischen der Synthese und den verschiedenen Schritten der Layoutgenerierung wird bei manchen Designs mehrfach durchlaufen. Um zu häufige Iterationen zwischen Synthese und Platzierung zu verhindern und die Verzögerungszeiten der Schaltung nach der Platzierung zu reduzieren, haben sich in den letzten Jahren verschiedene Werkzeuge zur Layoutverbesserung etabliert, die in Abschnitt 3.4 genauer beschrieben werden. Dabei ist jedoch zu berücksichtigen, dass in der nachfolgenden Verdrahtungsphase das Problem der Laufzeitabweichungen erneut auftreten kann, wenn sich die vorgenommene Abschätzung erneut als zu optimistisch erweist. Dies erfordert, wenn die Verdrahtung - wie in Abbildung 1.4 dargestellt - die erforderlich maximale Taktrate nicht erreicht, eine wiederholte Platzierung und u. U. sogar eine erneute Synthese.

Dieses Vorgehen im Entwurfsprozess kostet sehr viel Zeit, da ein Platzierungsdurchlauf beispielsweise mehr als einen Tag dauern kann.

1.4 Motivation und Zielsetzung

Die traditionelle Divide-and-Conquer-Strategie mit der Teilung des Entwurfsprozesses in Synthese, Platzierung und Verdrahtung erweist sich für den Entwurf hochkomplexer Schaltungen mit Taktfrequenzen von mehr als zwei Gigahertz als ungeeignet. Deshalb ist es im Zuge von immer kürzer werdenden Time-to-Market-Zeiten wichtig, eine neue Designmethodik zu entwickeln, die einen einmaligen Designdurchlauf mit einem „Correct-by-Construction“-Ansatz verfolgt. Nur auf diesem Wege kann die Entwurfszeit deutlich verkürzt und können die Entwicklungskosten gesenkt werden.

Ziel dieser Arbeit ist es daher, zur Lösung des Timing-Closure-Problems einen neuartigen Beitrag zu leisten. Dazu wird ein Designflow vorgestellt, der die Eigenschaften des Verbindungsnetzwerks in früheren Entwurfsphasen stärker berücksichtigt als andere aktuell zum Einsatz kommende Entwurfsverfahren und so Iterationen zwischen den Entwurfsschritten vermeidet. Dieser Designflow basiert auf drei neuen Algorithmen: Der erste ist ein neuartiger Platzierungsansatz zur Berücksichtigung von Pfad-Constraints, welche die maximal zulässige Verzögerungszeit vom ersten zum letzten Gatter eines Pfads beschreiben. Der zweite Algorithmus ist ein Ansatz für die simultane Platzierung

und Globalverdrahtung, der ohne eine Modellierung der Globalverdrahtung bei der Platzierung arbeitet. Der dritte ist ein bibliotheksbasierter Verdrahtungsalgorithmus.

Im Pfad-Constraint-basierten Platzierungsalgorithmus kommt eine neue Kostenfunktion zur Berücksichtigung der Constraints zur Bewertung der Platzierungsschritte zum Einsatz. Während die meisten Platzierungsverfahren zur Bewertung eines Platzierungsschritts die Gesamtverdrahtungslänge verwenden, werden in diesem Ansatz für alle Pfade Längenrestriktionen erzeugt. Diese Restriktionen werden nicht nur zur Bewertung der Platzierung verwendet, sondern auch zu ihrer iterativen Verbesserung. Dieses Vorgehen ermöglicht die Erzeugung einer Platzierung, die die Laufzeitrestriktionen aus der Synthese einhält, sofern dieses möglich ist.

Im zweiten Teil des neuen Designflows wird basierend auf der neuen Kostenfunktion ein Algorithmus entwickelt, der die Platzierung und die Verdrahtung nicht mehr getrennt voneinander behandelt, sondern die Globalverdrahtung gleichzeitig mit der Platzierung durchführt. Dieses Vorgehen bietet den entscheidenden Vorteil, dass keine Modellierung der Globalverdrahtung während der Platzierung mehr erforderlich ist, wodurch keine Iterationen zwischen der Platzierung und der Globalverdrahtung mehr auftreten.

Gleichzeitig mit der Zellplatzierung werden die Start- und Endpunkte aller Leitungssegmente festgelegt. Ein Leitungssegment wird in diesem Zusammenhang als ein Leitungsstück einer bestimmten Form und einer begrenzten Länge verstanden. Dadurch kann eine gemeinsame Bewertung von Platzierung und Globalverdrahtung in einem Schritt erfolgen. Eine Anpassung der Platzierung und Globalverdrahtung bei nicht erfüllten Pfad-Restriktionen kann ebenfalls in einem Schritt durchgeführt werden.

Die Festlegung der physikalischen Ausführungsform der Leitungssegmente - entsprechend der Detailverdrahtung - erfolgt im dritten Teil des neuen Designflows. Dazu kommt eine Bibliothek von charakterisierten Leitungsformen (Shapes) zum Einsatz, die vergleichbar zu einer Zellbibliothek aufgebaut ist. Diese Bibliothek beinhaltet Verdrahtungselemente unterschiedlicher Form mit einer variablen Anzahl von Vias. Für jedes Element der Bibliothek werden während der Charakterisierung der Widerstand, die Kapazität und die Verzögerungszeit bestimmt.

In der Detailverdrahtungsphase wird für jedes Netz ein Verdrahtungselement aus der Bibliothek ausgewählt und an die geometrischen Randbedingungen angepasst. Aufgrund der Längenbeschränkung der Leitungssegmente sind die Verzögerungszeiten auf den Leitungssegmenten im Gegensatz zu denen der Gesamtleitung mit ausreichender Genauigkeit abschätzbar. Damit kann bereits nach jedem Platzierungsschritt eine genaue Modellierung der Verdrahtung erfolgen.

Dieser neue Designflow bietet gegenüber dem konventionellen Entwurfsablauf entscheidende Vorteile. Durch den Einsatz eines Pfad-Constraint-basierten Platzierungsalgorithmus werden die Pfad-Constraints deutlich besser als in anderen Designflows berücksichtigt. Es ist daher zu erwarten, dass die Anzahl der Pfade mit Constraint-Verletzungen abnimmt. Aufgrund der gleichzeitigen Durchführung von Platzierung und Globalverdrahtung ist in dieser Phase eine genauere Abschätzung der Verdrahtungslänge möglich. Diese Genauigkeit wird durch den bibliotheksbasierten Verdrahter unterstützt, da die bei der Detailverdrahtung zum Einsatz kommenden Verdrahtungselemente durch die Bibliothek im Voraus bekannt sind.

Die weitere Arbeit ist wie folgt gegliedert: In Kapitel 2 werden zur Einordnung dieser Arbeit in einen Gesamt-Designflow der ideale Entwurfsablauf und - zum Vergleich - ein industrieller Designflow für den Entwurf integrierter digitaler Schaltungen vorgestellt. Dieses Kapitel beinhaltet außerdem eine Übersicht über die in den verschiedenen Entwurfsphasen zum Einsatz kommenden Modelle, die die Ursache für das Timing-Closure-Problem sind. In Kapitel 3 werden die grundlegenden Algorithmen zur Platzierung und Verdrahtung integrierter digitaler Schaltungen und ihre aktuellen Modifikationen beschrieben. In Kapitel 4 wird das im Rahmen dieser Arbeit entwickelte Verfahren zur Platzierung und Globalverdrahtung in einem Schritt dargestellt. Kapitel 5 beschreibt den neu entwickelten Algorithmus zur bibliotheksbasierten Detailverdrahtung. Das anschließende Kapitel stellt die Implementierung und die erzielten Ergebnisse vor. Abgeschlossen wird die Arbeit durch eine Zusammenfassung in Kapitel 7.

2 Entwurfsstil, Designflow und Verzögerungszeitmodelle

Zur Entwicklung des neuen Standardzell-Designflows ist ein vollständig neuer Entwurfsablauf erforderlich, in dem mehrere Entwurfsschritte eines Standard-Designflows zusammengefasst werden und andere entfallen.

Im Abschnitt 2.1 werden zunächst die Eigenschaften und Besonderheiten des Standardzellentwurfs vorgestellt, für den dieser neue Designflow entwickelt wurde.

Zum Vergleich mit den bisher etablierten Entwurfsabläufen wird ein idealisierter Designflow in Abschnitt 2.2 beschrieben. Dieser Ansatz führt in der Praxis zu dem in Abschnitt 1.3 beschriebenen Timing-Closure-Problem. Dieses wird durch die bei der Modellierung von Verzögerungszeiten (Delays) zum Einsatz kommenden Modelle verursacht. Die in den unterschiedlichen Designphasen verwendeten Modelle werden daher in Abschnitt 2.3 vorgestellt.

Zur Lösung des Timing-Closure-Problems werden in der Praxis verschiedene Iterationen im Entwurfsablauf durchgeführt. Diese sollen zu einer Konvergenz zwischen den während der Synthese abgeschätzten und den nach der Layouterstellung berechneten Verzögerungszeiten führen. Exemplarisch wird in Abschnitt 2.4 ein Designflow beschrieben, der zurzeit bei verschiedenen Halbleiterherstellern zum Einsatz kommt.

In Abschnitt 2.5 werden zwei neuartige Designflows vorgestellt, die Lösungsansätze für das Timing-Closure-Problem bieten und seit kurzem auch industriell eingesetzt werden.

2.1 Standardzellentwurf

Der Standardzellentwurf ist durch eine Einschränkung der geometrischen Freiheitsgrade charakterisiert. Dazu wird einmalig vom Halbleiterhersteller eine Bibliothek von Zellen geringer Komplexität entworfen, auf die dann in allen Entwürfen zurückgegriffen wird. Die Layouts dieser Zellen können mit einem beliebigen Entwurfsstil entworfen sein. In der Praxis werden Standardzellen zur Flächenoptimierung häufig von Hand entworfen. Standardzellen sind hinsichtlich ihrer Korrektheit und ihrer elektrischen Eigenschaften vollständig verifiziert.

Beim Standardzellentwurf werden gewöhnlich rechteckige Zellen identischer Höhe, aber unterschiedlicher Breite verwendet. Alle Zellen führen, wie in Abbildung 2.1 dargestellt, am oberen und unteren Zellrand die Versorgungsleitungen in der gleichen Metalllage.

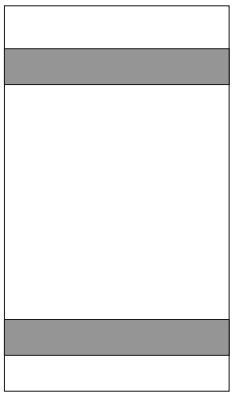

Abbildung 2.1: Typische Standardzelle

In der Regel werden innerhalb der Zellen nur zwei bis drei Metalllagen zur Verdrahtung verwendet, so dass in Technologien mit mehr als drei Metalllagen die Ebenen ab Lage vier ausschließlich zur Verdrahtung der Zellen miteinander verwendet werden.

Die Platzierung von Standardzellen auf der Layoutfläche erfolgt - wie in Abbildung 2.2 dargestellt - in Reihen. Durch die festen Positionen der Versorgungsleitungen erfolgt die Verbindung dieser Leitungen ohne zusätzliche Verbindungsstrukturen durch das Nebeneinanderlegen der Zellen in Reihen (Abutment). Sind nicht alle Zellreihen vollständig ausgefüllt, werden die Lücken durch so genannte Füllzellen (Feedthrough-Cells) aufgefüllt. Diese Zellen sind bis auf die Versorgungsleitungen leer und ermöglichen eine Verdrahtung durch die Zellreihen hindurch. Dieses ist insbesondere bei einer kleinen Anzahl von Metalllagen notwendig, wenn die Verdrahtung orthogonal zu den Zellreihen nicht über die Zellen hinweg erfolgen kann.

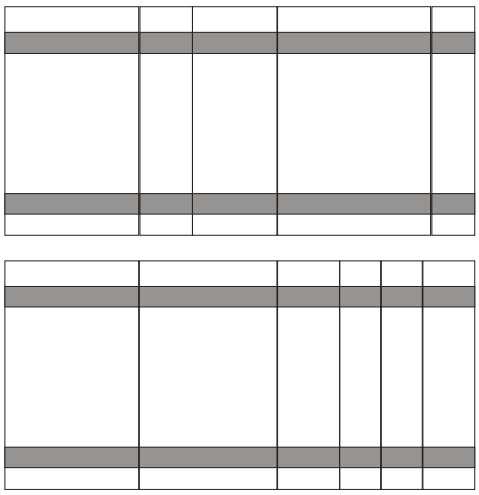

Abbildung 2.2: Standardzelllayout

Folgende - in Kapitel 1 genannte - Entwurfsschritte vereinfachen sich bei einem Standardzell-Designflow im Vergleich zu einem allgemeinen Designablauf:

Schaltungs-Synthese

Die Synthese der Schaltung kann voll automatisch erfolgen, so dass die Eingabe des Designers in einer Hochsprache ohne manuelle Eingriffe in eine strukturelle Beschreibung umgesetzt wird.

Platzierung

Die Platzierung der Standardzellen erfolgt nicht kontinuierlich auf der gesamten Fläche, sondern die Zellen werden in Reihen platziert. Dazu gibt es zwei Möglichkeiten: Entweder werden die Zellen zunächst jeweils einer Reihe zugewiesen und anschließend erfolgt die Festlegung der Zellreihenfolge innerhalb jeder Reihe oder nach einer Platzierung ohne Reihenrestriktionen wird die Reihenzuweisung in einem Legalisierungsschritt durchgeführt.

Verdrahtung

Bei der Verdrahtung einer Schaltung muss zwischen der Verdrahtung der Versorgungsleitungen (Powerrouting) und der Verdrahtung von Signalleitungen unterschieden werden. Durch die feste Position der Versorgungsleitungen und die Anordnung der Zellen in Reihen ist kein separates Powerrouting wie bei allgemeinen Zellen oder Full-Custom-Entwürfen notwendig. Das Verdrahtungsproblem der Signalleitungen kann sich in Abhängigkeit von der Anzahl der Verdrahtungslagen vereinfachen. Bei wenigen Verdrahtungslagen erfolgt die Verdrahtung teilweise zwischen den Standardzellen in so genannten Verdrahtungskanälen wie in Abbildung 2.3 dargestellt. Die Anschlusspins der Zellen befinden sich dabei jeweils an der Ober- und Unterseite des Kanals.

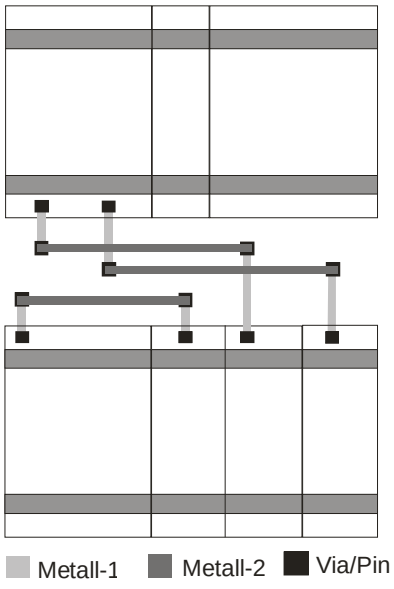

Abbildung 2.3: Kanalverdrahtung

Spezielle Algorithmen [Brück93] ermöglichen eine effiziente Verdrahtung innerhalb der Kanäle. Im Allgemeinen ist die Komplexität der Kanalverdrahtung geringer, als die der allgemeinen Verdrahtung über der gesamten Layoutfläche. Bei modernen Technologien mit vier und mehr Metalllagen gibt es diesen Vorteil in den meisten Fällen nicht mehr. Aufgrund der Vielzahl der Metalllagen werden die Zellreihen ohne Verdrahtungskanal direkt nebeneinander platziert und die Verdrahtung erfolgt auf den Metalllagen über den Zellen. Dieses entspricht dem Fall der allgemeinen Verdrahtung.

Die Einschränkung der Freiheitsgrade bei der Layouterstellung durch die einheitlich hohen Standardzellen zur Reduktion des Entwurfsaufwands führt zu einem entscheidenden Nachteil: Die Fläche einer mit Standardzellen realisierten Schaltung wird immer größer sein als die Fläche eines Chips, dessen Layout durch einen Designer ohne die Beschränkung auf Standardzellen manuell entworfen und optimiert wurde. Allerdings sind für den Fall von handoptimiertem Layout die Entwurfskosten und das Fehlerrisiko deutlich höher. Im Allgemeinen wird man deshalb nur bei Standardschaltungen, die in hohen Stückzahlen gefertigt werden, das Layout von Hand optimieren, da die Herstellungskosten unmittelbar proportional zur Chipfläche sind.

2.2 Idealisierter Designflow

Der Entwurf einer digitalen Schaltung wird grob in zwei Phasen eingeteilt: Im Frontend-Design erfolgt die funktionale Beschreibung der Schaltung und die Umsetzung in eine Gatternetzliste, während in der Backend-Designphase das geometrische Layout erzeugt und verifiziert wird.

Ausgangspunkt des Frontend-Entwurfs ist in den meisten Fällen eine funktionelle Schaltungsbeschreibung in einer Hardwarebeschreibungssprache wie Verilog oder VHDL. Mit Hilfe eines Synthesewerkzeugs wird die Verhaltensbeschreibung in eine strukturelle Beschreibung - eine Gatternetzliste - umgesetzt. Dieses entspricht im Y-Modell dem Übergang von der funktionellen Sicht in die strukturelle Sicht und gleichzeitig mindestens einem Ebenenwechsel von der Register-Transfer-Ebene auf die Gatterebene. Die während der Synthese erzeugte Gatternetzliste stellt den Übergang zwischen Front- und Backend-Design dar.

Im Backend-Design wird - ausgehend von der Gatternetzliste - das geometrische Layout erstellt. Aufgrund der Komplexität des Layoutproblems wird die Layouterstellung in drei Schritte eingeteilt: Im ersten Schritt erfolgt das Floorplanning, in dem eine grobe Zuweisung zwischen den funktionalen Elementen und den Layoutteilflächen erfolgt. Im zweiten Schritt der Layouterstellung erfolgt die Platzierung der Zellen. Dabei wird jeder Zelle ein eindeutiger Platz auf der Layoutfläche zugeordnet. Zur Bewertung der Platzierung ist die Verdrahtung zwischen den Zellen erforderlich, da lange Verbindungen

zwischen ungünstig zueinander platzierten Zellen zu hohen Verzögerungszeiten führen können. Um nicht nach jedem Platzierungsschritt die Verdrahtung berechnen zu müssen, wird die Verdrahtung mit einfachen, schnell zu berechnenden Verdrahtungsmodellen abgeschätzt. Der dritte Schritt der Layoutgenerierung ist die Verdrahtung.

Abschließend sind Verifikationsschritte erforderlich. Diese sollen zum einen geometrische Entwurfsregelverletzungen lokalisieren und zum anderen Abweichungen zwischen der Netzliste und der durch das Layout repräsentierten Schaltung aufdecken. Die Ursache für geometrische Entwurfsregelverletzungen und Schaltungsabweichungen können sowohl manuelle Eingriffe in den Layoutprozess, Werkzeugfehler als auch parasitäre Elemente sein. Parasitäre Elemente entstehen im Layout beispielsweise durch die nichtidealen Eigenschaften der Leitungen (Leitungswiderstände) oder durch die räumliche Nachbarschaft mehrerer Elemente auf der Layoutfläche zueinander (Kopplungskapazitäten).

Der erste Verifikationsschritt ist die geometrische Layoutprüfung. Diese soll Verletzungen von Entwurfsregeln (Design Rules) finden. Die Entwurfsregeln repräsentieren technologische und fertigungstechnische Randbedingungen. Die wichtigsten Entwurfsregeln definieren minimale Abstände, Längen und Weiten sowie Überlappungen.

Die Überprüfung der Übereinstimmung der Gatternetzliste mit dem erzeugten Layout erfolgt in zwei Teilschritten. In der Extraktionsphase werden alle Bauelemente aus dem Layout extrahiert und in eine Netzliste überführt. Im Y-Diagramm entspricht das einem Übergang von der physikalischen in die strukturelle Sicht. Man unterscheidet bei der Extraktion zwischen der Bauelemente- und der Parasitenextraktion. Bei der Bauelementeextraktion werden nur die entworfenen Elemente extrahiert, während bei der Parasitenextraktion neben den entworfenen Bauelementen auch alle parasitären Bauelemente in die Netzliste eingefügt werden.

Nach der Extraktion kann die Prüfung auf Übereinstimmung der Nominalnetzliste mit der extrahierten Netzliste erfolgen. Dieser Vorgang wird als Layout-versus-Schematic (LVS) bezeichnet. Das Ergebnis des LVS ist die Aussage, ob beide Netzlisten übereinstimmen.

Zum Abschluss der Verifikation erfolgt eine Überprüfung der Pfad-Verzögerungszeiten mit Hilfe der statischen Timing-Analyse. In diesem Schritt werden für alle Pfade auf Basis der extrahierten Netzliste mit parasitären Elementen die Pfad-Verzögerungen bestimmt. Dabei wird für jeden Pfad zunächst der ungünstigste Signalwechsel - also low-high oder high-low – bestimmt und anschließend die Verzögerungszeit berechnet. Die berechnete Pfad-Verzögerung muss für alle Pfade kleiner als die Taktperiode sein.

Abbildung 2.4 zeigt zusammenfassend alle Schritte des Designflows. In jedem Entwurfsschritt erfolgt eine Modellierung der Verzögerungszeiten der betrachteten Schaltung. Die Modelle und die damit verbundene Genauigkeit der Verzögerungszeit-Abschätzung unterscheiden sich in den Schritten jedoch sehr stark. Im Abschnitt 2.3 werden die in den unterschiedlichen Phasen zum Einsatz kommenden Modelle zunächst vorgestellt und anschließend die damit verbundenen Probleme erläutert.

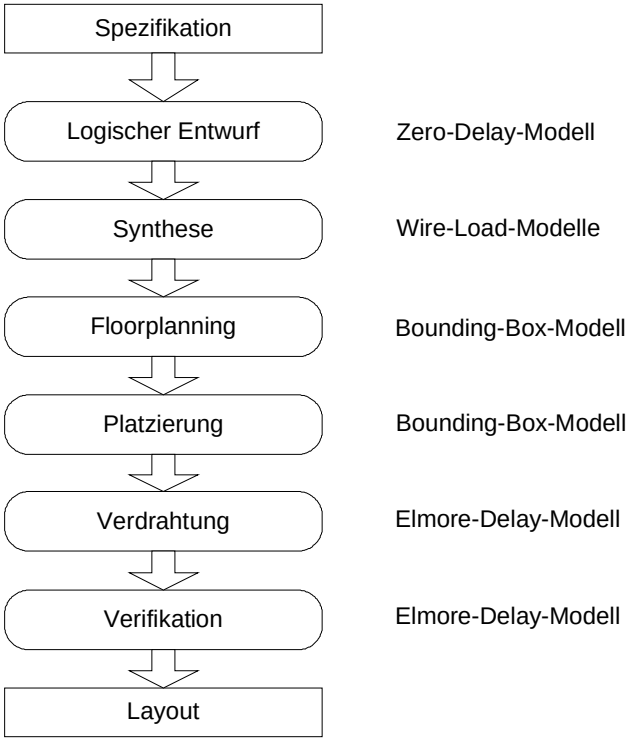

Abbildung 2.4: Typischer Digital-Designflow mit typischen Timing-Modellen

2.3 Timing-Modellierung

In jedem Schritt des Designflows müssen die Verzögerungszeiten der Zielschaltung modelliert werden. Diese Modellierung verwenden die Algorithmen in den verschiedenen Stufen des Designflows zur Durchführung der Layoutsyntheseschritte. Gleichzeitig ermöglicht die Timing-Modellierung die Erkennung von Timing-Problemen vor der Fertigstellung des Layouts. Die Modellierungsgenauigkeit steigt mit jedem Syntheseschritt des Designflows an, weil zusätzliche Informationen zu einer genaueren Berechnung zur Verfügung stehen.

Während der Synthese der Hochsprachenbeschreibung in eine Gatternetzliste, der so genannten Register-Transfer-Level-Synthese (RTL-Synthese), erfolgt die Modellierung mit statistischen Wire-Load-Modellen, die in Abschnitt 2.3.1 genauer beschrieben werden. In der Platzierungsphase kommen die in Abschnitt 2.3.2 beschriebenen Bounding-

Box-Modelle zum Einsatz, während in der Routing-Phase und den Extraktionsschritten häufig auf dem so genannten Elmore-Delay basierende Modelle verwendet werden. Diese Modelle werden in Abschnitt 2.3.3 beschrieben. Die Genauigkeit der Modelle wird in dieser Reihenfolge immer größer, gleichzeitig steigt der Berechnungsaufwand an. In Abschnitt 2.3.4 befindet sich eine Darstellung über neuartige Modellentwicklungen, die teilweise über eine einfache Längenabschätzung deutlich hinausgehen.

2.3.1 Wire-Load-Modelle

Wire-Load-Modelle sind statistische Modelle zur Abschätzung der Verzögerungszeit von jeweils einem Netz. Diese Modelle benötigen keine geometrischen Informationen, sondern schätzen die Verzögerungszeit ausschließlich aufgrund des Fanouts der treibenden Zelle ab. Als Fanout einer Zelle bezeichnet man dabei die Anzahl der getriebenen Zellen, d.h. die Anzahl der Eingänge anderer Zellen, die mit dem Ausgang der betrachteten Zelle verbunden sind. Beispielhaft ist die Fan-Out-Bestimmung in Abbildung 2.5 dargestellt.

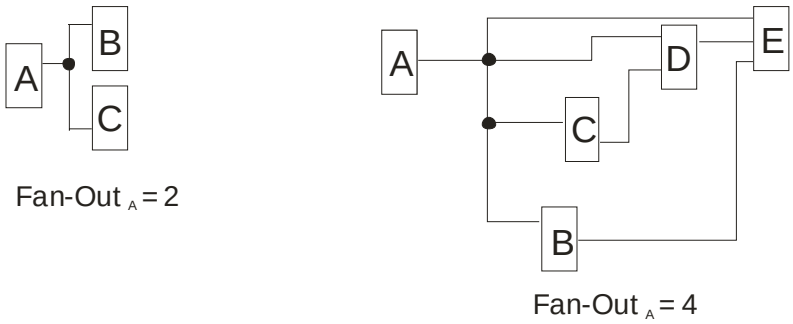

Abbildung 2.5: Fanout-Bestimmung

In der Synthese werden fast ausschließlich Wire-Load-Modelle eingesetzt, weil in diesem Schritt des Designprozesses noch keine Platzierungsinformationen zur Verfügung stehen. Ein Beispiel für ein Wire-Load-Modell ist in der folgenden Tabelle zu finden:

10K-WLM

Fanout	Länge/mm	Kapazität/nF	Widerstand/Ω	Fläche/μm^2
1	0.002	0.002	0.005	0.500
2	0.005	0.005	0.005	1.000
3	0.013	0.013	0.139	1.500
4	0.022	0.022	0.276	2.000
7	0.033	0.033	0.550	3.500
11	0.054	0.054	0.785	5.500

Tabelle 2.1: Beispiel für ein Wire-Load-Modell

Die Bestimmung der Verzögerungszeit eines Netzes erfolgt mit Tabelle 2.1 wie folgt: Treibt eine Zelle beispielsweise ein Netz mit drei Zellen, werden die zugehörigen Werte für den Widerstand und die Kapazität des Netzes aus der dritten Zeile der Tabelle 2.1 - entsprechend dem Fanout von drei - entnommen. In diesem Fall liest man aus der Tabelle eine Kapazität von 0.013 nF und einen Widerstand von 0.139 Ohm ab. Die Verzögerungszeit wird dann mit einem einfachen RC-Modell mit $T = R \cdot C$ berechnet. Für alle nicht in der Tabelle enthaltenen Fanout-Werte erfolgt die Verzögerungszeit-Berechnung durch lineare Interpolation. Beispielsweise berechnet sich die Verzögerungszeit eines Netzes mit einem Fanout von 10 zu

$$T_{10} = \frac{1}{4} \cdot Fanout(7) + \frac{3}{4} \cdot Fanout(11) . \qquad \textbf{(2.1)}$$

Die Daten für ein Wire-Load-Modell müssen durch Messungen bestimmt werden und sind jeweils nur für eine Technologie gültig. In den meisten Fällen werden die Wire-Load-Modelle für eine bestimmte Blockgröße erzeugt. Das in Tabelle 2.1 dargestellte Modell 10K-WLM ist für Blöcke mit weniger als 10.000 Zellen gedacht und hat die höchste Genauigkeit für Blöcke dieser Größe. Für größere Blöcke wird ein anderes Wire-Load-Modell mit höheren Kapazitäten und Widerständen bestimmt, da die durchschnittliche Leitungslänge in größeren Blöcken höher ist.

Die Genauigkeit dieser Modelle im Vergleich zu den nach der Fertigung gemessenen Daten ist beschränkt. Ursache dafür ist vorwiegend der statistische Charakter der Modelle. Dafür sind zur Berechnung dieser Modelle keine Geometriedaten erforderlich.

2.3.2 Bounding-Box-Modelle

Das einfachste Modell zur Berechnung von Verzögerungszeiten unter Berücksichtigung von Geometrieinformationen ist das so genannte Bounding-Box-Modell. Bei diesem Modell wird das umschreibende Rechteck aller an einem Netz beteiligten Pins berechnet. Der halbe Umfang des Rechtecks wird dann als Längenabschätzung des Netzes - wie in Abbildung 2.6 dargestellt - verwendet.

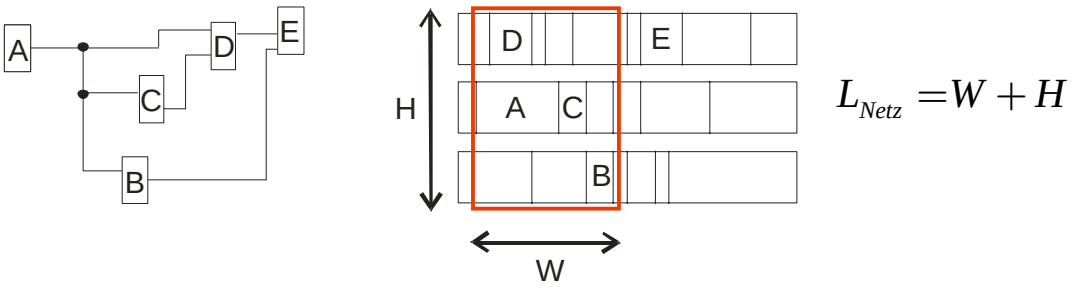

Abbildung 2.6: Bounding-Box-Modell

Die Verzögerungszeit eines Netzes wird mit Hilfe eines technologiespezifischen Faktors λ zu $T = \lambda \cdot L_{Netz}$ berechnet.

Diese Modelle bieten im Vergleich zu den Wire-Load-Modellen den Vorteil, dass reale Geometrieinformationen berücksichtigt werden und sind daher im Allgemeinen genauer als die Abschätzung mit einem Wire-Load-Modell.

Auf umschreibenden Rechtecken basierende Verzögerungszeit-Modelle lassen sich relativ schnell berechnen. Deshalb werden diese Modelle häufig in der Platzierungsphase zur Bewertung der Platzierungsiterationsschritte verwendet.

Diese Art der Verzögerungszeit-Berechnung bietet eine gute Approximation der Leitungslänge und damit auch eine gute Abschätzung des Widerstands sowie der Eigenkapazität einer Leitung. Alle anderen elektrischen Eigenschaften der Leitungen wie z.B. Kopplungskapazitäten bleiben jedoch unberücksichtigt.

2.3.3 Elmore-Delay

Das Elmore-Delay [Elmore48] ist allgemein wie folgt definiert:

$$T_{Delay} = \int_0^\infty t \cdot \dot{u}(t) dt \qquad \textbf{(2.2)}$$

Dabei ist $u(t)$ die Sprungantwort des betrachteten Systems. Diese Verzögerungszeit-Definition entspricht dem Moment erster Ordnung des Systems.

Wendet man Gleichung (2.2) auf eine Kette von RC-Elementen - wie in Abbildung 2.7 dargestellt - an [Rubinstein83], so erhält man folgende Gleichung zur Berechnung des Elmore-Delays.

$$T_{Delay,i} = T_{Delay,i-1} + R_i \cdot \sum_{k=i}^{n} C_k \qquad \textbf{(2.3)}$$

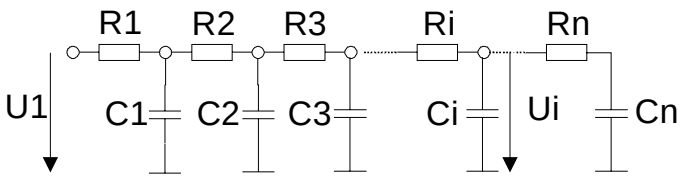

Abbildung 2.7: RC-Kette

Das Elmore-Delay vernachlässigt bei der Bestimmung der Verzögerungszeit alle Induktivitäten sowie die Kopplungseffekte zwischen den Elementen.

Bei der Berechnung des Elmore-Delays von elektrischen Schaltungen erfolgt die Modellierung der Leitungen in den meisten Fällen nicht durch verteilte RC-Elemente, sondern es wird jeweils ein Teil eines Netzes zwischen zwei Abzweigungen bzw. zwischen Start- oder Endpunkt des Netzes mit einem Widerstand und einer Kapazität modelliert. Dieses reduziert den Berechnungsaufwand, führt aber gleichzeitig zu einem Modellierungsfehler. Streng genommen wird dadurch nur eine Näherung des Elmore-Delays der elektrischen Schaltung berechnet.

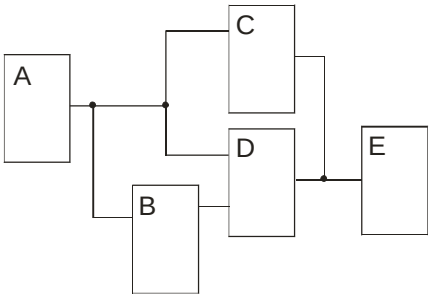

Abbildung 2.8: Beispielschaltung zur Berechnung des Elmore-Delays

Für das in Abbildung 2.8 dargestellte Beispiel erkennt man für das Netz, das die Gatter A und D miteinander verbindet, insgesamt fünf Teilnetze. Damit bestimmt man folgendes RC-Modell, in dem nur die Leitungskapazitäten nicht aber die Lastkapazitäten durch die Gatter dargestellt sind.

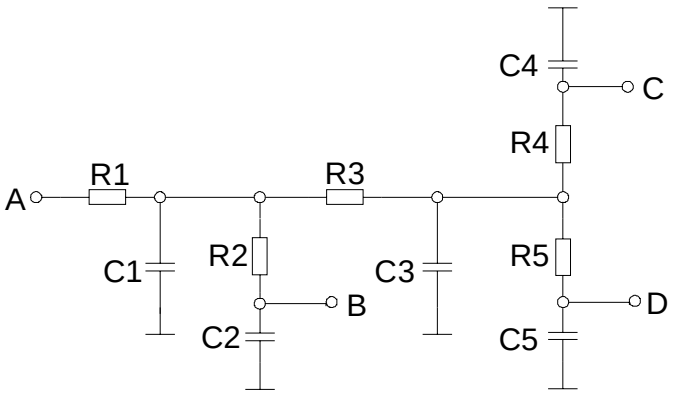

Abbildung 2.9: RC-Modell zur Berechnung des Elmore-Delays

Das Elmore-Delay lässt sich zur Berechnung anschaulich in vier Komponenten zerlegen [Paskalev02]:

- Senkenverzögerung:
  Die Senkenverzögerung entsteht durch die kapazitive Last der Signalsenke, die über den Widerstand der Leitung getrieben wird.

- Eigenverzögerung der Leitung:
  Die Eigenverzögerung ergibt sich aus dem Widerstand der Leitung, der die Eigenkapazität der Leitung treibt.

- Off-Path-Wire-Load-Verzögerung:
  In dieser Komponente des Elmore-Delays werden alle kapazitiven Lasten durch Leitungen berücksichtigt, die mit dem Pfad verbunden sind, aber nicht direkt auf dem Weg zwischen Quelle und Senke liegen.

- Off-Path-Gatterverzögerung:
  Die Off-Path Gatterverzögerung modelliert die kapazitive Last durch die mit dem Pfad verbundene Gatter, die nicht direkt zwischen Quelle und Senke liegen.

Für das in Abbildung 2.9 dargestellte Ersatzschaltbild berechnen sich die vier Verzögerungszeit-Komponenten wie folgt:

- Senkenverzögerung: $t_{Senke} = (R_1 + R_3 + R_5) \cdot C_{Gatter\ D}$
- Eigenverzögerung: $t_{Eigen} = R_1 \cdot (C_1 + C_3 + C_5) + R_3 \cdot (C_3 + C_5) + R_5 \cdot C_5$
- Off-Path-Wire-Load-Verzögerung: $t_{OP,WL} = R_1 \cdot C_2 + (R_1 + R_3) \cdot C_4$
- Off-Path-Gatterverzögerung: $t_{OP,G} = (R_1 + R_3) \cdot C_{Gatter\ C} + R_1 \cdot C_{Gatter\ B}$

Das Elmore-Delay des gesamten Pfads entspricht der Summe der vier Teilverzögerungen

$$T_{Delay} = t_{Senke} + t_{Eigen} + t_{OP,WL} + t_{OP,G} \tag{2.4}$$

und ist damit:

$$\begin{aligned} t_{Delay,\ AD} &= R_1 \cdot (C_1 + C_2 + C_3 + C_4 + C_5 + C_{Gatter\ B} \\ &+ C_{Gatter\ C} + C_{Gatter\ D}) + R_3 \cdot (C_3 + C_4 + C_{Gatter\ C} \\ &+ C_{Gatter\ D}) + R_5 \cdot (C_5 + C_{Gatter\ D}) \end{aligned} \tag{2.5}$$

2.3.4 Neue Modellentwicklungen

Neben diesen etablierten Modellen existieren mittlerweile viele Ansätze, die die Leitungsverzögerungen vor der Layouterstellung genauer modellieren. Diese so genannten Pre-Layout-Verzögerungszeit-Abschätzungen basieren zumeist auf komplexen statistischen Modellen. Ein Überblick über aktuelle Entwicklungen von Pre-Layout-Modellen kann in [Stroobandt01] nachgelesen werden.

Speziell für die Modellierung der Verdrahtung während der Platzierung gibt es ebenfalls neu entwickelte Modelle [Pedram89, Caldwell99]. Diese Modelle bieten für die Platzierung eine deutlich bessere Modellierung der Verdrahtungslänge im Vergleich zu den Bounding-Box-Modellen.

In [Bodapi00] wird erstmals ein völlig anderer Weg zur Modellierung der Verdrahtungslängen beschritten. In diesem Ansatz wird ein Black-Box-Modell der Platzierungs- und Verdrahtungswerkzeuge erstellt. Dabei wird versucht, eine Charakterisierung des verwendeten Platzierungs- und Verdrahtungswerkzeugs zu erzeugen.

Zur statistischen Abschätzung der Verdrahtungslängen werden in diesem Ansatz alle verwendeten Parameter in zwei Gruppen eingeteilt. Die globalen Parameter beschreiben die implementierungsunabhängigen Parameter des Designs wie beispielsweise Gesamtzahl der Zellen, Anzahl der Zwei-, Drei, Vier- und Mehrpunktnetze. Die lokalen Parameter beschreiben die Eigenschaften eines Netzes, indem die Anzahl der benachbarten Zwei-, Drei, Vier und Mehrpunktenetze beschrieben wird.

Anschließend wird die so genannte Basisdistanz bestimmt. Diese geht davon aus, dass zwei miteinander verbundene Zellen entweder unmittelbar nebeneinander oder unmittelbar übereinander platziert werden. Da diese Abschätzung die realen Anordnungen nicht ausreichend genau beschreibt, werden so genannte Zellanordnungen definiert. Diese beschreiben die Lagen von Zellen zueinander. Für jede Anordnung werden die Verdrahtungslänge sowie die Wahrscheinlichkeit, dass eine Anordnung verwendet wird, bestimmt. Die Wahrscheinlichkeiten für die Auswahl der unterschiedlichen Zellanordnungen wird experimentell für jedes Platzierungswerkzeug bestimmt. Die Berücksichtigung der Belegung durch bereits vorhandene Leitungen erfolgt lediglich auf Basis der Zwei-Punkt-Netze. Alle zu einem Netz nicht benachbarten Zwei-Punkt-Netze sind potenzielle Hindernisse und führen dazu, dass die Wahrscheinlichkeit, eine Zellanordnung mit einer längeren Leitungslänge zu selektieren, ansteigt.

Aus der Basislänge, den Zellanordnungen sowie der Belegungsabschätzung wird statistisch eine Abschätzung für die Netzlänge bestimmt. Der Vergleich zwischen der Abschätzung von [Bodapi00] und den realen Netzlängen hat gezeigt, dass der mittlere Modellierungsfehler bei ungefähr 25% liegt.

2.3.5 Bewertung der Modellierung

Bei der Modellierung von Verzögerungszeiten treten zwei wesentliche Probleme auf:

2.3.5.1 Modellierungsgenauigkeit

Die Genauigkeit der erforderlichen Modelle nimmt mit fortschreitendem Design kontinuierlich zu. Ursache für die Ungenauigkeiten der Modelle in den frühen Designstufen ist das Fehlen von Informationen für genauere Modelle. Auf der anderen Seite haben Entscheidungen in diesen frühen Entwurfsphasen einen großen Einfluss auf alle späteren Designentscheidungen. Wird zum Beispiel in der Synthese ein Gatter mit einer höheren Treiberfähigkeit ausgewählt, wirkt sich eine höhere kapazitive Last am Ausgang dieses Gatters weniger stark aus, als wenn ein Gatter mit geringerer Treiberfähigkeit ausgewählt wird. Diese Designentscheidungen in der Synthesephase werden meistens aufgrund der Modellierung mit Wire-Load-Modellen getroffen.

2.3.5.2 Rechenzeitbedarf

Während in den frühen Entwurfsphasen aufgrund des Fehlens von Informationen einfache Modelle eingesetzt werden, kommen einfache Modelle in späteren Designphasen aufgrund des geringeren Rechenzeitbedarfs zum Einsatz. Das Bounding-Box-Modell zur Bewertung einer Platzierung ist ein Beispiel für den Einsatz von einfachen Modellen. Prinzipiell könnte man ein genaueres Modell einsetzen, das beispielsweise die Belegung von Verdrahtungsressourcen oder Kopplungseffekte zwischen Leitungen berücksichtigt. Solche Modelle existieren bereits, erfordern aber ein Vielfaches an Rechenzeit im Vergleich zu einem Bounding-Box-Modell. Da in jedem Platzierungsschritt bei der Auswertung der Kostenfunktion eine Modellierung der Leitungen erfolgen muss, würde der Einsatz dieser Modelle die Laufzeit der Platzierung vervielfachen. Bisher wird der Kompromiss zwischen Genauigkeit und Rechenzeit im Backend-Design in nahezu allen Werkzeugen in Richtung geringe Genauigkeit und hohe Berechnungsgeschwindigkeit gewählt. Das damit auftretende Timing-Closure-Problem erfordert somit eine modifizierte Vorgehensweise. Im folgenden Abschnitt wird beispielhaft ein klassischer Designflow beschrieben, der von dem idealisierten Designflow gemäß 2.2 abweicht.

Im Rahmen dieser Arbeit wird zur Verzögerungszeitberechnung eine Kombination aus Bounding-Box-Modell und Elmore-Delay-Modell verwendet. Die Verdrahtungslängen aller Leitungen werden mit Hilfe von Bounding-Boxen bestimmt, während die Verzögerungszeiten entsprechend dem Elmore-Delay-Modell basierend auf den zuvor bestimmten Längen berechnet werden.

2.4 Klassischer Designflow

Bis zur Verdrahtung entspricht der klassische Designflow dem idealisierten Designflow. Nach der Verdrahtungsphase wird, wie in Abschnitt 2.2 beschrieben, eine statische Timing-Analyse durchgeführt. In vielen Fällen stellt man dabei fest, dass es Pfade gibt, in denen die spezifizierte Taktfrequenz nicht eingehalten werden kann. In diesem Fall hängt das weitere Vorgehen von der Anzahl der Constraint-Verletzungen ab. Bei einer geringen Anzahl von Constraint-Verletzungen haben sich Werkzeuge zur nachträglichen Layoutverbesserung etabliert, die durch geringfügige lokale Layoutänderungen die Verzögerungszeiten in einigen wenigen Pfaden reduzieren können. Eine detaillierte Übersicht über die zum Einsatz kommenden Verfahren befindet sich in Abschnitt 3.4. Bei einer größeren Anzahl von Pfaden mit Constraint-Verletzungen muss eine erneute Synthese durchgeführt werden und ein vollständig neues Layout erzeugt werden. Um ein verbessertes Syntheseergebnis zu erreichen, werden die Informationen aus dem bereits erzeugten Layout an die Netzliste annotiert und während der Resynthese verwendet. Dadurch werden im Technology-Mapping-Schritt, in dem die Abbildung der in der erste Phase der Synthese erzeugten generischen Netzliste auf die Zellbibliothek erfolgt, Gatter mit höherer Treiberfähigkeit selektiert, wenn diese mit einem potenziell langen Netz verbunden sind. Trotz dieser Iterationen gelingt es nicht bei allen Designs, eine Konvergenz zwischen dem Timing nach der Synthese und dem Timing nach der Platzierung bzw. Verdrahtung zu erreichen. In diesen Fällen setzt man zusätzlich ein Verfahren zur nachträglichen Layoutverbesserung ein.

Insgesamt ergibt sich damit der in Abbildung 2.10 dargestellte Designflow.

2.5 Neue Designflows

Aufgrund der in Kapitel 1 beschriebenen Probleme beim Einsatz eines klassischen Designflows bieten mehrere EDA-Anbieter einen Designflow an, in dem die einzelnen Werkzeuge nicht mehr voneinander unabhängig sind. Von diesen neuen Designflows sollen in den folgenden beiden Kapiteln die Ansätze von Magma Design Automation und Monterey Design vorgestellt werden.

2.5.1 Magmas „Blast Fusion“-Designflow

Der Blast-Fusion-Designflow von Magma [MAGMA02] beginnt nach der RTL-Synthese. Diese kann mit einem beliebigen Synthese-Werkzeug durchgeführt werden, wobei keine Abbildung der Schaltung auf die Zellen der Zieltechnologie erfolgen darf.

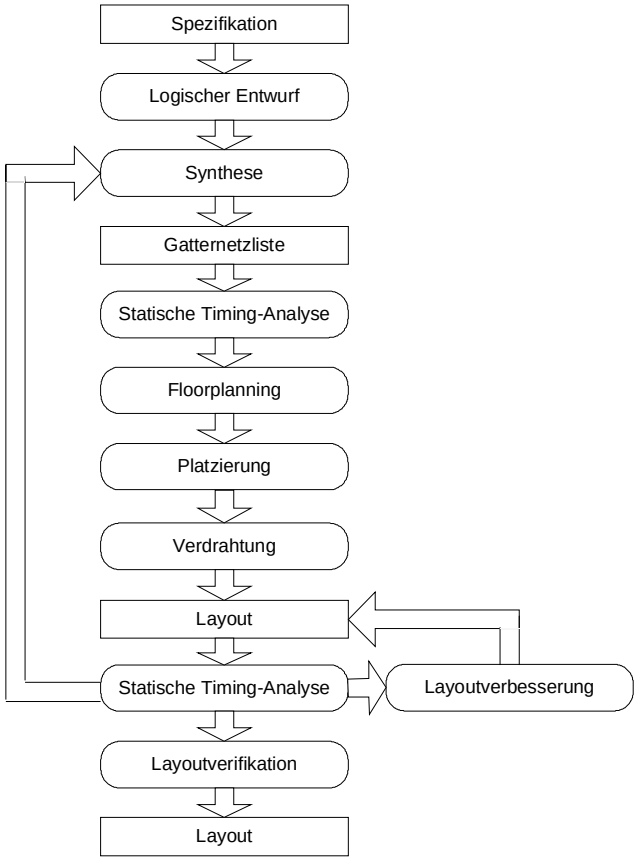

Abbildung 2.10: Exemplarischer klassischer Designflow

Der Blast-Fusion-Designflow umfasst die Schritte Platzierung, Verdrahtung, Clock-Tree-Erzeugung sowie Buffer-Insertion, Sizing (vgl. Abschnitt 3.4) und Extraktion. Diese Schritte sind nicht mehr voneinander unabhängig und erfolgen teilweise ineinander verschachtelt.

Kernpunkt des Magma-Werkzeugs sind so genannte Superzellen. Diese Zellen erhalten während der Synthese einen Verzögerungszeit-Wert, der sich im Laufe des weiteren Designflows nicht mehr verändern darf. Die Größe dieser Zellen ist im Gegensatz zu Standardzellen jedoch variabel. Eine Superzelle stellt dabei jeweils die Zusammenfassung logisch gleicher Zellen mit unterschiedlicher Treiberfähigkeit und damit unterschiedlicher Größe aus einer Standardzell-Bibliothek dar. Die Auswahl der Treiberfähigkeit einer Zelle wird damit nicht wie in einem konventionellen Designflow während des Technology Mappings durchgeführt, sondern erst im Backenddesign.

Nach der RTL-Synthese erfolgt als erster Schritt des Blast-Fusion-Designflows das Mapping der Zellen. Dabei wird die Netzliste jedoch nicht auf die Zieltechnologie, sondern auf die Superzellbibliothek abgebildet. Diese kann automatisch aus einer konventionellen Standardzell-Bibliothek erzeugt werden. In diesem Schritt erfolgt gleichzeitig mit der Abbildung der Zellen eine Logikoptimierung.

Nach der Logikoptimierung und der Abbildung der Zellen wird die Verzögerungszeit aller Superzellen berechnet und für den Rest des Designflows als fest definiert angenommen. Sollten bei der Verzögerungszeit-Berechnung Constraint-Verletzungen festgestellt werden, wird der Designflow abgebrochen, da Constraint-Verletzungen in diesem Entwurfsablauf nicht mehr in späteren Entwurfsschritten beseitigt werden können, und der Designer muss eine erneute Synthese durchführen. Anschließend erfolgt die Sizing-Driven-Platzierung. Unter Sizing-Driven-Platzierung wird in diesem Fall verstanden, dass gleichzeitig mit der Platzierung die Größe der Zellen festgelegt wird. Dabei erfolgt die Anpassung der Zellgröße derart, dass die Verzögerungszeit der Zelle konstant bleibt. Das bedeutet, dass in diesem Designflow bei der Platzierung die Treiberfähigkeit und damit die Größe einer Zelle erhöht werden kann, wenn eine hohe kapazitive Last am Ausgang einer Zelle bestimmt wird. Dabei kann die Last durch lange Leitungen oder eine große Anzahl getriebener Zellen bestehen. Gleichzeitig mit der Platzierung erfolgen das Einfügen von Buffern und die Erzeugung des Clock-Trees wie auch die Extraktion. Die in jedem Schritt inkrementell extrahierten Informationen werden in den jeweils folgenden Designschritten zur Platzierung verwendet. Am Ende der Platzierungsphase werden die Superzellen durch die Standardzellen mit entsprechender Treiberfähigkeit und damit auch entsprechender Größe ersetzt.

Im letzten Schritt des in Abbildung 2.11 dargestellten Magma Designflows erfolgt die Verdrahtung. Dabei wird eine Einhaltung der Timing-Constraints durch die Anpassung der Breite aller Leitungen wie auch durch das Einfügen von Abständen zwischen den Leitungen erreicht.

Nach der Fertigstellung des Layouts muss eine Verifikation des Layouts mit einem Layoutverifikationswerkzeug erfolgen.

2.5.2 Montereys „Sonar-Dolphin“-Designflow

Der “Sonar-Dolphin”-Designflow von Monterey [MONTEREY01] beginnt mit einer Designbeschreibung in einer Hochsprache und besteht aus den zwei Werkzeugen „Sonar“ und „Dolphin“.

„Sonar“ ist ein Synthesewerkzeug, dessen Ergebnis deutlich über das eines konventionellen Synthesetools hinausgeht. Am Ende der Synthese wird ein so genannter physikalischer Prototyp erzeugt. Dieser besteht aus einer Globalplatzierung und einer Globalverdrahtung, die basierend auf einem sehr groben Modell berechnet werden. Dieses Modell kann sehr schnell erzeugt werden und bietet Analysemöglichkeiten

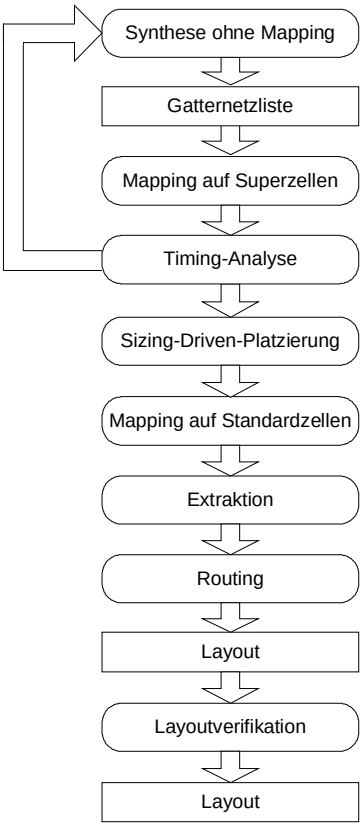

Abbildung 2.11: „Blast Fusion“-Designflow

bezüglich Timing, Chipgröße und Stromverbrauch. Treten bereits bei der Erstellung des Prototypen Timing-Probleme auf, so muss der Syntheseschritt wiederholt werden, ohne dass zur Feststellung der Constraint-Verletzungen das komplette Layout erzeugt werden muss. Erst wenn der Prototyp die Timing-Constraints erfüllt, wird mit der Layouterzeugung begonnen.

„Dolphin“ ist das Monterey-Werkzeug zur Layoutgenerierung. Dabei erhält „Dolphin“ neben der Netzliste auch die Informationen des physikalischen Prototypen. „Dolphin“ erzeugt die Platzierung und Verdrahtung gleichzeitig mit einer Logikoptimierung. Um diese Schritte in sinnvoller Laufzeit gleichzeitig erzeugen zu können, beginnt „Dolphin“ mit einem sehr groben Modell und führt in jedem Designschritt eine Modellverfeinerung durch.

„Dolphin“ gehört zu der Gruppe der Min-Cut basierten Platzierungsansätze. Es teilt die Layoutfläche im ersten Schritt in vier gleich große Teile und verteilt die Zellen anschließend auf die Teilflächen. Anschließend wird jede Layoutteilfläche wieder in vier Flächen geteilt. In jedem Schritt erfolgt nicht nur die Verteilung der Zellen auf die Teilflächen, was der Platzierung der Zellen entspricht, sondern auch die Verdrahtung und die Clock-Tree-Erzeugung. Dabei werden Verdrahtung und Clock-Tree jeweils nur auf der Ebene der Layoutteilflächen bestimmt und können dadurch in den ersten

Schritten sehr schnell berechnet werden. Die Genauigkeit von Platzierung, Verdrahtung und Clock-Tree-Berechnung nimmt mit der Anzahl der erzeugten Teilflächen kontinuierlich zu. Die nach jedem Verfeinerungsschritt gewonnenen Informationen über die Verdrahtung und den Clock-Tree werden in jedem Schritt zu einer Logikoptimierung genutzt. Die Layoutteilflächen werden so lange verfeinert, bis in jeder Teilfläche nur noch eine Zelle platziert ist. Dann sind sowohl Platzierung als auch Verdrahtung abgeschlossen. Der Ablauf des „Sonar-Dolphin“-Designflows ist in Abbildung 2.12 dargestellt.

Nach dem Einsatz von „Sonar“ und „Dolphin“ muss wie bei dem Ansatz von Magma noch die Layoutverifikation erfolgen.

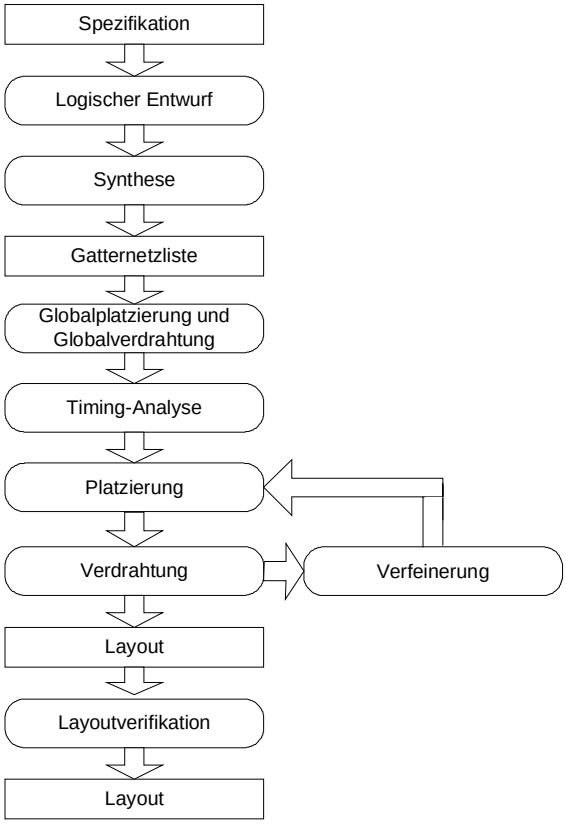

Abbildung 2.12: „Sonar-Dolphin“-Designflow

3 Platzierung und Verdrahtung

Nachdem im letzten Kapitel unterschiedliche Designflows beschrieben worden sind, werden in diesem Kapitel, Algorithmen für die Schritte Platzierung und Verdrahtung detailliert vorgestellt. Dies soll eine Einordnung der vorliegenden Arbeit ermöglichen, im Rahmen derer ein neues, leitbahnorientiertes Platzierungs- und Verdrahtungsverfahren entwickelt worden ist.

Das Problem der Platzierung und Verdrahtung von integrierten Schaltungen kann folgendermaßen formuliert werden:

Gegeben sind eine Menge von Zellen sowie eine Netzliste, die die Verbindungen zwischen den Zellen eindeutig beschreibt. Gesucht sind eine bezüglich einer Kostenfunktion optimale Platzierung der Bauelemente und der Verlauf der Verbindungen zwischen den Zellen. Bei der Ausführung von Platzierung und Verdrahtung sind technologiespezifische Randbedingungen einzuhalten. Das Layoutproblem ist mathematisch gesehen ein gemischt kombinatorisch kontinuierliches Optimierungsproblem mit Nebenbedingungen. Dieses ist nachgewiesenermaßen NP-vollständig [Brück93] und entzieht sich damit einer exakten Lösung in polynomialer Zeit mit deterministischen Verfahren.

Die Kostenfunktion des Layoutproblems kann unterschiedliche Parameter berücksichtigen. Diese sind sowohl vom Design als auch von der Technologie abhängig. In der Vergangenheit war das primäre Optimierungsziel die Minimierung der Gesamtfläche, weil die Fläche eines Layouts einen unmittelbaren Einfluss auf die Herstellungskosten hat:

Je kleiner die Fläche ist, desto geringer sind die Fertigungskosten. Zusätzlich ist die Ausbeute, die den Anteil funktionierender Chips beschreibt, bei kleinerer Fläche größer. Die Größe der Fläche kann in zwei unterschiedlichen Phasen während der Layouterzeugung beeinflusst werden. Zum einen bestimmt der Grad der Kompaktheit der Platzierung die erforderliche Fläche, zum anderen wird die Fläche bei Technologien mit Verdrahtungskanälen von der Verdrahtungsfläche bestimmt. Wenn man bei Digitalschaltungen davon ausgeht, dass alle Verbindungen mit Minimalbreite verdrahtet werden, kann die Verdrahtungsfläche direkt in die Verdrahtungslänge umgerechnet werden. Daher muss zur Optimierung der Chipfläche die Gesamtverdrahtungslänge minimiert werden.

Für aktuelle Technologien und Schaltungen mit hohen Taktfrequenzen werden bei neuartigen Platzierungswerkzeugen aufgrund des Timing-Closure-Problems zunehmend auch andere Faktoren wie beispielsweise Einhaltung von Signallaufzeiten (Timing-Constraints), Minimierung parasitärer Effekte und Minimierung der Anzahl von Durchkontaktierungen in der Kostenfunktion der Platzierung berücksichtigt. Die Gesamtverdrahtungslänge ist bei diesen Designflows trotzdem häufig noch Bestandteil

der Kostenfunktion. Dieses soll einen Beitrag zur Längenoptimierung aller Netze leisten und damit verhindern, dass durch zu lange Netze Timing-Probleme entstehen und gleichzeitig zu einer Flächenoptimierung führen.

3.1 Separates Platzieren und Verdrahten

Aufgrund der Komplexität des Layoutproblems wird das Layoutproblem in nahezu allen aktuellen Designflows in zwei Teilprobleme - Platzierung und Verdrahtung - unterteilt. Dieses Vorgehen reduziert die Komplexität des Layoutproblems, führt aber zu einigen Problemen: Das größte Problem ist die Bewertung der Platzierungsqualität. Diese wird durch verschiedene Kriterien wie beispielsweise Einhaltung von Timing-Randbedingungen, Gesamtverdrahtungslänge und Verdrahtbarkeit beschrieben. Um einige Kriterien der Platzierungsqualität bestimmen zu können, wird die Verdrahtung benötigt. Da diese bei der zweiphasigen Layouterzeugung noch nicht zur Verfügung steht, muss die Verdrahtung abgeschätzt werden. Diese Abschätzung bietet jedoch nur eine endliche Genauigkeit, die dazu führen kann, dass eine Platzierung, die nach Ausführung der Verdrahtung im Sinne der Kostenfunktion besser als eine andere Platzierung wäre, bereits vor der Verdrahtung verworfen wird.

In den folgenden Abschnitten sollen aktuell zum Einsatz kommende Algorithmen für die Platzierung und Verdrahtung von integrierten Schaltungen vorgestellt werden.

3.1.1. Platzieren

Bei der Platzierung handelt es sich um einen Syntheseschritt, bei dem die Komponenten einer strukturellen Beschreibung, die meist in Form einer Netzliste gegeben ist, in eine Anordnung auf der verfügbaren Layoutfläche umgesetzt werden sollen. Ziel aktueller Platzierungsverfahren ist es, die Zellpositionen derart zu berechnen, dass die der Platzierung folgende Verdrahtung unter Einhaltung aller maximal zulässigen Verzögerungszeiten erfolgen kann. Schon das Platzierungsproblem allein ist NP-vollständig [Brück93]. Bei den am weitesten verbreiteten Platzierungsverfahren handelt es sich um die kräftegesteuerte Platzierung, die in Abschnitt 3.1.1.1 vorgestellt wird, sowie um Platzierungsverfahren, die auf Simulated Annealing basieren. Diese Verfahren werden in Abschnitt 3.1.1.3 genauer erläutert. Die in den letzten Jahren immer häufiger zum Einsatz kommenden Verfahren, die auf dem Min-Cut-Algorithmus [Ou00] aufbauen, werden im Abschnitt 3.1.1.5 beschrieben. Die in der Praxis wenig verbreiteten und vorwiegend akademisch orientierten Ansätze, die auf Branch-and-Bound-Verfahren [Onodera91] basieren, werden aufgrund der sehr hohen Laufzeiten und der damit verbundenen Beschränkung auf kleine Schaltungen hier nicht weiter betrachtet.

3.1.1.1 Kräftegesteuerte Platzierung

Die kräftegesteuerte Platzierung [Eisenmann98, Antreich82, Kleinhans91] ist ein konstruktives Platzierungsverfahren, bei dem die Initialplatzierung iterativ verbessert wird. Dazu werden in jedem Iterationsschritt Kräfte auf alle Zellen berechnet und jede Zelle wird anschließend zur Optimierung der Gesamtverdrahtungslänge in Richtung der resultierenden Kraft verschoben. Ziel ist die Erreichung eines Kräftegleichgewichts, welches unter Zugrundelegung eines einfachen Federmodells dem energetisch günstigsten Zustand des Systems entspricht. Dieser Zustand entspricht im Allgemeinen einer kompakten und guten Platzierung [Brück93].

Die kräftegesteuerte Platzierung basiert in der Regel auf quadratischer Optimierung. Deshalb wird zur Berechnung der Kosten eines Netzes der quadrierte euklidische Abstand zwischen jeweils zwei Zellen verwendet. Damit ergeben sich für zwei miteinander verbundene Zellen i und j an den Positionen (x_i, y_i) bzw. (x_j, y_j) folgende Verbindungskosten:

$$c_{ij} = w_k \cdot \left[\left(x_i - x_j \right)^2 + \left(y_i - y_j \right)^2 \right] \qquad \textbf{(3.1)}$$

Der Konstanten w_k aus Gleichung (3.1) kann für jedes Netz ein unterschiedlicher Wert zugewiesen werden. Je größer der Wert von w_k ist, desto höher werden die Kosten für eine Verbindung. Dadurch wird beim Lösen des Optimierungsproblems eine Verbindung mit einem großen w_k in der Regel stärker verkürzt als eine Verbindung mit einem kleineren w_k. Im Folgenden wird w_k deshalb auch als Netzgewicht bezeichnet.

Formuliert man die Kostenfunktion für alle Verbindungen einer Schaltung in Matrixform, so ergeben sich die Gesamtkosten für eine Platzierung zu

$$Cost(\underline{p}) = \frac{1}{2} \underline{p}^T \cdot \underline{C} \cdot \underline{p} + \underline{d}^T \cdot \underline{p} + const. \qquad \textbf{(3.2)}$$

Dabei beinhaltet $\underline{p}$ die Koordinaten aller Zellen, also $\underline{p}^T = [x_1, ..., x_n, y_1, .., y_n]$. Die $2n \times 2n$ große Matrix $\underline{C}$ enthält alle Netzgewichte. Die Berechnung der Komponenten der Matrix $\underline{C}$ erfolgt für x- und y-Komponenten getrennt.

Betrachtet man den x-abhängigen Teil der Gleichung (3.1), ergibt sich

$$c_{ij,x} = w_k \cdot x_i^{\,2} - 2 \cdot w_k \cdot x_i \cdot y_j + w_k \cdot x_j^{\,2} \quad . \qquad \textbf{(3.3)}$$

Für Netze ohne feste Zellen führt dies zu einem Beitrag von w_k an den Positionen (i,i) und (j,j) sowie zu einem Beitrag von $-w_k$ an den Positionen (i,j) und (j,i) in der Matrix $\underline{C}$. Für Netze, die eine feste Zelle wie z.B. eine Padzelle beinhalten, ergibt sich nur ein Eintrag in der Matrix $\underline{C}$. Unter der Annahme, dass die feste Zelle j ist, wird w_k an der Position (i,i) in die Matrix eingetragen. Der verbleibende lineare Term der Gleichung (3.3) von $-2w_k \cdot x_j$ wird an der i. Stelle von $\underline{d}$ eingetragen, während der konstante Anteil der Gleichung (3.3) in dem konstanten Term von Gleichung (3.2) berücksichtigt wird.

Die Berechnung der y-abhängigen Komponenten der Matrix $\underline{C}$ erfolgt analog.

Das gesuchte Minimum der Kostenfunktion entsprechend der minimalen gewichteten Gesamtverdrahtungslänge ergibt sich, wenn die erste Ableitung der Kostenfunktion null ist. Diese Bedingung führt auf die lineare Gleichung

$$\underline{C} \cdot \underline{p} + \underline{d} = 0 \ . \qquad \textbf{(3.4)}$$

Diese Gleichung lässt sich physikalisch mit einem Federmodell interpretieren, wenn man jedes Netz durch eine Feder ersetzt. Das Produkt aus $\underline{C} \cdot \underline{p}$ entspricht den anziehenden Federkräften zwischen den Zellen $\vec{F} = k \cdot (\vec{x}_i - \vec{x}_j)$ entsprechend dem Hookschen Gesetz, wobei der Federkonstanten k das Netzgewicht w_k entspricht. $\underline{d}$ beinhaltet die Positionen der festen Zellen. Löst man Gleichung (3.4), nimmt das Federsystem seinen energetisch günstigsten Zustand an.

Einige Implementierungen der kräftegesteuerten Platzierung verwenden auch eine lineare Kostenfunktion zur Berechnung der Verbindungskosten. Einen Vergleich der Vor- und Nachteile von linearen und quadratischen Kostenfunktionen findet man in [Sigl91].

Gleichung (3.4) berücksichtigt weder Überlappungen noch die maximal zur Verfügung stehende Layoutfläche. Dadurch entstehen viele Überlappungen zwischen Zellen. Zur Vermeidung von Überlappungen führt man deshalb eine zusätzliche abstoßende Kraft ein, so dass eine anziehende und eine abstoßende Kraft auf jede Zelle wirken. Anschaulich modellieren die anziehenden Kräfte $\vec{F}_{A,i}$ die Verbindungen zwischen den Zellen, während die abstoßenden Kräfte $\vec{F}_{R,i}$ die Belegung der Layoutfläche repräsentieren. Die Richtung der Kräfte für eine Drei-Zellen-Anordnung kann der Abbildung 3.1 entnommen werden. Die Richtung der abstoßenden Kräfte $\vec{F}_{R,i}$ ergibt sich dabei durch die in der Abbildung 3.1 nicht dargestellte Belegung der Layoutfläche.

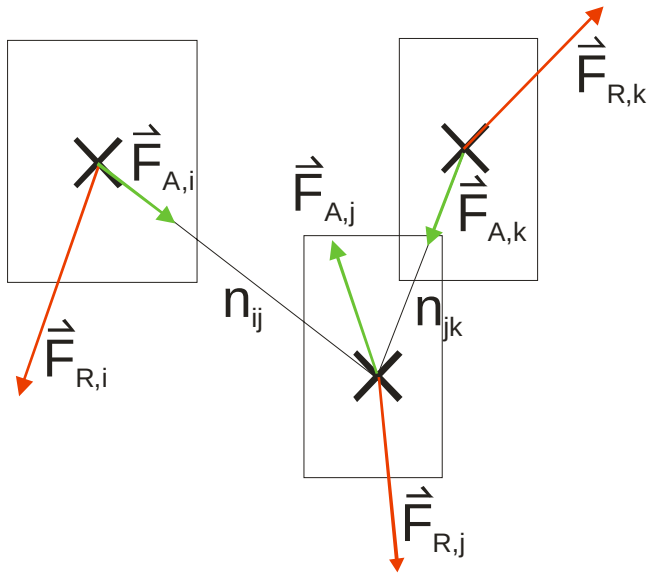

Abbildung 3.1: Kräftegesteuertes Platzieren

Die abstoßenden Kräfte sollen in Gleichung (3.5) durch eine zusätzliche einspaltige Matrix $\underline{e}$ repräsentiert werden:

$$\underline{C} \cdot \underline{p} + \underline{d} + \underline{e} = 0 \qquad \textbf{(3.5)}$$

Die durch die Matrix $\underline{e} = (e_1, e_2, ..., e_{2n})$ repräsentierte Kraft soll für eine gleichmäßigere Verteilung der Zellen über der Layoutfläche sorgen und soll folgende Eigenschaften haben:

- Die Kraft hängt nur von der Position der Zelle ab, d.h. zwei Zellen an der gleichen Position erfahren die gleiche abstoßende Kraft.
- Bereiche mit höherer Zelldichte sind Startpunkte der Kräfte, d.h. Zellen werden von Bereichen mit höherer Zelldichte in Bereiche mit geringer Belegung verschoben.
- Im Unendlichen soll die Kraft null sein.

In Analogie zur Elektrostatik ergibt sich eine mögliche Berechnung des abstoßenden Kraftbeitrags für die Zelle i im m-ten Platzierungsschritt nach [Eisenmann99] zu

$$\vec{f}_{R,m}(x_i, y_i) = \frac{k}{2\pi} \int_{x'=-\infty}^{\infty} \int_{y'=-\infty}^{\infty} D(x', y') \frac{\vec{r}_i - \vec{r}\,'}{|\vec{r}_i - \vec{r}\,'|^2} dx'dy' \;, \qquad \textbf{(3.6)}$$

wobei k eine Konstante zur Skalierung der Kraft, D(x', y') die Dichtefunktion (vgl. Abbildung 3.3) an der jeweiligen Position und $\vec{r}_i = (x_i, y_i)$ sowie $\vec{r}\,' = (x', y')$ ist.

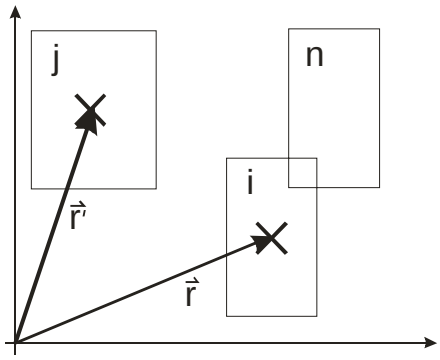

Abbildung 3.2: Berechnung der abstoßenden Kraft auf Zelle i

In Abbildung 3.2 sind die Komponenten der Gleichung (3.6) visualisiert. Dabei beschreibt der Vektor $\vec{r}_i$ die Zelle i, auf die die durch alle anderen Zellen verursachte abstoßende Kraft berechnet werden soll.

Die resultierende abstoßende Kraft im m-ten Platzierungsschritt $F_{R,m}$ wird aus der abstoßenden resultierenden Kraft des vorangegangenen Schritts und dem Kraftbeitrag des m-ten Iterationsschritts $f_{R,m}$ zu

$$\vec{F}_{R,m} = \vec{F}_{R,m-1} + \vec{f}_{R,m} \tag{3.7}$$

berechnet. Die Berücksichtigung der abstoßenden Kräfte aus vorangegangenen Platzierungsschritten soll sprunghafte Änderungen in den abstoßenden Kräften vermeiden und so zur Stabilität der Gleichung (3.5) beitragen.

Die Matrix der resultierenden abstoßenden Kraft $\underline{e}$ wird wie folgt berechnet:

$$\underline{e}_{i,m} = \vec{F}_{r,m}(x_i, y_i) \tag{3.8}$$

Die Dichtefunktion $D(x, y)$ aus Gleichung (3.6) berechnet sich zu

$$D(x,y) = \sum_i a_i(x,y) - s \cdot A(x,y) \tag{3.9}$$

mit

$$A(x,y) = R(\frac{x}{W}) \cdot R(\frac{y}{H}) \;, \quad s = \frac{\sum w_i \cdot h_i}{W \cdot H} \tag{3.10}$$

sowie

$$a_i(x,y) = R(\frac{x - x_i}{w_i}) \cdot R(\frac{y - y_i}{h_i}) \tag{3.11}$$

und

$$R(z) = \begin{cases} 1, \ falls -\frac{1}{2} < z < \frac{1}{2} \\ 0, \ sonst \end{cases} . \qquad \textbf{(3.12)}$$

Anschaulich entspricht die Zelldichte an einem Punkt der Layoutfläche der Anzahl der Zellen, die diesen Punkt überdecken, vermindert um die so genannte Belegung des Platzierungsgebiets. Die Belegung s beschreibt dabei das Verhältnis der Summe aller Zellflächen zur Gesamtlayoutfläche.

Abbildung 3.3 zeigt exemplarisch den Verlauf der Dichtefunktion für zwei Zellen entlang der Linie y=0.

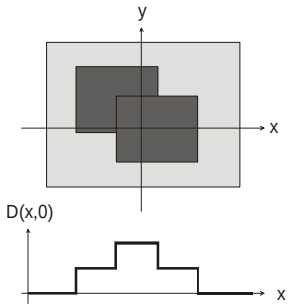

Abbildung 3.3: Verlauf der Dichtefunktion

Der Algorithmus zur kräftegesteuerten Platzierung verläuft in drei Schritten. In der Initialisierungsphase ist $\underline{e} = \vec{0}$ und Gleichung (3.4) wird gelöst. Die entstehende Initialplatzierung hat in der Regel eine große Anzahl von überlappenden Zellen. Anschließend wird die so berechnete Platzierung durch wiederholtes Lösen der Gleichung (3.5) iterativ verbessert, wobei für jeden Iterationsschritt die abstoßenden Kräfte $\vec{F}_{R,m}$ gemäß Gleichung (3.6) und die Kraftmatrix $\underline{e}$ neu bestimmt werden müssen. In den meisten Implementierungen wird die Platzierung solange iterativ verbessert, bis ein Abbruchkriterium erfüllt ist. Als ein solches Abbruchkriterium eignet sich zum Beispiel der Anteil der Freiflächen im Platzierungsbereich oder eine Gleichgewichtsbedingung. Die mit einem kräftegesteuerten Verfahren erzeugte Platzierung kann auch nach einer größeren Anzahl von Iterationen noch einige Überlappungen zwischen Zellen beinhalten. Deshalb erfolgt als letzter Schritt der kräftegesteuerten Platzierung ein so genannter Legalisierungsschritt. In diesem Schritt werden alle Überlappungen durch möglichst geringfügige Verschiebungen der Zellen beseitigt. Gleichzeitig kann ggf. die durch den Entwurfsstil erforderliche Anordnung der Zellen z.B. in Reihen erzeugt werden.

In der Standardimplementierung der kräftegesteuerten Platzierung erfolgt die Berechnung der anziehenden und abstoßenden Kräfte mit einer Laufzeitkomplexität von $O(n \cdot \log_2 n)$ [Kleinhans88], wobei n die Anzahl der Zellen ist. Geht man davon aus, dass die Lösung des linearen Gleichungssystems mit einem iterativen Gleichungslö-

sungsverfahren mit einer Laufzeitkomplexität von $O(n^{1,5})$ [Spiro90] erfolgen kann, hat die kräftegesteuerte Platzierung eine Laufzeitkomplexität von $O(n^{1,5})$. Damit ist dieses Verfahren eines der schnellsten Verfahren zur Zellplatzierung. Ein Nachteil ist die erforderliche Legalisierung. Dabei kann es bei vielen Überlappungen zu deutlichen Veränderungen der Platzierung kommen, was den Wert der Kostenfunktion nach Abschluss der eigentlichen Platzierung stark beeinflussen kann. Der entscheidende Nachteil dieses Verfahrens liegt jedoch in der fehlenden Möglichkeit, Timing-Restriktionen berücksichtigen zu können. Die einzige Möglichkeit, eine Gleichbehandlung aller Zellen bzw. Netze in diesem Ansatz zu verhindern, besteht in der Erhöhung der Netzgewichte w_k von kritischen Netzen, was einer höheren anziehenden Kraft zwischen den beteiligten Zellen entspricht. Da die abstoßende Kraft vom Netzgewicht nicht beeinflusst wird, werden diese - potenziell kritischen - Zellen stärker aufeinander zu bewegt.

In den meisten aktuellen Standardzell-Designs ist aufgrund der hohen Taktfrequenzen ein Großteil der Pfade laufzeitkritisch. Unterschiedliche Netzgewichte bieten daher nur eine begrenzte Möglichkeit, die Platzierung zu beeinflussen, da eine große Anzahl von Netzen das Netzgewicht für laufzeitkritische Netze zugewiesen bekommen würde. Deshalb eignet sich die Standardimplementierung der kräftegesteuerten Platzierung nicht gut für die Platzierung von Designs mit vielen laufzeitkritischen Pfaden. In den letzten Jahren sind deshalb verschiedene Modifikationen der kräftegesteuerten Platzierung entwickelt worden, die das Laufzeitverhalten der Schaltungen stärker berücksichtigen.

3.1.1.2 Modifizierte kräftegesteuerte Platzierung

Es gibt viele Versuche, die kräftegesteuerte Platzierung in einen Timing-Driven-Designflow zu integrieren. Die wichtigsten Modifikationen werden im Folgenden genauer beschrieben.

In [Eisenmann98] werden die Verbindungsgewichte w_k aus Gleichung (3.1) für jedes Netz iterativ in jedem Platzierungsschritt neu berechnet. Durch diese dynamischen Anpassungen werden in jedem Platzierungsschritt die Zellen, die zu Netzen mit einer geringen Differenz aus zulässiger Verzögerungszeit und aktuell geschätzter Verzögerungszeit gehören, mit höheren anziehenden Kräften beaufschlagt als die Zellen, die zu unkritischen Netzen gehören.

Dazu werden zunächst alle geschätzten Pfadlängen auf der Basis eines Bounding-Box-Modells bestimmt und die maximal zulässige Verzögerungszeit für die Netze jedes Pfads berechnet. Aus der Differenz der zulässigen Verzögerungszeit und der aus der Pfadlänge geschätzten Verzögerungszeit werden in diesem Ansatz die 3% Pfade bestimmt, die am laufzeitkritischsten sind. Diese Pfade erhalten für den folgenden

Platzierungsschritt Netzgewichte, die höher als die Netzgewichte unkritischer Pfade sind. Zur Berechnung dieser Netzgewichte wird ein Wert c eingeführt, der zunächst für alle Netze auf null initialisiert wird. Im m-ten Iterationsschritt erhält c für jedes laufzeitkritische Netz den Wert $(c_j^{(m-1)}+1)/2$, während c für alle anderen Netze den Wert $c_j^{(m-1)}/2$ erhält. Die Netzgewichte w_k werden im ersten Iterationsschritt auf eins initialisiert und im m-ten Iterationsschritt mit Hilfe von c für das j-te Netz zu $w_j^{(m)} = w_j^{(m-1)} \cdot (1+c_j^{(m)})$ bestimmt.

Dieses Verfahren zur Bestimmung der Pfadgewichte führt teilweise zu einer sehr deutlichen Verkürzung der kritischen Pfade. Im Vergleich zu TimberWolf aus [Swartz95] ergibt sich für einige Benchmark-Schaltungen eine Verkürzung auf 10% der Länge des kritischsten Pfads. Die Laufzeit steigt durch die iterative Berechnung der Netzgewichte aufgrund der einfachen Berechnungsvorschrift nur geringfügig an. Entscheidender Nachteil dieses Vorgehens ist die strikte Trennung von laufzeitkritischen und laufzeitunkritischen Netzen. Solange nur eine geringe Anzahl von Pfaden laufzeitkritisch ist, liefert dieses Verfahren gute Platzierungsergebnisse. Wenn - wie bei vielen aktuellen Designs - ein größerer Anteil der Netze laufzeitkritisch ist, wird ein großer Anteil der Netzgewichte nach der Berechnungsvorschrift für zeitkritische Netze bestimmt und die bei einem kleinen Anteil zeitkritischer Netze beobachtete starke Verkürzung kritischer Pfade wird nicht mehr erreicht.

In [Kong02] wird ebenfalls ein auf Pfad-Verzögerungszeiten basierender Algorithmus zur Berechnung von Netzgewichten vorgestellt.

Ein weiterer Ansatz zur stärkeren Einbindung von Timing-Randbedingungen in die kräftegesteuerte Platzierung basiert auf dem Einfügen von Pseudoverbindungen [Chou01, Chou02].

Um die Länge eines Pfads steuern zu können, wird in diesem Ansatz - wie in Abbildung 3.4 dargestellt - zwischen den Flipflops des Pfads ein so genannter Pseudo-Link eingefügt.

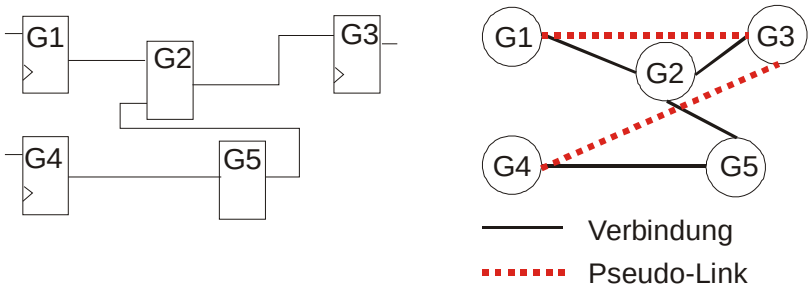

Abbildung 3.4: Graph mit Pseudo-Links

Dieser repräsentiert eine virtuelle Verbindung zwischen den beiden Elementen. Diese rein virtuelle Verbindung erhält nun - vergleichbar zu einer realen Verbindung - Verbindungskosten zugewiesen. Diese hängen von der Länge des Pfads und der Anzahl der Zellen im Pfad ab. Die Zellanzahl des jeweiligen Pfads wird in Gleichung (3.13) durch $Length(Path_{ij})$ beschrieben. Konkret werden die Verbindungskosten einer virtuellen Verbindung zu

$$c_{ij} = \left[\left(x_i - x_j \right)^2 + \left(y_i - y_j \right)^2 \right] (Length(Path_{ij}))^{\alpha} \qquad \textbf{(3.13)}$$

bestimmt, wobei α ein Benutzerparameter zur Gewichtung der Verbindungskosten zwischen einer virtuellen zu einer realen Verbindung ist und in [Chou01] aus dem Intervall [2..2,5] gewählt wird. Die Kosten einer realen Verbindung werden analog zu Gleichung (3.1) zu

$$c_{ij} = \left(x_i - x_j \right)^2 + \left(y_i - y_j \right)^2 \qquad \textbf{(3.14)}$$

berechnet. Damit hängen die Verbindungskosten der Pseudo-Verbindung nicht nur von dem geometrischen Abstand der beiden Zellen, sondern auch von der Länge des dazugehörigen Pfads ab. Dieses führt zu einer zusätzlichen anziehenden Kraft, die die beiden Flipflops aufeinander zu bewegt.

Dieses Vorgehen führt bei [Chou01] zu einer mittleren Verkürzung der Pfade im Vergleich zu einem kommerziellen Platzierungswerkzeug ohne Platzierungsoptimierung um 21%. Bei dem Einsatz eines kommerziellen Platzierers in Verbindung mit In-Place-Optimization (vgl. Abschnitt 3.4) führt es zu einer Verbesserung um 4%. Dabei ist die Laufzeit im Vergleich zu einem kommerziellen Platzierungsverfahren mit Optimierungsalgorithmus kürzer.

Aufgrund des starken Einflusses sehr langer Pfade auf die Netzgewichte durch die polynomiale Berücksichtigung der Zellanzahl im Pfad mit dem Faktor α mit $\alpha \geq 2$ in Gleichung (3.13) wird die Verkürzung der durchschnittlichen Pfadlänge vorwiegend durch eine Verkürzung dieser Pfade erreicht. Pfade mit einer geringen Überschreitung der Timing-Restriktionen werden durch dieses Vorgehen nicht in jedem Fall verkürzt. Außerdem kann dieses Verfahren nicht die Einhaltung aller erforderlichen maximalen Pfadlängen sicherstellen, sondern nur eine Verkürzung vieler längerer Pfade erreichen.

Der Ansatz von [Mo01a] sieht die Ursache für die schlechte Eignung eines kräftegesteuerten Platzierers als Timing-Driven-Platzierer in der ungenauen Berechnung der anziehenden Kräfte. Deshalb werden die anziehenden Kräfte in diesem Ansatz in Abhängigkeit von den geschätzten Netz-Verzögerungszeiten berechnet.

Die Verzögerungszeit zwischen jeweils zwei Zellen besteht aus der konstanten Verzögerung der treibenden Zelle und einem lastabhängigen Anteil, der proportional zu den getriebenen Kapazitäten ist. Zunächst werden in diesem Ansatz alle Mehrpunktnetze mit einem Sternpunkt in mehrere Netzteile zerlegt. Anschließend werden Widerstand und Kapazität und daraus resultierende Verzögerungszeit für jedes Teilnetz bestimmt. Ausgehend von diesen Verzögerungszeiten wird eine Kraft berechnet, die anziehend in Richtung der Verbindung wirkt.

Dieser Ansatz führt im Vergleich zur Standardimplementierung der kräftegesteuerten Platzierung zu weniger Pfaden mit Constraint-Verletzungen, aber zu einem deutlichen Anstieg der Laufzeit. Im Vergleich zu kommerziellen Platzierungswerkzeugen ergibt sich teilweise ein Anstieg der Laufzeit um den Faktor zehn.

Das größte Problem in diesem Ansatz ist die Berechnung der Verzögerungszeiten. Sowohl die Position des Sternpunkts als auch der Verlauf der Leitungen können während der Platzierungsphase nur abgeschätzt werden. Deshalb ist die sehr genaue Berechnung durch die Einteilung in eine lastabhängige und eine lastunabhängige Verzögerungszeit nicht sicher, sondern auch nur eine Abschätzung. Die Platzierungsergebnisse von [Mo01a], bei denen die Anzahl der Pfade mit Constraint-Verletzungen vor und nach dem Routing deutlich differiert, weisen genau auf dieses Problem hin. Deshalb ist der hohe Rechenaufwand basierend auf einer ungenauen Leitungsabschätzung nicht unbedingt sinnvoll.

Alle Modifikationen der kräftegesteuerten Platzierung führen in bestimmten Teilbereichen der Platzierung zu verbesserten Ergebnissen. Der erste Ansatz verbessert das Zeitverhalten der 3% kritischsten Pfade durch eine separate Behandlung dieser Pfade. Der zweite Ansatz reduziert die Pfadlänge durch Einführung einer Pseudokraft zwischen Flipflops und der dritte Ansatz kann die Pfadlänge durch genauere Bestimmung der Gatter-Verzögerungszeiten reduzieren.

Keine dieser Modifikationen ist jedoch in der Lage, eine Garantie für die Erstellung eines funktionsfähigen Layouts zu geben. Im Allgemeinen wird das Platzierungsergebnis besser als das mit der Standardimplementierung erzielte Ergebnis sein, aber es wird trotzdem Platzierungen geben, die nicht unter Einhaltung alle Restriktionen verdrahtbar sind.

3.1.1.3 Simulated Annealing

Simulated Annealing ist ein naturanaloges, iteratives, zufallsgesteuertes Platzierungsverfahren. Als Initialplatzierung kann eine beliebig erzeugte Zellanordnung verwendet werden. In jedem Iterationsschritt werden Änderungen an der Platzierung vorgenom-

men wie beispielsweise eine paarweise Zellvertauschung. Anschließend wird mit Hilfe einer simulierten Temperaturkurve und einer Zufallsfunktion entschieden, welche Platzierungsmodifikationen, die die Kostenfunktion verschlechtern, akzeptiert werden. Durch die Akzeptanz von Verschlechterungen der Kosten, können lokale Minima der Kostenfunktion verlassen werden und so das globale Optimum gefunden werden. Platzierungsänderungen, die das Ergebnis der Kostenfunktion verbessern, werden in jedem Fall akzeptiert.

In der Standardimplementierung [Sechen98] werden ausgehend von einer Initialplatzierung in jedem Iterationsschritt zwei zufällig ausgewählte Zellen paarweise vertauscht bzw. eine Zelle an einen freien Platz verschoben. Anschließend wird der Wert der Kostenfunktion für die modifizierte Platzierung bestimmt. Als Kostenfunktion wird in den meisten Fällen die Gesamtverdrahtungslänge verwendet, wobei prinzipiell auch beliebige andere Funktionen eingesetzt werden können. Dabei eignen sich Kostenfunktionen besonders gut, die nach einem Vertauschungsschritt inkrementell aktualisiert werden können und nicht vollständig neu berechnet werden müssen. Eine vollständige Neuberechnung der Kostenfunktion würde in diesem Verfahren aufgrund der häufigen Auswertung der Kostenfunktion zu unvertretbar hohen Rechenzeiten führen.

Sind die Kosten nach einem Vertauschungsschritt geringer geworden, wird der vollzogene Schritt akzeptiert und der nächste wird durchgeführt. Für die Behandlung von Vertauschungen, die die Kosten erhöhen, wird gleichzeitig mit der Platzierung eine Abkühlkurve simuliert. Diese beschreibt das Temperaturverhalten eines Körpers, der erwärmt wurde und anschließend abgekühlt wird und verläuft gewöhnlich exponentiell. In [Kirkpatrick83] wird die Berechnungsvorschrift der Abkühlkurve zu

$$T_n = \alpha(T) \cdot T_{n-1} \tag{3.15}$$

mit

$$0 < \alpha(T) < 1 \tag{3.16}$$

bestimmt.

Daraus ergibt sich der in der folgenden Abbildung dargestellte Verlauf der Abkühlkurve, der teilweise auch als Abkühlplan bezeichnet wird.

Die Temperatur wird entsprechend dem Abkühlplan jeweils beim Erreichen eines Gleichgewichts abgesenkt. Ein Gleichgewicht gilt als erreicht, wenn sich die Kostenfunktion über eine festgelegte Anzahl von Vertauschungsschritten nicht verändert hat.

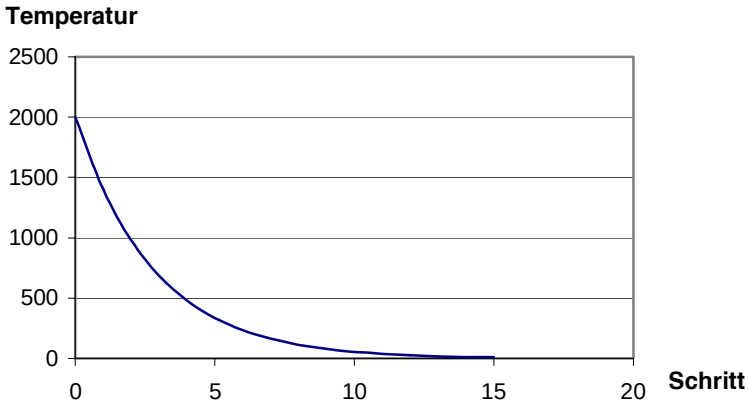

Abbildung 3.5: Abkühlkurve

Verschlechtern sich die Kosten nach einem Vertauschungsschritt, wird mit Hilfe einer Akzeptanzfunktion und einer Zufallsfunktion ermittelt, ob der Vertauschungsschritt trotz der Erhöhung der Kosten akzeptiert wird. Die Akzeptanzfunktion wird in [Kirkpatrick83] zu

$$f_{Akzept}(\Delta Kosten, T) = \min(1.0, e^{-\frac{\Delta Kosten}{T}}) \qquad \textbf{(3.17)}$$

mit

$$\Delta Kosten = Kosten(Schritt\ i) - Kosten(Schritt\ i-1) \qquad \textbf{(3.18)}$$

bestimmt.

$\Delta Kosten$ hat einen positiven Wert, wenn sich die Kosten verschlechtern. Dadurch wird der Wert der Akzeptanzfunktion (3.17) bei gleicher Kostenänderung umso kleiner, je größer die Temperatur T der simulierten Abkühlkurve ist. Die Akzeptanzentscheidung erfolgt durch den Vergleich des Wertes der Akzeptanzfunktion mit einem Zufallswert aus dem Intervall $[0, 1]$. Ist der Wert der Zufallsfunktion kleiner als der Wert von f_{Akzept}, wird der neue Platzierungsschritt trotz der Verschlechterung der Kosten akzeptiert, im umgekehrten Fall hingegen nicht. Anschaulich gesehen sinkt also die Akzeptanzwahrscheinlichkeit einer Kostenverschlechterung mit sinkender Temperatur.

Insgesamt ergibt sich damit folgender Ablauf, wobei T die Temperatur und P die Platzierung beschreiben:

```
Simulated Annealing
  T = T0;
  P = P0;
  while (Abbruchkriterium nicht erfüllt)
    while (Gleichgewicht nicht erreicht)
      P_neu = VeränderePlatzierung(P);
      if (AkzeptierePlatzierung(P, P_neu, T))
        P = P_neu;
      end
    end
    T = α T
  end
```

Durch die Akzeptanz von Verschlechterungen bezüglich der Kostenfunktion ermöglicht Simulated Annealing das Verlassen lokaler Minima. Gleichzeitig wurde nachgewiesen [Brück93], dass durch geschickte Steuerung des Temperaturverlaufs und Wahl der Gleichgewichtsbedingung das Verlassen des globalen Minimums verhindert werden kann.

Simulated Annealing besitzt zwei entscheidende Nachteile: Zum einen hängen die Platzierungsergebnisse stark von der Wahl des Abkühlplans ab. Die Wahl der Parameter des Abkühlplans erfordert jedoch einige Erfahrung und unter Umständen auch einige zeitaufwendige Testläufe. Zum anderen berücksichtigt die Standardimplementierung von Simulated Annealing keine Timing-Informationen. Während die Standardimplementierung der kräftegesteuerten Platzierung eine Steuerung mit Hilfe der Netzgewichte ermöglicht, gibt es beim Simulated Annealing keine vergleichbaren Steuerungsmöglichkeiten. Dies hat zur Konsequenz, dass alle Zellen bzw. Netze gleich behandelt werden. Eine bevorzugte Behandlung von laufzeitkritischen Netzen kann jedoch durch eine einfache Modifikation der Kostenfunktion (vgl. Abschnitt 3.1.1.4) erreicht werden.

3.1.1.4 Modifiziertes Simulated Annealing

Der Ansatz von [Swartz95] bietet eine Erweiterung eines Simulated-Annealing-Platzierers für einen Timing-Driven-Designflow durch Berücksichtigung von Pfad-Constraints. In diesem Ansatz können für beliebig viele Pfade Timing-Constraints angegeben werden, die in der Platzierungsphase berücksichtigt werden. Die Verzögerungszeit innerhalb eines Pfades a wird in diesem Ansatz zu

$$T_a(p) = \sum D_n \qquad \textbf{(3.19)}$$

bestimmt. Dabei berechnet sich die Verzögerungszeit D_n gemäß Gleichung (3.20) als Summe aus der Eigenverzögerung des treibenden Gatters und dem Produkt aus Ausgangswiderstand und kapazitiver Last.

$$D_n = T_n + R_n \cdot C_n \tag{3.20}$$

Die Lastkapazität C_n besteht aus der Eingangskapazität der Folgegatter sowie der parasitären Kapazität durch Leitungen. Die berechnete Pfadverzögerung T_a wird mit der Taktperiode des Designs verglichen. Für alle Pfade, bei denen die Pfadverzögerung die Taktperiode übersteigt, wird ein Strafterm P_i zur Kostenfunktion des Simulated Annealings hinzugefügt, für alle anderen Pfade ist P_i null. Die Kostenfunktion f_C besteht aus der Gesamtverdrahtungslänge W und der Summe der Strafterme P_T

$$f_C = W + \lambda \cdot P_T \ , \tag{3.21}$$

wobei sich die Summe der Strafterme zu

$$P_T = \sum P_i \tag{3.22}$$

berechnet.

Entscheidend für die Erzeugung guter Layouts ist der Parameter λ, der die Gewichtung der Strafterme der Pfade mit Constraint-Verletzungen bestimmt. Wird λ zu groß gewählt, so wird die Gesamtverdrahtungslänge zu groß, ist λ zu klein, werden nicht alle Constraints eingehalten.

Diese Kostenfunktion kann in einen Standard-Platzierer auf Simulated Annealing-Basis integriert werden und ermöglicht ein Timing-Driven-Placement.

Die Ergebnisse eines solchen Timing-Driven-Placement-Verfahrens verkürzen in [Swartz95] den längsten Pfad um 20-50%, die Gesamtverdrahtungslänge steigt daher nur geringfügig an.

Die geschilderte Modifikation von Simulated Annealing ermöglicht den Einsatz einer Kostenfunktion zur Berücksichtigung von Pfad-Constraint-Verletzungen für Platzierungen von laufzeitkritischen Designs. Allerdings können trotz der deutlichen Verkürzung des kritischen Pfads bei einer größeren Anzahl kritischer Pfade weiterhin Pfade mit Verletzung der Pfad-Constraints auftreten. Damit kann dieser Algorithmus nicht für einen Correct-by-Construction-Ansatz eingesetzt werden.

3.1.1.5 Min-Cut-Platzierung

Die Min-Cut-Platzierung ist ein konstruktives Top-Down-Platzierungsverfahren. Es basiert auf der rekursiven Teilung der Netzliste und der Layoutfläche in jeweils zwei Teile. In jedem Iterationsschritt wird nach der Teilung ein Netzlistenteil einer Layoutteilfläche zugewiesen. Die rekursive Aufteilung von Netzliste und Layoutfläche wird so lange fortgesetzt, bis jede Layoutteilfläche nur noch ein Element beinhaltet und die Platzierung damit definiert ist. Damit ergibt sich folgender Platzierungs-Ablauf:

Min-Cut-Platzierung
While (Netzliste enthält mehr als ein Element)
Teile die Layoutfläche in zwei Teile
Partitioniere die Netzliste in zwei Teile
Weise jedem Layoutflächenteil eine Netzlistenpartition zu
Wende das Verfahren rekursiv auf jede der Teilflächen an
end

Die Partitionierung der Netzliste erfolgt im idealen Fall derart, dass die Anzahl der geschnittenen Netze minimal wird. Dies hat dem Platzierungsverfahren seinen Namen gegeben. Abbildung 3.6 zeigt eine prinzipielle Darstellung der Min-Cut-Platzierung. Dabei ist der erste Partitionierungsschritt der Netzliste und der Layoutfläche dargestellt.

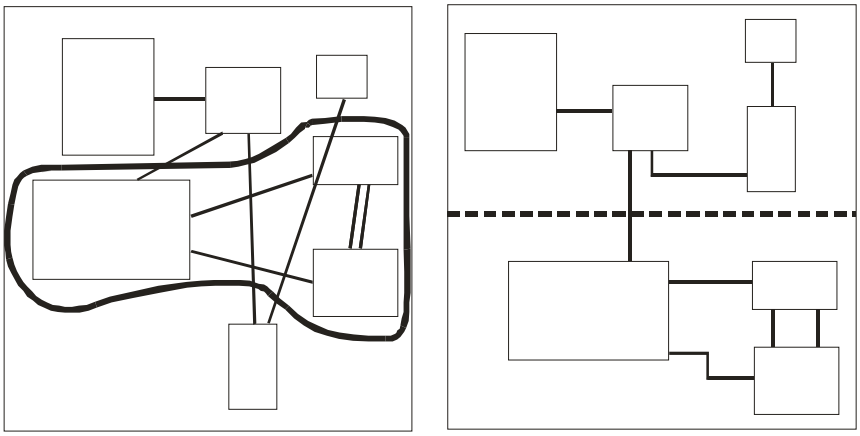

Abbildung 3.6: Prinzip der Min-Cut-Platzierung

Aufgrund der Laufzeitkomplexität der Berechnung einer optimalen Netzlistenpartitionierung, kann eine geeignete Platzierung nur unter Verwendung einer Heuristik zur Partitionierung berechnet werden. Dafür setzt man häufig den Kerninghan-Lin-Algorithmus [Brück93] ein. Dieser basiert auf der Idee, ausgehend von einer beliebigen Anfangspartitionierung, jeweils Zellpaare mit einer maximalen Reduzierung bzw. minimaler Erhöhung der Anzahl von Verbindungen, die den Schnitt kreuzen, zu finden, und

solange Vertauschungen durchzuführen, bis eine minimale Schnittzahl erreicht ist. Dieses Verfahren wird so lange fortgesetzt bis keine Verbesserung mehr erreicht werden kann.

In der Standardimplementierung der Min-Cut-Platzierung werden keine Timing-Restriktionen berücksichtigt. Um zu verhindern, dass alle Netze gleich behandelt werden, kann - vergleichbar zur kräftegesteuerten Platzierung - jedem Netz ein Gewicht zugewiesen werden, das bei der Berechnung der Schnittkosten ausgewertet wird. Dadurch werden Netze mit einem geringen Gewicht häufiger als Netze mit einem höheren Gewicht geschnitten. Das führt dazu, dass Zellen, die mit einem Netz mit hohem Netzgewicht verbunden sind, in einer Partition bleiben und dementsprechend im Layout enger zueinander platziert werden. Außer dieser Priorisierung gibt es keine weiteren Möglichkeiten zur Berücksichtigung von Timing-Constraints bei der Verwendung der Standardimplementierung von Min-Cut.

3.1.1.6 Modifizierte Min-Cut-Platzierung

Zur Berücksichtigung von Timing-Constraints gibt es viele Erweiterungen [Wang00, Ou00, Yang02] des Min-Cut-Algorithmus. Alle modifizierten Verfahren verwenden weiterhin die Netzlistenpartitionierung als Kernkomponente des Verfahrens. Dabei setzen die meisten Verfahren hMetis [Karypis99] zur Partitionierung ein. Allerdings erfolgt die Partitionierung nicht wie im Standard-Min-Cut-Verfahren jeweils in zwei Partitionen, sondern in jeweils vier (Quadri-Section).

In [Sarrafzadeh97] wird die Min-Cut-Platzierung mit Simulated Annealing kombiniert. Mit der Min-Cut-Partitionierung wird eine Globalplatzierung erzeugt, d.h. jeder Layoutteilfläche werden entsprechend der Netzlistenpartitionierung Zellen zugewiesen, die alle auf dem Mittelpunkt der Fläche platziert werden. Die rekursive Partitionierung wird abgebrochen, wenn ungefähr 10-30 Zellen in einer Partition liegen. Anschließend erfolgt die Optimierung der Globalplatzierung mit einem Simulated-Annealing-Ansatz. Dabei werden jeweils zwei Zellen aus unterschiedlichen Partitionen zur Vertauschung ausgewählt. Die Detailplatzierung erfolgt für jede Teilfläche separat und basiert ebenfalls auf Simulated Annealing (vgl. Abschnitt 3.1.1.3).

Die Verdrahtungslängen bei [Sarrafzadeh97] sind im Vergleich zur TimberWolf-Version 1.2 aus [Swartz95] um durchschnittlich ungefähr 5% kürzer, dabei wird designabhängig eine Laufzeitbeschleunigung von 2,5-20 erreicht. Während dieser Ansatz die Längen der einzelnen Netze gut optimiert, ist eine Berücksichtigung von Timing-Restriktionen auf Pfadebene nicht möglich.

In [Wang00] erfolgt die Globalplatzierung analog zu [Sarrafzadeh97]. Nach jedem Globalplatzierungsschritt erfolgt zusätzlich eine Verdrahtungslängenoptimierung. Dabei werden alle Layoutteilflächen zur Optimierung der Gesamtverdrahtungslänge miteinander vertauscht. Die Detailplatzierung wird in diesem Ansatz mit einem Greedy-Algorithmus durchgeführt. Dieser verteilt zunächst die Zellen auf der Layoutfläche und versucht anschließend, die Netzlängen durch heuristische Vertauschungsoperationen zu optimieren.

[Ou00] beschreibt einen Ansatz zur Beschränkung der Schnittanzahl von kritischen Pfaden, da ein Pfad in der Regel umso länger sein wird, je mehr Schnitte er enthält. Dazu erfolgt zunächst eine Berechung der Netzgewichte unter Berücksichtigung der Pfadlängen, so dass kritische Pfade höhere Netzgewichte als unkritische Pfade erhalten. Anschließend wird die maximale Schnittzahl für jeden Pfad festgelegt. Dann erfolgt ein Platzierungsschritt mit einem beliebigen Min-Cut-basierten Platzierungsalgorithmus, was genau einem Partitionierungsschritt entspricht. Im darauf folgenden Schritt wird die Einhaltung der maximalen Schnittzahl überprüft und es erfolgt ggf. ein Korrekturschritt. Anschließend wird eine Aktualisierung der Netzgewichte durchgeführt und es erfolgt der nächste Partitionierungsschritt. Die Detailplatzierung erfolgt in diesem Ansatz nicht mit Simulated Annealing, sondern durch quadratische Optimierung. [Ou00] erreicht mit diesem Platzierungsverfahren bei Verwendung des Partitionierers aus [Ou99] für die MCNC-Benchmarks im Vergleich zur Platzierung mit dem Partitionierungsverfahren von [Ou99] eine durchschnittliche Verkürzung der Verzögerungszeiten um 23%.

3.1.2 Verdrahtung

Die Aufgabe der Verdrahtung besteht in der Bestimmung der exakten Geometrie der Verbindungen zwischen den platzierten Zellen. Dabei sind die prozessspezifischen geometrischen Entwurfsregeln (Design Rules), die beispielsweise Mindestabstände und Mindestbreiten beinhalten, einzuhalten und es ist sicherzustellen, dass das gesamte Design verdrahtet werden kann. Zusätzlich muss die Einhaltung der maximalen Verzögerungszeiten zwischen den Zellen überprüft werden.

Wie man aus der Aufgabe der Verdrahtung erkennen kann, baut diese Phase des physikalischen Entwurfs auf den Platzierungsergebnissen auf. Damit ist die Qualität wie auch die Durchführbarkeit der Verdrahtung unmittelbar von den Platzierungsergebnissen abhängig.

Um die Komplexität des nachgewiesenermaßen NP-vollständigen [Brück93] Problems zu reduzieren, wird die Verdrahtung von integrierten digitalen Schaltungen zumeist in zwei Phasen durchgeführt. Man unterscheidet die Globalverdrahtung, in der die Verdrahtungsbereiche für die Leitungen belegt werden und die detaillierte Verdrahtung, in

der die geometrische Form der Leitungen festgelegt wird. Die aktuell zum Einsatz kommenden Algorithmen zur globalen und detaillierten Verdrahtung werden in den beiden folgenden Abschnitten genauer beschrieben.

3.1.2.1 Globalverdrahtung

Während der Globalverdrahtung erfolgt eine Zuweisung von Verdrahtungsteilflächen zu jedem zu verdrahtenden Netz. Damit wird für jedes Netz eine grobe Festlegung des Verlaufs getroffen, ohne dass die genaue Geometrie des Netzes festgelegt wird.

Die Globalverdrahtung besteht aus drei Phasen: Regioneneinteilung, Regionenzuweisung und Pinzuweisung [Sherwani95].

In der ersten Phase wird die gesamte Layoutfläche in so genannte Verdrahtungsregionen eingeteilt. Jede Verdrahtungsregion hat dabei eine bestimmte Kapazität, die die Anzahl der Leitungen angibt, die durch diese Region hindurchgeführt werden können, ohne dass Entwurfsregeln verletzt werden.

In der zweiten Phase erfolgt die Zuweisung von Regionen zu einem Netz. Dabei müssen zwei Randbedingungen berücksichtigt werden: Zum einen muss während der Globalverdrahtung gewährleistet werden, dass keiner Layoutteilfläche mehr Leitungen zugewiesen werden als die entsprechende Teilfläche Kapazität hat, zum anderen muss sichergestellt werden, dass die Verbindungen nicht länger als zulässig werden.

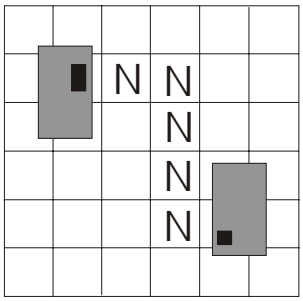

Abbildung 3.7: Globalverdrahtung

Abbildung 3.7 zeigt die Globalverdrahtung des Netzes N, welches zwei Zellen, deren Anschlusspins schwarz dargestellt sind, miteinander verbindet. Jedes weiße Rechteck repräsentiert eine Verdrahtungsteilfläche, ein „N“ in einem Rechteck kennzeichnet die Zuweisung des Netzes N zu der jeweiligen Teilfläche.

Letzter Schritt der Globalverdrahtung ist die Pinzuweisung. Bei der Pinzuweisung wird jedem Netz, das über mehr als eine Region verläuft, ein Pin auf der Grenze zwischen

zwei Regionen zugewiesen. Dieser Pin wird in der Detailverdrahtungsphase als Start- bzw. Endpunkt der Leitungen in der jeweiligen Verdrahtungsteilfläche verwendet.

Für das Kernproblem der Globalverdrahtung - der Regionenzuweisung - gibt es verschiedene Ansätze, die im Folgenden vorgestellt werden.

In den meisten Fällen erfolgt die Zuweisung der Leitungen zu den Regionen mit Hilfe eines Coarse-Grid-Modells. Dabei wird die Layoutfläche in gleich große Rechtecke mit einer definierten Kapazität eingeteilt. Jedes Rechteck repräsentiert eine Verdrahtungsregion und wird für die folgenden Schritte durch einen Punkt repräsentiert. Auf diesem Raster kann zur Globalverdrahtung prinzipiell jeder Verdrahtungsalgorithmus, der auch zur detaillierten Verdrahtung eingesetzt wird, angewendet werden. Eine Übersicht dieser Algorithmen befindet sich im Abschnitt 3.1.2.2.

Alternativ kann in der Zuweisungsphase ein Kanalschnittgraph verwendet werden. Dieser besteht aus Kanten, die die Nachbarschaft zwischen den Regionen repräsentieren, und Knoten, die die Schnittpunkte zwischen zwei Regionen darstellen. Die Kapazität jeder Region wird als Kantengewicht zu jeder Kante notiert. Die Zuweisung der Regionen erfolgt dann mit einem Algorithmus zur Bestimmung des kürzesten Pfads, wie zum Beispiel dem Dijkstra-Algorithmus [Ottmann90], der auf diesen Graphen angewendet wird oder mit Hilfe einer Steinerbaum-Heuristik [Steiner].

Ein neuerer Ansatz [Mo01b] zur Globalverdrahtung basiert auf der Kombination von kräftegesteuerter Platzierung und Maze-Routing (vgl. Abschnitt 3.1.2.2).

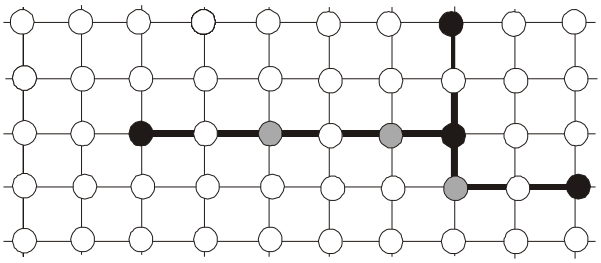

Abbildung 3.8: Kräftegesteuerte Globalverdrahtung

Dazu wird zunächst ein grobes Verdrahtungsraster erzeugt und alle in Abbildung 3.8 schwarz dargestellten Zellen werden mit einem Spannbaum auf diesem Raster verbunden. Anschließend werden die in der Abbildung hellgrau dargestellten Leitungspunkte erzeugt. Durch diese Punkte verläuft zu einem späteren Zeitpunkt die globale Verdrahtung, die mit einem Maze-Router erzeugt wird.

Ein Leitungspunkt wird immer genau dann erzeugt, wenn die Distanz zwischen zwei Leitungspunkten bzw. einem Terminal und einem Leitungspunkt ein bestimmtes Inter-

vall überschreitet, zwei Leitungspunkte nicht auf einer gemeinsamen horizontalen oder vertikalen Linie liegen oder eine größere Anzahl überbelegter Bereiche zwischen den Leitungspunkten liegt. Diese drei Fälle sind in Abbildung 3.9 dargestellt. Zwei Leitungspunkte werden zu einem zusammengefasst, wenn ihr Abstand einen bestimmten Schwellwert unterschreitet.

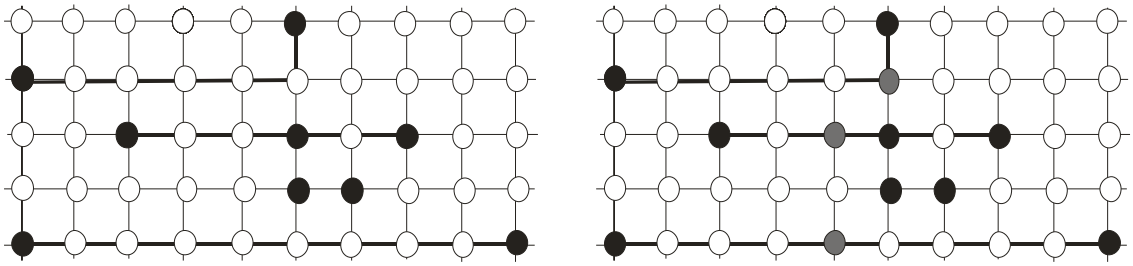

Abbildung 3.9: Einfügen von Leitungspunkten

Anschließend wird eine Kraft auf alle Leitungspunkte berechnet. Wie bei der kräftegesteuerten Platzierung besteht diese Kraft aus zwei Komponenten, einer anziehenden Kraft, die aus der Adjazenzmatrix berechnet wird und einer abstoßenden Kraft, die für eine gleichmäßige Verteilung der Leitungspunkte sorgen soll. Diese Kraft hängt nur von der Position ab und ist proportional zu der Anzahl der Zellen, die an dem aktuellen Ort liegen sowie zu der Wahrscheinlichkeit, dass eine Verbindung durch diese Position verlaufen wird.

Nach der Berechnung aller Kräfte werden die Leitungspunkte in Richtung der Kräfte bewegt und dadurch eine neue Platzierung der Leitungspunkte bestimmt. Danach erfolgt eine Neugenerierung der Leitungspunkte. Ist die Verschiebung der Leitungspunkte abgeschlossen, erfolgt die eigentliche Globalverdrahtung. In jedem Schritt werden zwei Leitungspunkte mit einem Maze-Router (vgl. Abschnitt 3.1.2.2.1) verdrahtet.

Ein anderer Ansatz zur Globalverdrahtung basiert auf vorgegebenen Routing-Pattern [Kastner00]. In diesem Verfahren wird die gesamte Layoutfläche zunächst in Kacheln eingeteilt und alle Mehrpunktnetze mittels einer Steiner-Heuristik [Steiner] in Zweipunktnetze zerlegt. Jede Kante einer Kachel hat eine feste Kapazität von Leitungen, die sie aufnehmen kann.

Als Routing-Pattern stehen in diesem Ansatz lediglich L-Shapes und Z-Shapes zur Verfügung. Bei der Durchführung der Verdrahtung muss ausgehend von Start- und Zielpunkt ein Pattern zur Verdrahtung ausgewählt werden. Die vier zur Verfügung stehenden Möglichkeiten, unter Verwendung von L-und Z-Shapes ein Zwei-Punkt-Netz zu verdrahten, sind in Abbildung 3.10 dargestellt.

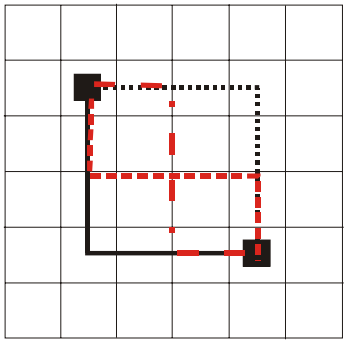

Abbildung 3.10: Verdrahtungsmöglichkeiten beim Pattern-Routing

Zur Auswahl der Verdrahtungsform wird zum einen die lokale Leitungsdichte, die an der Belegung der Kantenkacheln gemessen wird, und zum anderen die Leitungslänge berücksichtigt. Die Verdrahtungsentscheidung erfolgt anhand beider Parameter, deren Einfluss durch den Benutzer gewichtet werden kann.

In [Kastner00] können ungefähr 80% der Verbindungen auf einem Chip mit diesem Verfahren verdrahtet werden. Die restlichen Verbindungen lassen sich nicht mit den gegebenen Shapes ohne Überbelegung einiger Kachelkanten realisieren. Diese Verbindungen werden deshalb abschließend mit einem Maze-Router verdrahtet.

Der entscheidende Vorteil des Pattern-Routings liegt in der Vorhersagbarkeit der Verbindungslängen. Bereits während der Platzierung kann für viele Netze ihre Länge sicher vorhergesagt werden und diese kann bereits in der Platzierungsphase verwendet werden. Nachteil dieses Vorgehens ist die Beschränkung der Routing-Pattern und die damit einhergehende Behandlung der Netze, die nicht mit den gegebenen Pattern verdrahtet werden können. Diese Leitungen können im ungünstigsten Fall deutlich länger als durch die Pattern vorhergesagt sein.

Der Algorithmus von [Hu00] verfolgt den entgegen gesetzten Weg. In der ersten Phase werden so genannte flexible Verbindungen eingesetzt. Eine flexible Verbindung hat eine feste Länge, aber keine feste geometrische Form. Alle Zellen werden zunächst mit diesen Verbindungen verdrahtet. Anschließend wird die Layoutfläche in Kacheln eingeteilt. Bei dieser sukzessiven Teilung werden den Begrenzungen der Kacheln Punkte zugewiesen, durch die später die Verdrahtung verläuft. Dabei müssen sowohl die Belegung der Kachel als auch die maximal zulässige Pfadlänge berücksichtigt werden. Die sukzessive Teilung und Punktzuweisung endet, wenn in jeder Kachel nur noch einige Zellen liegen. Anschließend erfolgt eine konventionelle Verdrahtung unter Verwendung der vorher bestimmten Punkte.

Der Vorteil dieses Ansatzes liegt in der späten Auswahl der Geometrie der Globalverdrahtung, wobei die maximale Leitungslänge und damit auch die Verzögerungszeit

bereits vorher fest liegen. Nachteilig kann sich auswirken, dass nicht garantiert ist, dass alle Verbindungen unter Verwendung der zuvor bestimmten Punkte verdrahtbar sind.

3.1.2.2 Detaillierte Verdrahtung

Während der detaillierten Verdrahtung wird die genaue geometrische Form jeder Leitung bestimmt. Dazu sind der Start- und die Endpunkte der Verbindung aus der Platzierung sowie die Flächen, durch die die Verbindung verlaufen soll, aus der Globalverdrahtung bekannt.

Bei der detaillierten Verdrahtung unterscheidet man zwischen allgemeinen Verdrahtern, die zwei beliebig angeordnete Punkte miteinander verbinden und eingeschränkten Verdrahtern, die nur speziell angeordnete Punkte verbinden. Zu den eingeschränkten Verdrahtern gehören Kanal-Verdrahter, Switchbox-Verdrahter und Fluß-Verdrahter. Diese drei Typen von Verdrahtern arbeiten auf speziellen Layoutstrukturen, die im allgemeinen Fall nicht vorhanden sind bzw. - wie z.B. Verdrahtungskanäle - bei aktuellen Technologien nicht mehr eingesetzt werden. Deshalb sollen hier nur die allgemeinen detaillierten Verdrahter betrachtet werden.

3.1.2.2.1 Rasterbasierte Verdrahtung

Rasterbasierte Labyrinth-Verdrahtungsverfahren (Maze-Router) suchen einen Weg zwischen Start- und Zielpunkt eines Netzes unter Verwendung eines gerasterten Verdrahtungsmodells und basieren in den meisten Fällen auf dem Algorithmus von Moore [Moore59]. Dieser Algorithmus simuliert die Ausbreitung einer Welle auf dem Verdrahtungsraster.

Dazu wird zunächst die gesamte Layoutfläche auf ein Raster abgebildet, wobei ein Rasterelement in der Regel genau die Breite einer Leitung plus den Minimalabstand zwischen zwei Leitungen hat [Adler01]. Die Verdrahtung kann als graphentheoretisches Problem betrachtet werden. Jedes Rasterfeld des Layouts entspricht einem Knoten des Graphen. Jeder Knoten ist über Kanten mit seinen Nachbarknoten verbunden. Bei Mehrlagenverdrahtung hat jeder Knoten sechs Nachbarn, vier in der gleichen Metallisierungsebene und je einen in der darüber bzw. darunter liegenden Ebene.

Ausgehend von einem Startknoten (S) werden alle Nachbarn dieses Knotens besucht. Befindet sich unter den besuchten Knoten der Zielknoten (Z), ist der Weg gefunden und die Suche wird beendet. Ist das nicht der Fall, werden alle Nachbarknoten in eine Warteschlange eingereiht und so lange rekursiv die Nachbarknoten dieser Knoten besucht, bis der Zielknoten erreicht wird oder die Warteschlange leer ist. In diesem Fall existiert

kein Weg zwischen den beiden Knoten. Jeder besuchte Knoten wird als besucht gekennzeichnet. Wenn der Zielknoten gefunden worden ist, kann man den Weg vom Ziel- zum Startknoten anhand der markierten Knoten zurückverfolgen. Damit ergibt sich folgender Ablauf für die rasterbasierte Verdrahtung:

Rasterbasierte Verdrahtung

Besuche den Startpunkt der Leitung und markiere diesen mit einer 1
for (alle besuchten Rasterelemente)
  Besuche und markiere alle direkten Nachbarelemente mit i+1, wenn das betrachtete Feld mit i markiert ist
end
Beginnend am Verbindungsendpunkt: Markiere einen zusammenhängenden Weg von markierten Rasterelementen mit absteigenden Markierungswerten

Abbildung 3.11 zeigt exemplarisch die rasterbasierte Verdrahtung eines Zweipunktnetzes

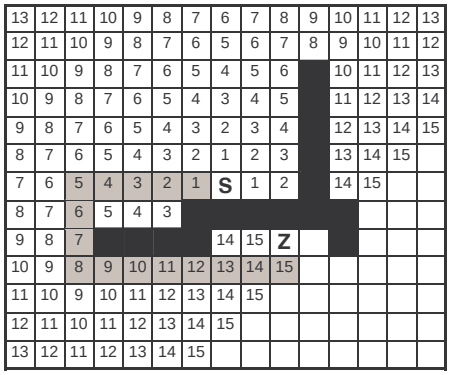

Abbildung 3.11: Maze-Routing

Bei der rasterbasierten Verdrahtung ist es möglich, den einzelnen Kanten unterschiedliche Kantengewichte zuzuweisen, die die Kosten der Verbindung repräsentieren. Dieses beeinflusst die quadratische Laufzeit in Abhängigkeit von der Anzahl der Rasterelemente nicht.

Zur Bestimmung der als nächstes besuchten Nachbarn können unterschiedliche Verfahren auf dem Rastergraphen angewendet werden. Der Lee-Algorithmus [Lee61] verwendet eine symmetrische Breitensuche und ist der wohl bekannteste Vertreter der rasterbasierten Verdrahtungsverfahren. Im Hadlock-Algorithmus wird zur Suche der Nachbarn die A*-Suchheuristik [Hart68] verwendet, im Soukup-Algorithmus eine zielgerichtete Tiefensuche.

Der entscheidende Vorteil der Labyrinth-Verdrahtungsverfahren liegt in der Eigenschaft, dass sie immer eine Verbindung finden, wenn diese existiert und diese Verbindung immer die kürzeste bezüglich der gewählten Kostenfunktion ist, wenn Breitensuche als Suchverfahren verwendet wird. Nachteil dieses Verfahrens ist die Abbildung auf ein Raster, bei dem auch Freiflächen gerastert und gespeichert werden müssen. Dadurch steigt die Anzahl der Rasterelemente quadratisch mit der Kantenlänge der Chips an und führt zu einem sehr hohen Speicherbedarf. Sehr große Schaltungen lassen sich aufgrund des Speicherbedarfs nicht mehr mit rasterbasierten Verfahren verdrahten.

3.1.2.2.2 Linienbasierte Verdrahtung

Diese Verfahren zur detaillierten Verdrahtung zeichnen sich dadurch aus, dass sie im Gegensatz zu den rasterbasierten Verdrahtungsverfahren ohne Raster arbeiten und so einen deutlich geringeren Speicherbedarf haben. Im Vergleich zu den Labyrinthverdrahtern, bei denen eine Teillösung durch das Hinzufügen eines Rasterpunkts expandiert wird, werden die Teillösungen bei Liniensuchverfahren durch das Hinzufügen von Liniensegmenten erweitert. Diese Liniensegmente werden auch als Versuchslinien bezeichnet. Da jedem Liniensegment in den meisten Fällen viele Rasterpunkten entsprechen, sind der Speicherbedarf und die Laufzeit bei linienbasierten Verdrahtungsverfahren deutlich geringer als bei rasterbasierten Verfahren.

Bei Liniensuchverfahren werden zunächst ausgehend von Start- und Zielpunkt horizontale und vertikale Versuchslinien (Basislinien) gezogen. Diese Linien werden so weit ausgedehnt bis sie auf ein Hindernis treffen. Wenn sich die so erzeugten Linien nicht schneiden, müssen neue, senkrecht auf den bisher erzeugten Linien stehende, Linien (Fluchtlinien) erzeugt werden. Dieses Vorgehen wird iterativ solange fortgesetzt, bis sich eine vom Startpunkt und eine vom Zielpunkt erzeugte Linie schneiden.

Man unterscheidet zwei prinzipiell verschiedene Ansätze der Linienverdrahtung: Die auf dem Algorithmus von Mikami [Mikami68] basierenden und in Abbildung 3.12 dargestellten Ansätze erzeugen die Versuchslinien in einem äquidistanten Raster, während die auf dem Ansatz von Hightower [Hightower69] basierenden Verfahren in jedem Iterationsschritt nur eine horizontale bzw. vertikale Linie heuristisch erzeugen.

Damit ergibt sich der folgende Ablauf der linienbasierten Verdrahtung:

Linienbasierte Verdrahtung
Erzeuge von Start- und Endpunkt der zu verdrahtenden Verbindung jeweils eine Linie bis zum Layoutrand oder nächsten Hindernis
Existiert ein Schnittpunkt dieser Linien, ist die Verbindung gefunden
Erzeuge auf jeder der beiden Linien Fluchtlinien senkrecht zu den zuvor erzeugten Linien
Setze das Verfahren so lange fort, bis der vom Startpunkt ausgehende Linienzug den von Endpunkt ausgehenden schneidet

Die Liniensuchverfahren zeichnen sich durch ihren geringen Speicherbedarf aus. Nach [Ohtsuki86] ist dieser proportional zu der Anzahl der erzeugten Linien. Der Geschwindigkeitsvorteil gegenüber den rasterbasierten Verfahren ist bei wenigen aber großen Elementen besonders groß. Nachteilig kann sich die prinzipbedingte minimale Anzahl von Richtungswechseln auf die Verdrahtbarkeit auswirken. Werden - wie bei den auf dem Hightower-Algorithmus basierenden Ansätzen - die Versuchslinien heuristisch erzeugt, wird ein existierender Pfad nicht immer gefunden und ein gefundener Weg muss nicht immer der kürzeste sein.

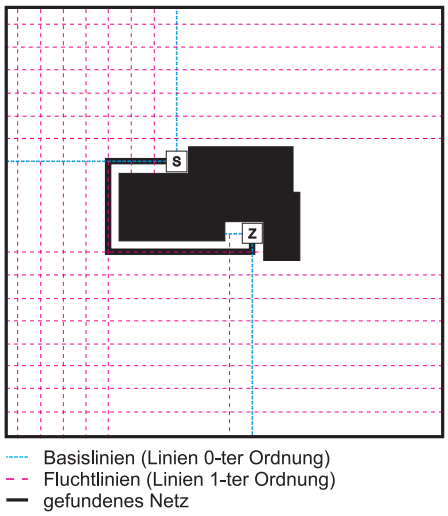

Abbildung 3.12: Mikami-Algorithmus

3.1.2.2.3 Timing-Driven-Verdrahtung

Alle bisher beschriebenen Verfahren zur Detailverdrahtung berücksichtigen keine Timing-Constraints. Da zukünftige Designs immer laufzeitkritischer werden, sind in den letzten Jahren unterschiedliche Ansätze zur Verbesserung des Zeitverhaltens des Verbindungsnetzwerks entwickelt worden.

Im Ansatz von [Tseng98] wird ein Timing-Optimierungsschritt zwischen der Global- und Detailverdrahtung, wie in Abbildung 3.13 dargestellt, durchgeführt.

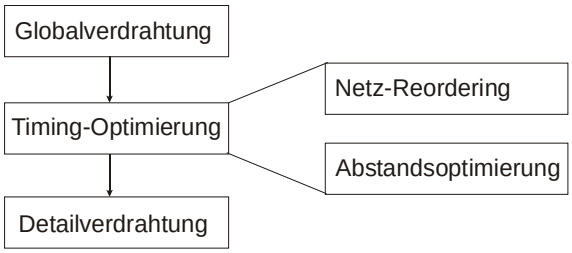

Abbildung 3.13: Timing-Driven-Routing

Der Timing-Optimierungsschritt besteht aus zwei Phasen. In der ersten Phase wird die Reihenfolge der Netze zueinander verändert, während in der zweiten Phase die Optimierung der Abstände zwischen den Leitungen durchgeführt wird.
In der ersten Phase soll die Länge der Leitungen und die Anzahl der Knicke in den Leitungen optimiert werden. Dazu wird die Globalverdrahtung auf Umwege und unnötige Knicke untersucht. Diese Untersuchung erfolgt durch die Betrachtung der in Abbildung 3.14 schwarz dargestellten Übergangspunkte zwischen den Verdrahtungsteilflächen (vgl. Abschnitt 3.1.2.1) der Globalverdrahtung. Enthält ein Netz, das über eine größere Distanz gerade verläuft, auf benachbarten Übergängen unterschiedliche Übergangspunkte, sind Verbindungen quer zur Verdrahtungsrichtung bei der Detailverdrahtung nicht mehr zu verhindern.

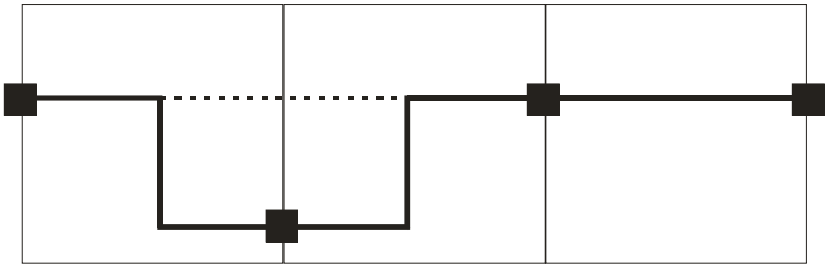

Abbildung 3.14: Übergangspunkte und Detailverdrahtung

Deshalb werden alle kritischen Netze selektiert und es wird versucht, die Länge dieser Netze durch Vertauschen der Netzreihenfolge in den Verdrahtungsteilflächen zu verkürzen, indem beispielsweise die in Abbildung 3.14 dargestellten Umwege entfernt werden. Damit wird der Umweg durch den gestrichelt dargestellten Leitungsverlauf ersetzt.

Während der Abstandsoptimierung im zweiten Teil der Optimierungsphase des Ansatzes von [Tseng98] werden zwischen kritischen Netzen und ihren Nachbarnetzen

größere Abstände eingefügt, um so die Kopplungskapazität und damit die Verzögerungszeit zu reduzieren. In [Tseng98] wird dadurch bezüglich der MCNC-Benchmarks eine Verringerung der Verzögerungszeit auf kritischen Pfaden von 8-25% erreicht.

Weitere Ansätze, die die Verzögerungszeit durch Ändern von Leitungsreihenfolgen und Abständen zwischen den Signalleitungen modifizieren, können in [Chaudhary93], [Zhou96] und [Jhang96] gefunden werden.

In [Hur99] wird ein Timing-Driven-Maze-Router vorgestellt. Dieser Ansatz arbeitet auf einem Graphen-Modell des Verdrahtungsrasters. Dabei repräsentiert jede Kante ein Verdrahtungsrasterelement, während jeder Knoten eine Auswahl mehrerer Wege darstellt. Zu jeder Kante des Graphen werden der Widerstand und die Kapazität des jeweiligen Rasterelements annotiert. Damit ergibt sich ein Graph, der in Abbildung 3.15 für ein 4x4 Verdrahtungsraster beispielhaft dargestellt ist.

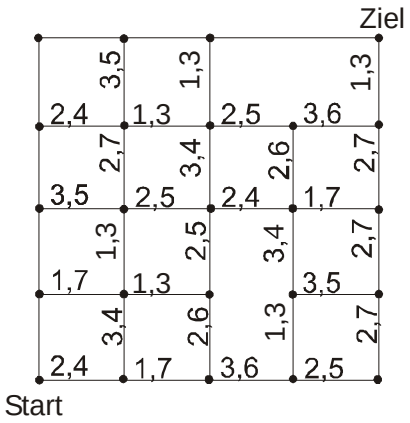

Abbildung 3.15: Routing-Graph mit annotierten RC-Paaren

Mit einem modifizierten Dijkstra-Algorithmus [Ottmann90] wird innerhalb des Graphen die bezüglich der Verzögerungszeit kürzeste Verbindung gesucht.

Aufgrund des Rechenaufwands zur Bestimmung des Routing-Graphen und der Laufzeitkomplexität von $O(n^3)$ des Dijkstra-Algorithmus, wobei n die Anzahl der Knoten des Graphen ist, ist dieses Verfahren jedoch nur für kleine Schaltungen anwendbar.

3.1.2.3 Multilevel-Verdrahtung

Multilevel-Routing [Cong00, Cong01, Lin02] ist ein Ansatz, bei dem Global- und Detailverdrahtung gleichzeitig in unterschiedlichen Genauigkeitsstufen durchgeführt werden.

Beim Multilevel-Routing wird die Layoutfläche zunächst gleichmäßig in Kacheln eingeteilt. In [Lin02] wird anschließend eine Global- und Detailverdrahtung für alle lokalen Verbindungen durchgeführt. Eine lokale Verbindung ist dabei auf eine Kachel beschränkt. Sukzessiv werden jeweils vier Kacheln zu einer größeren Kachel, wie in der linken Hälfte von Abbildung 3.16 dargestellt, verschmolzen.

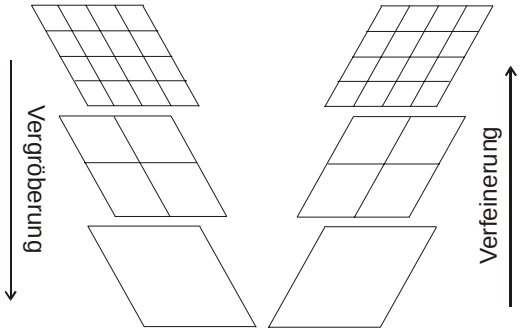

Abbildung 3.16: Multilevel-Verdrahtung

Nach jedem Verschmelzungsschritt werden alle noch nicht verdrahteten Netze der jeweiligen Kachel verdrahtet. Dieses wird so lange fortgesetzt, bis ein Schwellwert für die maximale Kachelgröße erreicht wird. Danach erfolgen wieder Verfeinerungsschritte. Jede Kachel wird in vier Kacheln unterteilt und innerhalb jeder Kachel erfolgt eine lokale Optimierung der Verdrahtung.

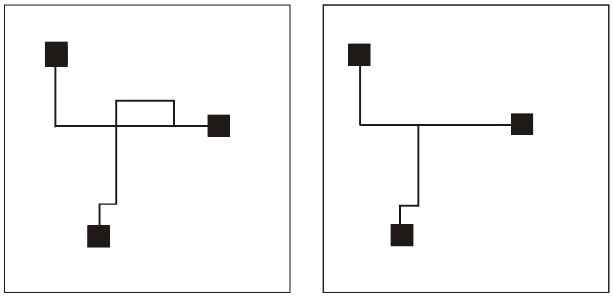

Abbildung 3.17: Lokale Verdrahtungsoptimierung

Abbildung 3.17 zeigt eine mögliche Netzoptimierung. Prinzipiell können alle Methoden zur Verdrahtungsoptimierung - beispielsweise die in Abschnitt 3.4.3 beschriebenen Verfahren - in diesem Schritt eingesetzt werden.

Multilevel-Routing bietet einen deutlich flexibleren Verdrahtungsablauf als das zweistufige Vorgehen bei der Teilung in Global- und Detailverdrahtung. Insbesondere die Optimierungsschritte während der Verfeinerung der Kacheln können zu einer signifikanten Verbesserung der Verdrahtung führen. Dieses ist bei dem zweistufigen Vorgehen nicht möglich, da während der Detailverdrahtung die Globalverdrahtung als unveränderbar angesehen wird. Außerdem können mit Multilevel-Routing-Verfahren

prinzipiell beliebig große Schaltungen verdrahtet werden, indem die Kacheln geeignet groß gewählt werden. Zusätzlich bietet Multilevel-Routing einige Vorteile bezüglich der Einhaltung von Timing-Restriktionen. Nach Erreichen des gröbsten Kachelrasters können alle Netz-Verzögerungszeiten berechnet werden und eventuell vorhandene Verzögerungszeit-Überschreitungen in den Verfeinerungsschritten beseitigt werden. Außerdem kann mit Hilfe einer Globalverdrahtung auf jedem Kachelraster eine Verzögerungszeit-Abschätzung erfolgen, deren Genauigkeit von der Rasterung abhängt.

3.2 Probleme bei separatem Platzieren und Verdrahten

In den letzten zehn Jahren war die zweistufige Layouterzeugung für Digitaldesigns Standard. Dabei wurden unterschiedliche Typen von Platzierern und Verdrahtern miteinander kombiniert.

Bei einem solchen mehrstufigen Vorgehen ist immer eine Abschätzung der Folgeschritte erforderlich. Diese führt häufig zu ungenauen Modellen in frühen Entwurfsphasen wie in Abschnitt 2.3 beschrieben.

Auch wenn die Entwicklung von Verdrahtungsmodellen in den letzten Jahren deutliche Fortschritte gemacht hat, sind die Abweichungen der Verdrahtungslängen zwischen der Netzlängenschätzung während der Platzierung und den realen Verdrahtungslängen immer noch so hoch, dass einige Schaltungen nicht unter Einhaltung aller Timing-Restriktionen verdrahtet werden können. Die Abweichungen der Laufzeitberechnungen zwischen Synthese und Layout sind sogar noch größer.

3.3 Integrierte Ansätze zur Platzierung und Verdrahtung

Neben den Ansätzen, die sich an der konventionellen Einteilung des Designflows in Synthese, Platzierung, Verdrahtung und Post-Layout-Optimierung orientieren, gibt es einige neue Ansätze, in denen zwei oder drei Entwurfsschritte in einem vereinigt werden. Dadurch soll das Problem der Modellierung umgangen werden. Diese integrierten Lösungen teilt man in synthesebasierte, platzierungsbasierte und einheitliche Ansätze ein [Keutzer97], je nachdem, ob der Schwerpunkt des Verfahrens bei der Synthese oder der Platzierung liegt bzw. ob Platzierung und Verdrahtung in einem Schritt durchgeführt werden. Diese drei unterschiedlichen Typen von Ansätzen werden in den folgenden Abschnitten vorgestellt.

3.3.1 Synthesebasierte Ansätze

Während das Ergebnis einer Digitalsynthese eine Gatternetzliste in der strukturellen Sicht ist, erweitern integrierte synthesebasierte Ansätze die Synthese um einige Aspekte des physikalischen Entwurfs. Dabei erhält jedes Element der Netzliste bereits im Syntheseschritt eine zusätzliche Platzierungsinformation.

In [Pedram91] werden die Abbildung der Gatter auf die Zielbibliothek (Technology-Mapping) und die globale Platzierung gleichzeitig durchgeführt. Dadurch stehen während des Technology-Mappings genauere Informationen über die Verdrahtungslänge und Platzierungsfläche zur Verfügung.

Nach jedem Mapping-Schritt wird die entsprechende Zelle platziert. Dabei erfolgt keine detaillierte Platzierung, sondern lediglich eine Globalplatzierung. Darunter versteht man in diesem Ansatz die Platzierung von Zellen ohne räumliche Ausdehnung, ohne mehrere Zellen an den gleichen Koordinaten auf der Layoutfläche zu platzieren. Dazu wird ein auf quadratischer Optimierung basierender Platzierungsalgorithmus verwendet. Nach jedem Platzierungsschritt erfolgt die Verdrahtungslängenabschätzung. Danach kann für die gleiche Zelle ein anderes Mapping durchgeführt und festgestellt werden, welches Mapping-Ergebnis bezogen auf die globale Platzierung bessere Ergebnisse liefert und deshalb verwendet werden sollte.

Ein ähnlicher Ansatz wird in [Gosti01] verwendet. Im Unterschied zu [Pedram91] erfolgt in diesem Verfahren die Globalplatzierung aufbauend auf [Eisenmann98] jeweils nach dem technologieunabhängigen Optimierungsschritt der Synthese und nach dem Mapping. Außerdem wird bei [Gosti01] ein einheitliches Netzmodell für Global- und Detailverdrahtung eingeführt, während in [Pedram91] zwei unterschiedliche Modelle zum Einsatz kommen.

Der Physical Compiler der Firma Synopsys [Synopsys_PC] berechnet eine Platzierung ebenfalls gleichzeitig mit der Synthese. Dort wird jeder Zelle während der Synthese eine potenzielle Platzierungsposition zugewiesen, die in der darauf folgenden Platzierungsphase ausgewertet werden kann.

Der Ansatz von [Pedram91] führt im Vergleich zu zwei voneinander unabhängigen Entwurfsschritten zu einer Verringerung der Verzögerungszeiten, aber auch in fast allen Fällen zu einer im Mittel bis zu 25% größeren Fläche. Da in diesem Ansatz nach dem Synthese-Platzierungsschritt erst anhand weiterer Schritte die Qualität des Mappings geprüft werden kann, steigt die Laufzeit dieses Verfahrens deutlich an. In [Gosti01] sind die Veränderungen kleiner. Während die Verzögerungszeiten der Leitungen um durchschnittlich 12% sinken, steigt die Fläche um 10% an.

3.3.2 Platzierungsbasierte Ansätze

Die platzierungsbasierten Ansätze zeichnen sich dadurch aus, dass ausgehend von der Platzierung Interaktionen mit der Synthese oder der Verdrahtung stattfinden.

In [Stenz97] wird zunächst eine Synthese mit einem Standardsynthese-Werkzeug durchgeführt. Anschließend erfolgt eine initiale Platzierung, die mit einem beliebigen Werkzeug durchgeführt werden kann. Zur Verringerung der Abweichung zwischen dem in der Synthese bestimmten Laufzeitverhalten der Schaltung werden in diesem Ansatz iterativ gleichzeitig Netzlistenänderungen und Platzierungsänderungen vorgenommen.

Als erstes wird in jedem Iterationsschritt der längste Pfad der Schaltung bestimmt. Dazu werden die Platzierungsinformationen benutzt. Anschließend wird versucht, den längsten Pfad durch Ändern der Netzliste zu verkürzen [Rohfleisch95]. Dabei wird die logische Funktion der Schaltung nicht verändert. Trotzdem können in diesem Modifikationsschritt Gatter entfernt oder auch Gatter hinzugefügt werden. Die hinzugefügten Gatter werden im Layout in räumlicher Nähe zu dem modifizierten Pfad platziert. Dabei ist eine Überlappung mit bereits platzierten Elementen zulässig. Nach einer gewissen Anzahl von Netzlistenänderungen wird die Platzierung wieder legalisiert. Diese Platzierungsänderungen sollen im Gegensatz zu einer inkrementellen Veränderung der Platzierung, mit der die hinzukommenden Zellen platziert werden könnten, deutlichere Änderungen der Platzierung bewirken. Diese sind für weitere sinnvolle Netzlistenänderungen erforderlich, um die Verzögerungszeiten möglichst vieler Pfade zu verändern, da bei näherungsweise gleichen Verzögerungen in allen Pfaden keine neuen Modifikationen vorgenommen werden können. Mit diesem Vorgehen erreicht [Stenz97] eine durchschnittliche Verbesserung der Pfadlänge um ca. 14%. Dabei steigt die Laufzeit nicht wesentlich an.

Ein ähnlicher Ansatz wird in [Lalgudi94] verfolgt. Hierbei wird ebenfalls ausgehend von einer Initialplatzierung eine Timing-Analyse durchgeführt. Anschließend wird der kritische Pfad resynthetisiert. Dabei werden in diesem Pfad zunächst alle Buffer entfernt und anschließend neue Buffer erzeugt, sowie ein Gate-Sizing durchgeführt. Die neu erzeugten Buffer erhalten von dem Resynthese-Werkzeug eine potenzielle Position im Layout zugewiesen. Diese Zellen werden anschließend mit möglichst geringem Änderungsaufwand in das Layout integriert. Dieser Ansatz verbessert bei [Lalgudi94] das Laufzeitverhalten in den kritischen Pfaden um durchschnittlich ca. 15%.

Platzierungsbasierte Ansätze können die Verzögerungszeiten der kritischen Pfade deutlich senken. Allerdings verändern diese Algorithmen nur das Verhalten der kritischen Pfade und erreichen auch nur dann deutliche Verbesserungen, wenn die Anzahl der als kritisch betrachteten Pfade nicht zu groß ist. Dies trifft jedoch bei modernen Designs häufig nicht zu, weil dort ein großer Teil der Pfade kritisch ist. Ein weiterer Nachteil ist,

dass diese Verfahren auf einem fertigen Layout aufsetzen. Dabei hängt der Erfolg dieser Verfahren von der zuvor mit einem anderen Werkzeug erzeugten Initialplatzierung ab.

3.3.3 Einheitliche Ansätze

In [Lou97] wird aufbauend auf einer konventionellen Register-Transfer-Level-Synthese ein Ansatz für gleichzeitiges Technology Mapping und lineare Platzierung vorgestellt. Zunächst wird das Technology Mapping ohne Kenntnis über Platzierung und Verdrahtung durchgeführt. Die während dieser Phase durchgeführten Optimierungsschritte müssen für die folgenden Schritte im Designprozess nicht unbedingt optimal sein. Deshalb werden in diesem Ansatz nicht-optimale Lösungen aus dem Mapping nicht einfach verworfen, sondern die Bewertung der Mapping-Qualität erfolgt erst nach der Platzierung mit Hilfe einer Kostenfunktion, die die Kombination zwischen Mapping und Platzierung bewertet. Abbildung 3.18 soll den Vorteil dieses Verfahrens erläutern.

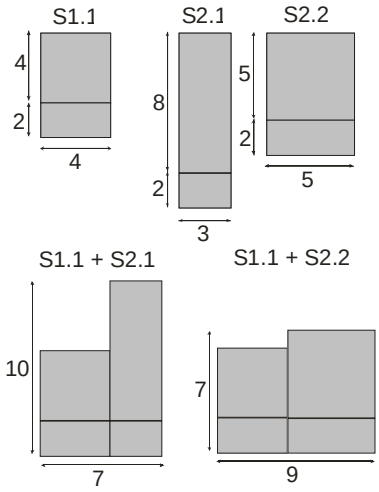

Abbildung 3.18: Flächenkombinationen

Die oberen drei Anordnungen der Abbildung zeigen den Flächenbedarf für zwei Module (S1 und S2) nach dem Mapping, wobei es für das Modul S2 zwei unterschiedliche Mapping-Lösungen gibt. Das jeweils größere Rechteck in der Abbildung repräsentiert die Gatterfläche und das kleinere die abgeschätzte Verdrahtungsfläche. Für das Modul S2 ist das Mapping S2.1 die flächengünstigere Abbildung. In der darauf folgenden Kombination der Module eins und zwei stellt man jedoch fest, dass die Lösung S2.2 für das Gesamtproblem die bessere Lösung ist. Deshalb werden in diesem Ansatz flächenungünstigere Mapping-Lösungen bis zur Platzierung gespeichert und die Auswahl der Mapping-Lösung erfolgt erst bei der Platzierung. Für bestimmte Schaltungen kann mit Hilfe dieses Ansatzes eine optimale lineare Platzierung erzeugt werden. Dazu wird der Keutzer-Algorithmus [Keutzer87] für das Technology Mapping mit den Algorithmus von Yannakakis [Yannakakis85] zur Platzierung kombiniert.

Bei der Betrachtung von über 20 Designs ergibt sich bei diesem Vorgehen in [Lou97] im Durchschnitt eine Verringerung der Verzögerungszeiten um 20% bei näherungsweise konstantem Flächenbedarf. Nachteil dieser integrierten Lösung ist zum einen die deutlich erhöhte Laufzeit, da sich die Laufzeiten von Mapping und Platzierung durch das Ineinanderschachteln multiplizieren. Zum anderen erhöht sich die Datenmenge deutlich, da mehrere Mapping-Lösung zwischengespeichert werden müssen. Der Vorteil dieses Verfahrens ist, dass unter den gegebenen Randbedingungen eine optimale Lösung gefunden werden kann. Leider ist dieses Verfahren auf lineare Platzierung beschränkt, die heute keine Praxisrelevanz mehr hat.

In [Salek98] bzw. [Salek99] wird der Ansatz von [Lou97] auf das Floorplanning erweitert. Dabei können beliebige Schaltungen bearbeitet werden. In einem ersten Schritt wird die Schaltung in baumartige Schaltungsteile zerlegt, die anschließend gemäß dem Ansatz von Lou abgebildet und verdrahtet werden. Die Ergebnisqualität entspricht dem oben beschriebenen Ansatz.

3.4 Post-Layout-Optimierungen

Unter Post-Layout-Optimierungen versteht man nach der Fertigstellung des Layouts durchgeführte Modifikationen am Layout. Diese Verfahren werden in den letzten Jahren immer häufiger eingesetzt, um einen wiederholten vollständigen Layoutdurchlauf zu vermeiden, wenn Laufzeitprobleme auftreten.

Ausgehend von einer Extraktion, bei der aus dem Layout eine Netzliste mit parasitären Elementen erstellt wird, soll während der Post-Layout-Optimierung das Verzögerungszeitverhalten der Schaltung verbessert werden, ohne das gesamte Layout neu zu erzeugen. Dabei setzt man die im folgenden beschriebenen Techniken ein [Cong97a, Cong97b].

3.4.1 Buffer-Insertion

Bei dieser auch als Repeater-Insertion bezeichneten Technik werden in die bestehende Platzierung zur Reduktion der Verzögerungszeiten zusätzliche Buffer eingefügt. Da die Verzögerungszeit einer Leitung quadratisch mit der Länge wächst [Zhou99], kann man durch das Einfügen von Buffern die Verzögerungszeit signifikant reduzieren. Dabei besteht die Buffer-Insertion aus zwei Schritten. Im ersten Schritt werden mögliche Positionen zum Einfügen von Buffern bestimmt. Diese Positionen ergeben sich aus den abgeschätzten Verzögerungszeiten in allen Pfaden, sowie den noch zur Verfügung stehenden Platzierungspositionen für Buffer. Dabei kann ein freier Platz im Layout

mehreren Netzen zugewiesen werden. Im zweiten Schritt werden dann aus den möglichen Buffern die optimalen Netze und Positionen ausgewählt.

In [Jagannathan00] wird ein Algorithmus zum Einfügen von Buffern mit pseudopolynomialer Laufzeit vorgestellt. Dieser Algorithmus mit der Bezeichnung DRCR (Delay Reduction to Cost Ratio Maximization) versucht, in jedem Pfad das Maximum des Verhältnisses Delay-Änderung zu Kosten zu erreichen. Dabei kann die Verzögerungszeit-Abschätzung mit verschiedenen Modellen erfolgen. Die Kosten werden durch die Anzahl der zusätzlichen Kapazitäten und den Flächenbedarf der Buffer abgeschätzt. Der DRCR-Algorithmus erlaubt es, für Verdrahtung und Buffer-Insertion unterschiedliche Regionen zu definieren.

3.4.2 Driver-Sizing

Nach der Erzeugung des Layouts ist die kapazitive Last in jedem Pfad genau bekannt. Zur Verzögerungszeitminimierung können daher die Ausgangs-Gates aller treibenden Transistoren größenmäßig an ihre Last angepasst werden. Da die Länge aller Transistoren in digitalen Designs zumeist gleich ist, beschränkt sich die Größenanpassung auf das Ändern der Gate-Weiten. Ziel des Driver-Sizing ist es, die Gates aller Transistoren genauso weit zu entwerfen, dass alle Timing-Restriktionen erfüllt werden, jedoch kein Platz durch zu weite Gates verschenkt wird.

In [Sundararajan00] wird das Driver-Sizing als ein Optimierungsproblem betrachtet. Dies bietet gegenüber anderen Ansätzen wie z.B. [Chen91] den Vorteil, dass es ein exaktes und schnelles Verfahren ist. Voraussetzung für dieses Verfahren ist lediglich ein Verzögerungszeit-Modell, das monotone Funktionen verwendet. Die Optimierung erfolgt zweistufig. Im ersten, als D-Phase bezeichneten Schritt, werden die Transistorgategrößen als fest und die Transistordelays als variabel betrachtet. Damit hat man in dieser Phase ein Min-Cost-Netzwerkfluss-Problem [Ottmann90] zu lösen. Anschließend folgt die W-Phase, in der die Verzögerungszeiten als fest betrachtet und die Gateweiten berechnet werden.

Die Beschreibung des Sizing-Problems als Optimierungsproblem bietet im Gegensatz zu Heuristiken die Möglichkeit, immer gute Lösungen zu finden. Allerdings hängt der Erfolg dieses Verfahrens von dem zur Verfügung stehenden Platz auf der Layoutfläche für die Sizing-Operationen ab.

3.4.3 Wire-Sizing und Leitungsvertauschung

Bei diesen beiden Verfahren wird versucht, das Zeitverhalten des Leitbahn-Netzwerks durch lokale Änderungen zu verbessern. Bei Standardzelldesigns ohne Verdrah-

tungskanäle und bei nicht voll genutzten Verdrahtungskanälen können einzelne Leitungen verbreitert werden. Damit wird der Widerstand der Leitung geringer und somit auch die Verzögerungszeit. Allerdings muss man dabei beachten, dass mit der Verringerung des Widerstands sowohl Eigen- als auch Kopplungskapazität der Leitung ansteigen. Zusätzlich zu den Modifikationen der Leitungsbreite kann man auch die Abstände oder die Reihenfolge nebeneinander liegender Leitungen verändern. Dabei ändern sich die parasitären Kopplungskapazitäten und das Zeitverhalten kann sich verbessern.

In [Saxena99] wird ein einfacher Algorithmus zur Bestimmung der optimalen Abstände zur Verringerung des Übersprechens zwischen Leitungen verwendet. Dabei wird jede Leitung in so genannte Trunks und Branches zerlegt. Als Trunks bezeichnet man die Verbindungen in einem Verdrahtungskanal, die in Kanalrichtung verlaufen, als Branches, die senkrecht zu den Trunks verlaufenden Leitungssegmente. Zur Verbesserung des Zeitverhaltens werden nur horizontale Trunks - wie in Abbildung 3.19 dargestellt - in ihrer Position verschoben.

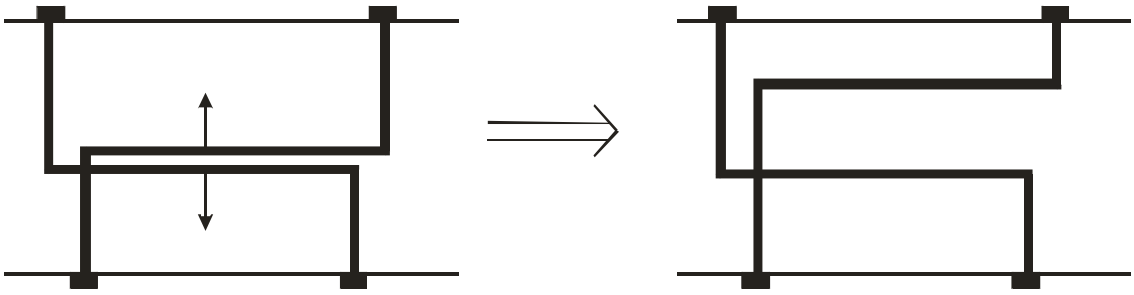

Abbildung 3.19: Trunk-Verschiebung

Für jeden Trunk wird zunächst das so genannte Verschiebungsintervall definiert. Dieses umfasst den Bereich, innerhalb dessen der Trunk verschoben werden kann, ohne einen Kurzschluss mit anderen Trunks zu verursachen. Zusätzlich werden so genannte Basis-Verschiebungsintervalle definiert. Ein Basis-Verschiebungsintervall beginnt und endet an Start- und Endpunkten von benachbarten Trunks, welche für aktuelle Trunks nicht durch andere Leitungen abgeschirmt sind.

Zur Bestimmung des Übersprechens zwischen zwei Leitungen wird jeweils ein Kopplungsfaktor bestimmt. Dieser ist proportional zur Länge, in der die Leitungen nebeneinander liegen und umgekehrt proportional zum Abstand der Leitungen. Vor jedem Iterationsschritt wird eine Liste der Trunks mit den höchsten Übersprechpegeln, entsprechend den höchsten Kopplungsfaktoren, bestimmt. Während jedes Iterationsschritts wird mit dem am stärksten beeinflussten Trunk begonnen und dieser innerhalb des Verschiebungsintervalls verschoben. Die optimale Position innerhalb des Verschiebeintervalls ergibt sich aus Beobachtungen über den Verlauf der Störungen in den einzelnen Intervallen. Wenn die optimale Position für diesen Trunk gefunden ist, werden

die unmittelbar benachbarten Trunks nach gleichem Prinzip verschoben. Anschließend werden alle verschobenen Trunks als „locked“ markiert und in darauf folgenden Schritten nicht mehr verschoben. Anschließend wird der nächste, in der Liste stehende Trunk analog bearbeitet.

Dieser Algorithmus erlaubt eine Verbesserung des Übersprechverhaltens in [Saxena99] um durchschnittlich ca. 15%. In Abbildung 3.19 erkennt man dieses durch eine Verkürzung der Strecke, auf der die koppelnden Leitungen eng nebeneinander liegen. Allerdings sind die Freiheitsgerade bei diesem Vorgehen relativ beschränkt. Es kann weder die Reihenfolge von Leitungen vertauscht werden, noch sind größere Änderungen im Übersprechverhalten zu erwarten, da nur lokale Änderungen durchgeführt werden. Insbesondere bei stark gefüllten Verdrahtungskanälen sind nur geringfügige Änderungen möglich.

4 Simultane kräftebasierte Platzierung und Globalverdrahtung

In diesem Kapitel soll der im Rahmen dieser Arbeit entwickelte platzierungsbasierte Algorithmus zur Lösung des Timing-Closure-Problems erläutert werden.

Das hier vorgestellte Verfahren besteht aus zwei Kernkomponenten: Der erste Teil ist ein Ansatz zur simultanen Platzierung und Globalverdrahtung. Während bei der konventionellen Trennung von Platzierung und Verdrahtung die Modellierung der Verdrahtung mit einer Bounding-Box (vgl. Abschnitt 2.3.2) pro Netz erfolgt, kann in diesem Ansatz die während der Platzierung erzeugte Globalverdrahtung zur Abschätzung der finalen Verdrahtung verwendet werden. Dabei werden für jedes Netz mehrere Bounding-Boxen (vgl. Abschnitt 4.2) zur Modellierung verwendet, was die Genauigkeit der Modellierung erhöht. Zusätzlich berücksichtigt die Globalverdrahtung bereits die Belegung von Verdrahtungsressourcen, was ebenfalls einen Beitrag zur Erhöhung der Modellierungsgenauigkeit leistet.

Die zweite Komponente des neuen Verfahrens ist ein Timing-Driven-Platzierungsansatz. Dieser erweitert das Verfahren zur kräftegesteuerten Platzierung und Globalverdrahtung um eine zusätzliche Kraft, die beim Auftreten von Pfad-Constraint-Verletzungen anziehend auf die Zellen des jeweiligen Pfads wirkt und auf diese Weise die Distanzen zwischen den Zellen verringern kann.

Im folgenden Abschnitt wird zunächst das Prinzip der simultanen kräftegesteuerten Platzierung und Globalverdrahtung beschrieben. Anschließend werden das zum Einsatz kommende Verdrahtungsmodell und die Berechnung der Kräfte beschrieben. Im Abschnitt 4.4.4 wird dann die neu entwickelte Pfadkraft vorgestellt. Das Kapitel endet mit einer Beschreibung der Behandlung von Mehrpunktnetzen.

4.1 Prinzip

Ziel des neuen Platzierungsansatzes ist es, die Platzierung aller Zellen gleichzeitig mit der Globalverdrahtung mit einem kräftegesteuerten Algorithmus durchzuführen.

Die Repräsentation der Globalverdrahtung erfolgt in dieser Arbeit anders als bei den meisten Globalverdrahtern. Während für viele Algorithmen die Layoutfläche in Rechtecke eingeteilt wird und die Globalverdrahtung durch Zuweisung der Netze zu den Layoutteilflächen erfolgt, wird hier eine Globalverdrahtung durch Punktzuweisungen erzeugt. Dazu werden für jedes Netz ein oder mehrere Punkte platziert, durch die später die detaillierte Verdrahtung verlaufen soll. Diese Punkte werden im Folgenden als virtuelle Zellen bezeichnet. Die räumliche Ausdehnung der virtuellen Zellen kann

unterschiedlich sein, so dass eine virtuelle Zelle eine größere rechteckige Layoutteilfläche oder eine Fläche mit genau der Breite einer Leitung repräsentieren kann.

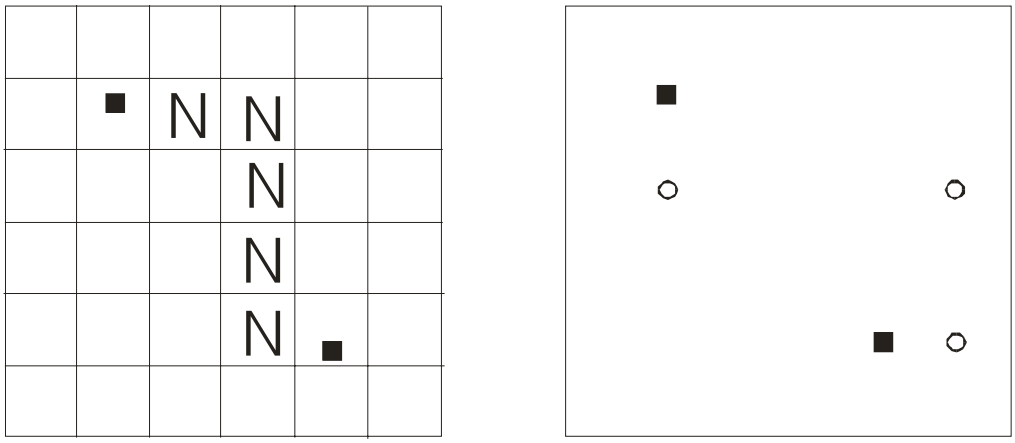

Abbildung 4.1: Repräsentation der Globalverdrahtung

Abbildung 4.1 zeigt - nicht maßstabsgerecht - die unterschiedlichen Repräsentationen der Globalverdrahtung. Dabei kennzeichnet jedes schwarze Quadrat einen Anschlusspin. Im linken Teil der Abbildung erkennt man die Zuweisung der Verdrahtung zu Layoutteilflächen, wobei ein „N“ in einer Layoutteilfläche bedeutet, dass das Netz durch die Teilfläche verlaufen wird. Im rechten Teil der Abbildung 4.1 ist die in dieser Arbeit zum Einsatz kommende Repräsentation dargestellt. Dabei entspricht jeder runde Punkt einer virtuellen Zelle.

Zur Platzierung und Globalverdrahtung in einem Schritt werden auf die Zellen wie auch auf die Elemente der Globalverdrahtung, die hier als Leitungssegmente bezeichnet und in Abschnitt 4.2 genauer beschrieben werden, jeweils zwei Kräfte wirksam.

Vergleichbar zum kräftegesteuerten Platzierungsverfahren aus Abschnitt 3.1.1.1 werden eine anziehende Kraft, die einer Modellierung der Verbindungen zwischen den Elementen entspricht, und eine abstoßende Kraft, die das Übereinanderliegen mehrerer Elemente verhindern soll, zur Berechnung der Platzierung verwendet. Zusätzlich wird eine dritte Kraftkomponente zur Timing-Driven-Platzierung eingeführt.

Das hier vorgestellte Verfahren arbeitet konstruktiv, d.h. es ist keine Initialplatzierung erforderlich. Nach der Erstellung der Initialplatzierung erfolgen so lange iterative Verbesserungen der Platzierung und Globalverdrahtung bis ein Abbruchkriterium erfüllt ist.

4.2 Verdrahtungsmodell

Die Globalverdrahtung in diesem Verfahren basiert auf der Platzierung von Leitungssegmenten. Ein Leitungssegment wird in dieser Phase durch seinen Start- und Endpunkt beschrieben. Alle Start- und Endpunkte der Verdrahtungssegmente eines Netzes repräsentieren, wie in Abschnitt 4.1 beschrieben, die Globalverdrahtung des jeweiligen Netzes. Die physikalische Ausführungsform eines Leitungssegments ist während der Platzierungs- und Globalverdrahtungsphase noch unbekannt und wird erst in der Detailverdrahtungsphase (Kapitel 5) festgelegt.

Alle Zellen einer Schaltung bilden jeweils einen Start- oder Endpunkt eines Leitungssegments. Dieses sind die einzigen Globalverdrahtungsinformationen im initialen Platzierungs- und Globalverdrahtungsschritt. Weitere Leitungssegmente werden im Verlauf der Platzierung und Globalverdrahtung durch das Einfügen von virtuellen Zellen erzeugt. Eine virtuelle Zelle entspricht einer Standardzelle in der Netzliste, besitzt aber eine nur virtuelle räumliche Ausdehnung. Für die Erzeugung eines zusätzlichen Verdrahtungssegments muss also jeweils eine Zelle in die Netzliste und das Layout eingefügt werden. Die Bestimmung der Größe einer virtuellen Zelle im Layout wird im Abschnitt 4.4.2 beschrieben.

In den nachfolgenden Abschnitten werden die folgenden Begriffe zur Beschreibung von Zellen verwendet:

Bei realen Zellen handelt es sich um Bibliothekszellen mit einer logischen Funktion. Virtuelle Zellen stellen die Start- bzw. Endpunkte der Verdrahtungssegmente dar und haben keine logische Funktion. Der Begriff Zellen ist der Oberbegriff für reale und virtuelle Zellen.

Die Längenabschätzung einer Verbindung zwischen zwei realen Zellen erfolgt in dieser Phase wie folgt: Sind zwei reale Zellen über eine oder mehrere virtuelle Zellen verbunden, wird die Verdrahtung mit mehreren Bounding-Boxen abgeschätzt. Die Gesamtlänge der Verbindung entspricht dabei der Summe der halben Umfänge aller Bounding-Boxen.

Sind zwei reale Zellen durch ein sehr kurzes Netz miteinander verbunden, ist dieses nicht durch virtuelle Zellen in mehrere Segmente geteilt. In diesem Fall wird die Länge der Verbindung beider Zellen durch den halben Umfang der umschreibenden Bounding-Box beider Zellen abgeschätzt.

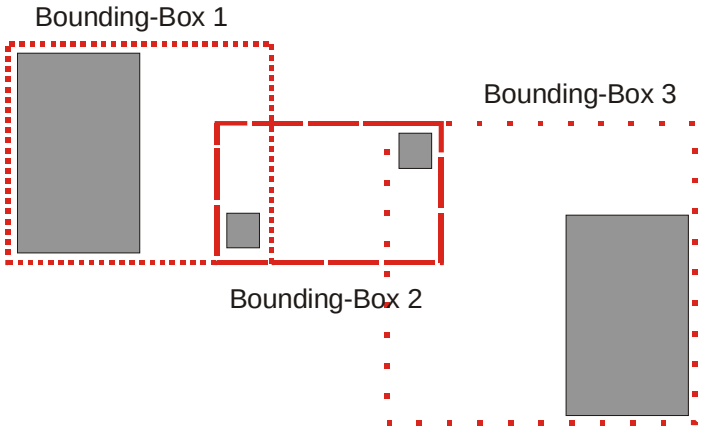

Abbildung 4.2: Verdrahtungslängenmodell

Für das Netz aus Abbildung 4.2, welches aus zwei Standardzellen und zwei virtuellen Zellen besteht, berechnet man die Netzlängenabschätzung zu

$$l_{Net} = \sum_{BoundingBoxen} \frac{1}{2} \cdot Umfang(BoundingBox) = \frac{1}{2}(U_{BB1} + U_{BB2} + U_{BB3}) \; . \qquad \textbf{(4.1)}$$

Diese Abschätzung mit mehreren Bounding-Boxen für jedes Netz erfordert mehr Rechenaufwand als die Abschätzung der Verdrahtungslänge mit nur einer Bounding-Box (vgl. Abschnitt 2.3.2), bietet jedoch eine höhere Genauigkeit.

4.3 Bestimmung der Leitungssegmente

Zur Platzierung der Globalverdrahtungselemente werden die auf die Start- und Endpunkte der Leitungssegmente wirkenden Kräfte berechnet. Die Berechnung von zwei Kräftepaaren pro Verdrahtungssegment stellt sicher, dass nicht nur eine Verschiebung des Segments wie bei der Platzierung von Zellen in jedem Iterationsschritt erfolgt, sondern dass eine Verformung des Verdrahtungselements erfolgen kann.

Abbildung 4.3 zeigt exemplarisch die Verformung eines L-Shapes, das an zwei virtuellen Zellen endet. Die Pfeile im linken Teil der Abbildung symbolisieren dabei die resultierende Kraft auf die jeweilige virtuelle Zelle.

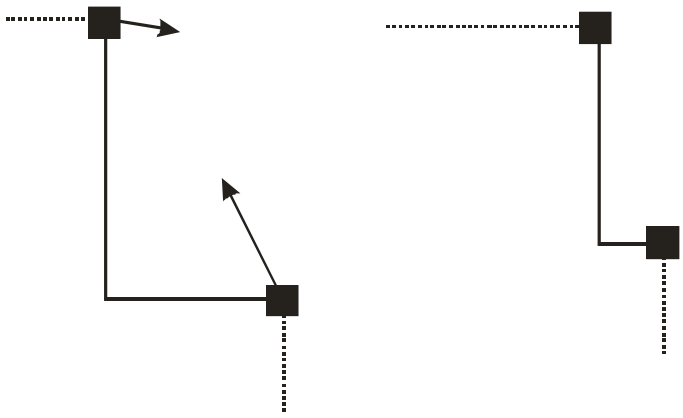

Abbildung 4.3: Verformung eines L-förmigen Verdrahtungselements

Neben der Deformation eines Verdrahtungselementes ist auch ein Wechsel der Form von einem L-förmigen Element zu einem geraden Element möglich.

Das Verfahren zur Berechnung von Verdrahtungssegmenten wird in den folgenden Abschnitten erläutert. Dabei wird zwischen der initialen Berechnung von Verdrahtungselementen, die eine Platzierung ohne Globalverdrahtung voraussetzt, und der Modifikation der Globalverdrahtungselemente unterschieden. Im Modifikationsschritt werden nicht nur neue Verdrahtungselemente erzeugt, sondern gleichzeitig nicht mehr benötigte Segmente entfernt.

4.3.1 Initiale Berechnung von Leitungssegmenten

Im Initialzustand bildet jede Zelle den Anfang bzw. das Ende eines Verdrahtungssegments. Durch diese beiden Punkte ist die Länge des Verdrahtungssegments für die Globalverdrahtung, nicht aber seine geometrische Form definiert.

Ziel der Globalverdrahtung ist eine grobe Festlegung des Verlaufs aller Netze. Dazu werden in Abhängigkeit von der Platzierung zusätzliche Verdrahtungssegmente erzeugt, um die Freiheiten in der Detailverdrahtungsphase durch zusätzliche Globalverdrahtungsfestlegungen einzuschränken. Dazu müssen zunächst Verdrahtungssegmente eingefügt werden.

Während der initialen Berechnung der Verdrahtungssegmente werden so lange neue Segmente für jedes Netz erzeugt, bis die Länge aller Verdrahtungssegmente einen benutzerdefinierten Schwellwert unterschreitet. Zur initialen Längenberechnung der Verdrahtungssegmente wird im Layout jeweils der Abstand zwischen zwei Zellen bestimmt. Anschaulich bestimmt der maximale Abstand zwischen zwei Zellen also die Obergrenze der Länge für ein Verdrahtungssegment.

Durch den gewählten Schwellwert für die maximale Segmentlänge kann der Benutzer die Genauigkeit der Globalverdrahtung festlegen. Dabei ist ein Kompromiss zwischen der Genauigkeit und der Laufzeit des Algorithmus erforderlich. Da das Einfügen jedes Verdrahtungssegments zu einer zusätzlichen (virtuellen) Zelle in der Netzliste führt, steigt die Laufzeit an. Dabei verhält sich eine virtuelle Zelle im Hinblick auf die Laufzeit wie eine reale Zelle, so dass die Laufzeit einer Schaltung mit n realen und v virtuellen Zellen genau der Laufzeit einer Schaltung mit n + v Zellen entspricht. Auf der anderen Seite wird durch jede zusätzliche virtuelle Zelle eine zusätzliche Globalverdrahtungsinformation erzeugt. Je detaillierter die Globalverdrahtung ist, desto weniger Freiheitsgerade gibt es in der Detailverdrahtung. Dadurch steigt die Genauigkeit der Modellierung der Detailverdrahtung während der Globalverdrahtung. Damit wird wiederum die Abweichung zwischen Global- und Detailverdrahtung kleiner und die Verzögerungszeiten können nach der Platzierung und Globalverdrahtung genauer vorhergesagt werden.

Beim Erzeugen von Verdrahtungselementen müssen Modifikationen im Layout und in der Netzliste vorgenommen werden. Dabei wird ein neues Verdrahtungssegment in der Netzliste wie im Layout durch eine virtuelle Zelle repräsentiert. Im Layout beschreibt eine virtuelle Zelle dabei eine Position, durch die später das globale Routing verlaufen soll. In der Netzliste erhält die virtuelle Zelle Verbindungen zu den Zellen, deren Globalverdrahtung sie repräsentieren soll.

Die initiale Platzierung einer neu eingefügten Zelle im Layout erfolgt auf der direkten Verbindung zwischen den beiden anderen beteiligten Zellen.

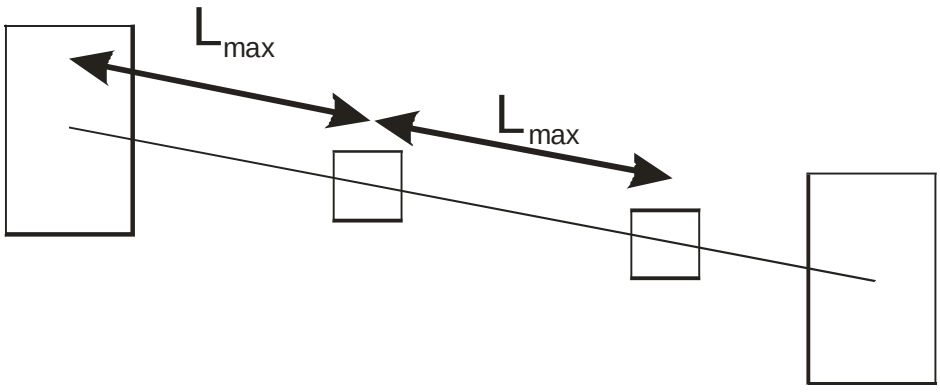

Abbildung 4.4: Einfügen von virtuellen Zellen im Layout

Abbildung 4.4 stellt das Einfügen virtueller Zellen im Layout dar. Sie werden jeweils im Abstand des Verdrahtungssegment-Schwellwerts L_{max} platziert. Das jeweils letzte Segment eines Netzes wird dadurch im Allgemeinen kürzer als der Schwellwert sein.

Das Einfügen der virtuellen Zelle in die Netzliste ist in Abbildung 4.5 dargestellt.

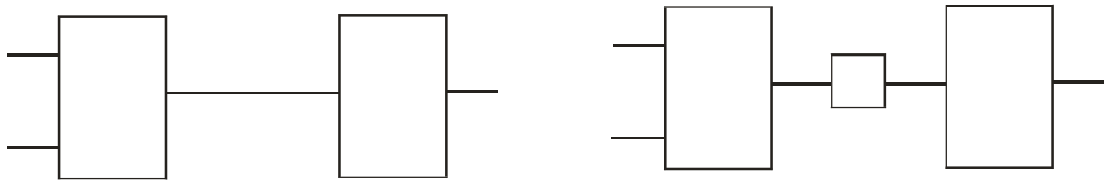

Abbildung 4.5: Netzlistenänderungen beim Einfügen virtueller Zellen

Zum Einfügen einer virtuellen Zelle wird die Verbindung zwischen den im Ausgangszustand verbundenen Zellen aufgetrennt. Die virtuelle Zelle wird anschließend mit beiden Zellen verbunden.

Dieses Vorgehen steht im Gegensatz zu anderen Timing-Driven-Platzierungsansätzen [Chou02], die zusätzliche virtuelle Verbindungen in die Netzliste einfügen, während in diesem Ansatz die Originalverbindung beim Einfügen zusätzlicher Zellen vollständig entfernt wird. Mit dem hier verwendeten Verfahren lässt sich jedoch bei geeigneter Wahl der Netzgewichte erreichen, dass die virtuelle Zelle innerhalb der durch die beiden Zellen aufgespannten Bounding-Box platziert wird, was bei dem Erhalt der Originalverbindung zwischen den Standardzellen nicht möglich ist.

Aus der modifizierten Netzliste müssen die Kostenmatrix $\underline{C}$ und die Matrix $\underline{d}$ gemäß Gleichung 3.2 neu berechnet werden. Ihre vollständige Berechnung hat eine Laufzeitkomplexität von O(k), wenn k die Anzahl der Netze ist. Zur Reduktion der Laufzeit wird beim Einfügen einer virtuellen Zelle nicht die ganze Matrix neu berechnet, sondern nur eine inkrementelle Änderung vorgenommen.

Dazu werden zunächst die Größe der Matrix $\underline{C}$ und der Matrix $\underline{d}$ angepasst, indem jeweils eine x-Komponente an der Position n+1 und eine y-Komponente an der Position 2n+2 eingefügt werden. Da dieser Einfügeschritt nur der Größenanpassung der Elemente dient, sind alle eingefügten Komponenten für $\underline{C}$ und $\underline{d}$ null.

Anschließend werden die Einträge, die die im Ausgangszustand vorhandene Verbindung zwischen den Zellen i und j darstellen, entfernt. Dazu sind die folgenden Operationen notwendig, wenn es sich bei den Zellen i und j nicht um Padzellen handelt:

$$c_{ii} = c_{ii} - w_k \;\; bzw. \;\; c_{jj} = c_{jj} - w_k \tag{4.2}$$

$$c_{ij} = c_{ij} + w_k \;\; bzw. \;\; c_{ji} = c_{ji} + w_k \tag{4.3}$$

w_k ist das Netzgewicht der Verbindung zwischen den Zellen i und j. In dem hier gewählten Verfahren ist das Netzgewicht für alle Netze im Initialzustand gleich groß.

Nach dem Zurücksetzen der Einträge zwischen den Zellen i und j erfolgt das Eintragen der neuen Verbindungen zwischen den Zellen i und v bzw. v und j. Dabei erhalten alle Verbindungen ein um den Faktor a_v höheres Netzgewicht als die Netze im Initialzustand.

Das höhere Netzgewicht $a_v \cdot w_k$ entspricht einer stärkeren anziehenden Kraft zwischen den am betreffenden Netz beteiligten Zellen. Diese soll sicherstellen, dass die virtuelle Zelle v in räumlicher Nähe zu den Zellen i und j platziert wird. Anschaulich gesehen kann man den Faktor a_v als Maß für die Netzlänge zwischen den Zellen i und j interpretieren. Dabei wird die Verbindung zwischen den Zellen i und j umso kürzer, je größer der Faktor a_v ist.

Folgende Einträge werden beim Einfügen der zwei neuen Verbindungen in der Matrix $\underline{C}$ vorgenommen:

$$c_{ii} = c_{ii} + a_V \cdot w_k \;\; bzw. \;\; c_{jj} = c_{jj} + a_V \cdot w_k \qquad \textbf{(4.4)}$$

$$c_{ij} = c_{ij} - a_V \cdot w_k \;\; bzw. \;\; c_{ji} = c_{ji} - a_V \cdot w_k \qquad \textbf{(4.5)}$$

Die in den Gleichungen 4.2 bis 4.5 dargestellten Änderungen beschreiben die Modifikation der x-Komponenten der Kostenmatrix. Analog sind ebenfalls die y-Komponenten zu modifizieren.

In der Matrix $\underline{d}$ werden an den Positionen n+1 bzw. 2n+2 Nullen eingefügt, da dieser Vektor die nicht verschiebbaren Zellen repräsentiert und virtuelle Zellen prinzipiell immer verschiebbar sind.

4.3.2 Dynamische Modifikation der Leitungssegmente

Die Längen der Verdrahtungssegmente sind wie in Abschnitt 4.3.1 beschrieben ein Maß für den Detailliertheitsgrad der Globalverdrahtung. Da sich durch die kräftegesteuerte Verschiebung aller Zellen neben den Positionen auch die Längen der Verdrahtungssegmente verändern, ändert sich auch die Genauigkeit der Globalverdrahtung. Um dieses zu verhindern, soll die Abweichung der Längen aller Verdrahtungssegmente von der vom Benutzer festgelegten Segmentlänge klein sein. Wird die Segmentlänge zu groß, werden die Modellierung der belegten Verdrahtungsressourcen und die Modellierung der Verzögerungszeiten sehr ungenau. Sinkt die Segmentlänge deutlich unter den gegebenen Schwellwert, steigt die Laufzeit durch eine höhere als vom Benutzer gewählte Genauigkeit an.

Außerdem ist es möglich, dass verschiedene Bereiche eines Layouts nach mehreren Iterationsschritten eine sehr unterschiedliche durchschnittliche Verdrahtungssegmentlänge haben. Dadurch wäre die Modellierungsgenauigkeit in unterschiedlichen Bereichen des Layouts unterschiedlich hoch, was nicht erwünscht ist.

Zur Vermeidung dieser Probleme erfolgt nach einer festgelegten Anzahl von Platzierungsschritten eine Anpassung der Verdrahtungssegmente. Dabei werden Verdrahtungssegmente entfernt, die den Einfügeschwellwert der Segmentlänge um mehr als 25% unterschreiten. Verdrahtungssegmente, die die maximale Segmentlänge überschreiten, werden in zwei oder mehr Segmente geteilt. Während das Einfügen wie bei der initialen Erzeugung von Verdrahtungssegmenten abläuft, entspricht das Entfernen eines Verdrahtungssegments dem Löschen eines virtuellen Knotens aus der Netzliste und dem Layout.

Sowie ein Leitungssegment die maximal zulässige Länge überschreitet, wird ein virtueller Knoten eingefügt. Durch den neuen Knoten und die höheren Netzgewichte der mit diesem Knoten verbundenen Netze werden die beiden mit dem Knoten verbundenen Zellen aufeinander zu bewegt. Dadurch kann der Schwellwert der Verdrahtungssegmentlänge unterschritten werden und der Knoten wird wieder entfernt. Im folgenden Platzierungsschritt kann sich die Segmentlänge durch die abstoßenden Kräfte dann wieder erhöhen und ein neuer Knoten müsste eingefügt werden. Diese Oszillation kann durch deutlich unterschiedliche Schwellwerte für das Einfügen und Verschmelzen von virtuellen Knoten verhindert werden.

Eine Modifikation der Verdrahtungssegmente findet nicht in jedem Iterationsschritt, sondern nur nach einer einstellbaren Anzahl von Schritten statt. Dies ist aus zwei Gründen sinnvoll. Zum einen können Stabilitätsprobleme auftreten, wenn die Netzliste und damit die anziehenden Kräfte in jedem Schritt stark verändert werden. Zum anderen steigt der Anteil der Programmlaufzeit für die Einfüge- und Löschoperationen im Vergleich zur Gesamtlaufzeit deutlich an. Außerdem hat sich gezeigt, dass die Anzahl der Netzlistenmodifikationen bei Durchführung in jedem Iterationsschritt relativ klein ist. Eine sinnvolle Anzahl von Schritten, nach denen Einfüge- und Löschoperationen durchgeführt werden sollten, liegt bei ungefähr zehn.

4.4 Berechnung der Kräfte

Für die Verschiebung der Zellen in jedem Iterationsschritt werden insgesamt fünf unterschiedliche Kräfte berechnet. Auf reale und virtuelle Zellen wirken jeweils eine anziehende und eine abstoßende Kraft. Dabei werden die anziehenden und die abstoßenden Kräfte für reale und virtuelle Zellen unterschiedlich berechnet. Auf alle Elemente kann zusätzlich eine Pfadkraft wirken, die zur Einhaltung der Pfad-Constraints dienen soll.

Die Berechnung der einzelnen Kräfte wird in den folgenden Abschnitten vorgestellt.

4.4.1 Kräfte auf reale Zellen

Die Berechnung der anziehenden Kraft auf jede Zelle erfolgt wie bei der konventionellen kräftegesteuerten Platzierung gemäß Gleichung 3.2. Dabei haben Netze, an denen eine virtuelle Zelle beteiligt ist, ein um den Faktor a_v höheres Netzgewicht als alle anderen Netze (vgl. Abschnitt 4.3.1).

Die Berechnung der abstoßenden Kräfte basiert, wie in Kapitel 3 beschrieben, auf der Zelldichte. Um eine gleichmäßige Verteilung von realen und virtuellen Zellen zu erhalten, ist die Berechnung von zwei unterschiedlichen Dichtefunktionen erforderlich. Mit der ersten Dichtefunktion wird die Dichte der realen Zellen berechnet, mit der zweiten die Dichte der virtuellen Zellen. Um zu verhindern, dass die Globalverdrahtung eine gleichmäßige Zellverteilung der realen Zellen stört, wird zur Berechnung der abstoßenden Kräfte auf die realen Zellen die erste Dichtefunktion verwendet.

Bei der Berechnung der abstoßenden Kraft aus der Zelldichte muss das Integral aus Gleichung 3.6 gelöst werden. Diskretisiert man die Layoutfläche in $g \times g$ Rasterelemente, wird aus dem Integral die Doppelsumme

$$\vec{f}_r(x_i, y_i) = \frac{k}{2\pi} \sum_{x'=0}^{g} \sum_{y'=0}^{g} D(x', y') \frac{\vec{r}_i - \vec{r}\,'}{\left|\vec{r}_i - \vec{r}\,'\right|^2} \quad , \tag{4.6}$$

wobei x und y von 0 bis zur Breite bzw. Höhe der Layoutfläche laufen und $\vec{r}_i = (x_i, y_i)$ sowie $\vec{r}\,' = (x', y')$ ist.

Die direkte Berechung der Doppelsumme über jeweils g Rasterelemente hat eine Laufzeitkomplexität von g^2. Da dies bei größeren Schaltungen zu sehr hohen Laufzeiten führen würde, muss ein effizientes Verfahren zur Berechnung der Doppelsumme gefunden werden.

Betrachtet man Gleichung 4.6 genauer, so stellt man fest, dass diese mathematisch gesehen der Faltung von

$$D(x', y')$$

mit

$$\vec{G}(x, y) = \frac{\begin{bmatrix} x - x' \\ y - y' \end{bmatrix}}{(x - x')^2 + (y - y')^2} \tag{4.7}$$

entspricht, wobei man $\vec{G}$ als Green'sche Funktion bezeichnet.

Einer Faltung im Zeitbereich entspricht eine Multiplikation im Frequenzbereich, wobei in diesem Zusammenhang die Ausgangskoordinaten als Zeitbereich verstanden werden. Um dieses auszunutzen, werden sowohl die Dichtefunktion als auch die Abstandsfunktion $\vec{G}$ mit Hilfe einer Fast-Fourier-Transformation (FFT) zunächst in den Frequenzbereich transformiert. Dann werden die Fouriertransformierten von D und $\vec{G}$ miteinander multipliziert. Das Ergebnis der Multiplikation ist die gesuchte abstoßende Kraft im Frequenzbereich. Durch eine inverse Fouriertransformation kann man abschließend die abstoßende Kraft im Zeitbereich berechnen. Der gesamt Ablauf der Kraftberechnung unter Verwendung einer FFT ist in Abbildung 4.6 dargestellt.

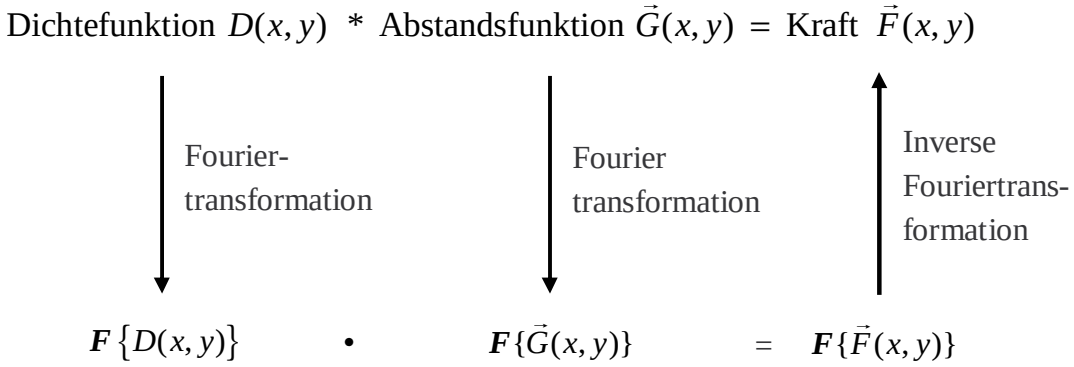

Abbildung 4.6: Ablauf der Kraftberechnung

Unter Verwendung effizienter Algorithmen für die Fast-Fourier-Transformation kann eine Lösung für Gleichung 4.6 mit einer Laufzeitkomplexität von $n \cdot \log n$ berechnet werden.

Eine weitere Reduktion des Rechenaufwands kann erreicht werden, indem die Abstandsfunktion $\vec{G}$ nur einmal für jedes Rasterelement berechnet wird, da sich diese Funktion während der verschiedenen Platzierungsiterationsschritte nicht verändert. Die Fouriertransformation der Dichtefunktion muss in jedem Iterationsschritt neu berechnet werden, da sich die Dichtefunktion in jedem Iterationsschritt verändert.

4.4.2 Kräfte auf Leitungssegmente

Zur Durchführung der Globalverdrahtung werden ebenfalls anziehende und abstoßende Kräfte berechnet. Diese Kräfte wirken jeweils auf den Start- und den Endpunkt eines Leitungssegments.

Wie bereits in Abschnitt 4.4.1 beschrieben, unterscheidet sich die Berechnung der anziehenden Kraft auf eine virtuelle Zelle nur um einen konstanten Faktor von der Kraft auf reale Zellen. Daher erfolgt die Kraftberechnung wie bei den realen Zellen beschrieben.

Die Berechnung der abstoßenden Kräfte soll bei den virtuellen Zellen ebenfalls auf einer Dichtefunktion basieren. Dazu ist den virtuellen Zellen zunächst eine geometrische Ausdehnung zuzuweisen. Virtuelle Zellen haben generell eine quadratische Form. Die Größe wird durch einen benutzerdefinierten Parameter gesteuert und sollte im Regelfall zwischen der Breite von zehn Leitungen und der halben Breite der kleinsten Standardzelle liegen. Diese Größe verhindert einerseits zu große abstoßende Kräfte zwischen virtuellen Zellen im Vergleich zu den abstoßenden Kräften zwischen realen und virtuellen Zellen und stellt andererseits sicher, dass die Genauigkeit der Globalverdrahtung ausreichend hoch ist.

Für die Dichtefunktion zur Berechnung der abstoßenden Kräfte auf virtuelle Zellen bestehen folgende Anforderungen:

Zum einen soll es möglichst wenige Überlappungen zwischen den virtuellen Zellen geben. Dieses entspricht der Vermeidung von überbelegten Verdrahtungsbereichen. Zum anderen soll es möglichst wenige Überlappungen mit überbelegten Bereichen durch reale Zellen geben.

Zur Verhinderung von Überlappungen zwischen virtuellen Zellen wird eine abstoßende Kraft aus der Dichtematrix aller virtuellen Zellen analog zur abstoßenden Kraft für reale Zellen gemäß Gleichung 4.6 berechnet. Gleichzeitig wird die zuvor berechnete Dichtematrix aller realen Zellen benutzt, um abstoßende Kräfte von realen auf virtuelle Zellen zu berechnen.

Die Berechnung der resultierenden abstoßenden Kraft erfolgt gemäß Gleichung 4.8 aus der Summe beider Kräfte, wobei die von den realen Zellen verursachte abstoßende Kraft mit einem Faktor von $r < 1$ gewichtet wird.

$$\vec{F}_{R,total} = \vec{F}_{R,virtuell} + r \cdot \vec{F}_{R,real} \qquad \textbf{(4.8)}$$

Je größer r gewählt wird, desto mehr werden die Leitungspunkte zwischen den Zellen und nicht über den Zellen platziert.

4.4.3 Berechnung der initialen abstoßenden Kraft auf virtuelle Zellen

Die abstoßende Kraft auf alle Zellen berechnet sich jeweils aus der Summe der abstoßenden Kraft des vorherigen Platzierungsschritts und der aktuell berechneten abstoßenden Kraft (vgl. Abschnitt 3.1.1.1).

Neu eingefügte virtuelle Zellen haben aus dem vorangegangenen Platzierungsschritt eine abstoßende Kraft von null. Dies führt dazu, dass diese Zellen zunächst verstärkt von den mit ihnen verbundenen Zellen angezogen werden. Erst nach einer gewissen Anzahl von Iterationen und damit verbundenen Summationen von abstoßenden Kraftbeiträgen wird die resultierende abstoßende Kraft so groß, dass sich die Zellen aus überbelegten Layoutbereichen weg bewegen können.

Um die zusätzlichen Schritte bis zu einer sinnvollen Platzierung einer neu eingefügten virtuellen Zelle zu reduzieren, wird eine initiale abstoßende Kraft berechnet. Diese soll so groß sein, dass die Zelle zum Zeitpunkt des Einfügens im Gleichgewichtspunkt liegt.

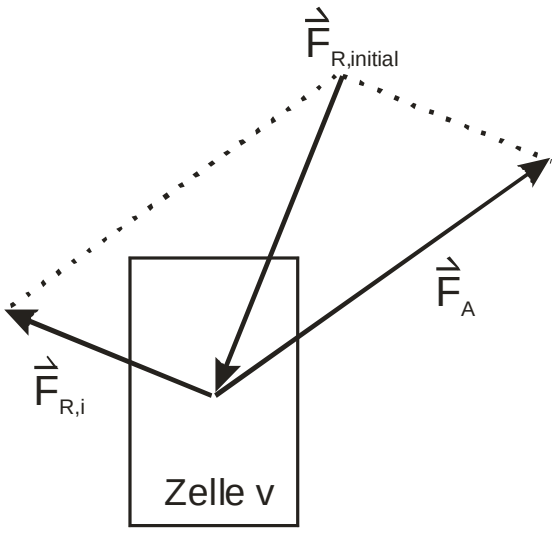

Abbildung 4.7: Berechnung der initialen abstoßenden Kraft

In Abbildung 4.7 ist die Berechnung der initialen Kraft visualisiert. F_A ist in dieser Darstellung die resultierende anziehende Kraft auf die Zelle v. $F_{R,i}$ ist der für diesen i-ten Platzierungsschritt berechnete Beitrag zur abstoßenden Kraft. Damit sich die Zelle im Gleichgewicht befindet, ist eine Zusatzkraft $F_{R,initial}$ erforderlich, die aus

$$\sum_i \vec{F}_{i,Zelle} = 0 \qquad \textbf{(4.9)},$$

d. h.

$$\vec{F}_A + \vec{F}_{R,i} + \vec{F}_{R,initial} = 0 \qquad \textbf{(4.10)}$$

zu

$$\vec{F}_{R,initial} = -\vec{F}_A - \vec{F}_{R,i} \quad \textbf{(4.11)}$$

berechnet wird. Die gemäß Gleichung (4.11) berechnete Initialkraft wird der neu eingefügten Zelle als abstoßende Kraft des vorangegangenen Platzierungsschritts zugewiesen und dazu an der Position der neu eingefügten Zelle in den Vektor der abstoßenden Kräfte eingetragen. Sie stellt sicher, dass ein im folgenden Schritt berechneter abstoßender Kraftbeitrag die Zelle verschieben kann.

4.4.4 Pfadkräfte

Wie in Abschnitt 3.1.1.1 beschrieben, berücksichtigt die kräftegesteuerte Platzierung keine Timing-Constraints. Anstatt diese in den Netzgewichten wie in anderen Ansätzen (vgl. Abschnitt 3.1.1.2) zu berücksichtigen, wird hier ein neuer Weg eingeschlagen und alle Netzgewichte werden auf den gleichen Wert initialisiert. Die Pfad-Constraints werden dafür in Form einer Pfadkraft bei der Platzierung berücksichtigt.

4.4.4.1 Eigenschaften der Pfadkraft

Zur Erzeugung einer Platzierung, die alle geforderten Laufzeitbedingungen einhält, wird eine Pfadkraft verwendet. Diese Kraft soll die folgenden Eigenschaften haben:

- Die Pfadkraft wirkt anziehend in Richtung des Pfadschwerpunkts, wenn die Constraints des Pfades verletzt sind.
- Die Pfadkraft wirkt anziehend, wenn die geschätzte Pfadverzögerung exakt dem Pfad-Constraint entspricht.
- Die Pfadkraft wirkt abstoßend, wenn das Pfad-Constraint deutlich übererfüllt ist.
- Der Betrag der Pfadkraft ist von der Größe der Zellen unabhängig.
- Der Betrag der Pfadkraft ist eine Funktion der Differenz zwischen geschätzter Pfadverzögerung und Pfad-Constraint.
- Die Pfadkraft wirkt auf reale und virtuelle Zellen gleich.

4.4.4.2 Richtung der Pfadkraft

Durch den Einsatz einer Pfadkraft soll das Auftreten von Laufzeitverletzungen durch zu weit voneinander entfernt platzierte Zellen eines Pfads verhindert werden. Dieses wird erreicht, indem alle Zellen eines Pfades durch die Pfadkraft auf den Schwerpunkt des Pfades und damit aufeinander zu bewegt werden.

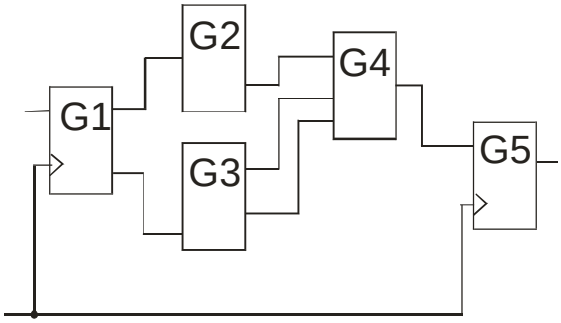

Abbildung 4.8: Schaltungsbeispiel

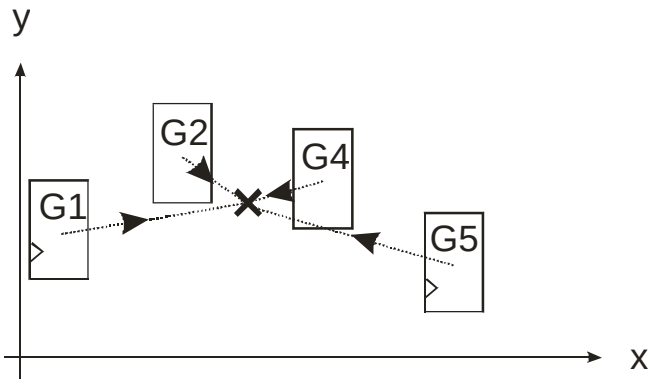

Abbildung 4.9: Richtung der Pfadkraft

Die Abbildungen 4.8 und 4.9 zeigen ein Schaltungsbeispiel und die dazugehörige Richtung der Pfadkraft. In dem Beispiel gibt es zwei Pfade, die von den Gattern G1, G2, G4 und G5 bzw. G1, G3, G4 und G5 gebildet werden. In Abbildung 4.9 wird die Pfadkraft für den Pfad durch die Gatter G1, G2, G4 und G5 dargestellt.

Zur Bestimmung der Kraftrichtung errechnet man den Pfadschwerpunkt mit

$$x_s = \frac{\sum_{Zellen\ i} x_i}{n_{Zellen}} \quad bzw. \quad y_s = \frac{\sum_{Zellen\ i} y_i}{n_{Zellen}}. \qquad \textbf{(4.12)}$$

Die Kraft auf die Zelle i hat dann die Richtung $\begin{pmatrix} x_i - x_s \\ y_i - y_s \end{pmatrix}$.

4.4.4.3 Betrag der Pfadkraft

Die Pfadkraft soll für kürzere Verbindungen zwischen den Zellen eines Pfades sorgen, wenn das Timing-Constraint für einen Pfad überschritten ist. Dabei müssen die Zellen eines Pfades zur Einhaltung des Constraints umso weiter bewegt werden, je größer die Überschreitung der Laufzeitbedingung ist. Deshalb wird die Pfadkraft zu

$f_{Pfad} = f(t_{ist} - t_{soll})$ gewählt. Dabei beschreibt t_{ist} die aus Gatter- und Leitungsverzögerung bestehende geschätzte Verzögerungszeit des Pfades. t_{soll} ist das Constraint für den betrachteten Pfad und entspricht in den meisten Fällen $t_{soll} = \frac{1}{f_{clock}}$, wenn f_{clock} die Taktfrequenz des betrachteten Systems ist.

Zur Berechung der Pfadverzögerungszeiten wird ein einfaches RC-Modell unter Berücksichtigung von Gatter- und Leitungsverzögerungen verwendet. Der Widerstand und die Kapazität einer Leitung werden näherungsweise als proportional zur Leitungslänge angenommen. Die Abschätzung der Leitungslänge erfolgt mit einem Bounding-Box-Modell wie in Abschnitt 2.2.2 beschrieben. Die Verzögerungszeit des in Abbildung 4.8 dargestellten Pfads durch die Gatter G1, G2, G4 und G5 ergibt sich mit diesem Modell zu:

$$\begin{aligned} t_{Delay} = &(R_{G1} + R_{L12}) \cdot (C_{L12} + C_{G2}) \\ &+ (R_{G2} + R_{L24}) \cdot (C_{L24} + C_{G4}) \\ &+ (R_{G4} + R_{L45}) \cdot (C_{L45} + C_{G5}) \end{aligned} \quad \textbf{(4.13)}$$

Anschließend wird die Differenz $t_{Pfad} = t_{soll} - t_{Delay}$ berechnet. Die Differenz aus t_{soll} und t_{Delay} wird auch als Slack bezeichnet.

Ist t_{Pfad} negativ, liegt eine Verletzung des Pfad-Constraints vor. Für diesen Fall muss eine anziehende Pfadkraft berechnet werden. Für den Fall, dass t_{Pfad} positiv ist, wird das gegebene Pfad-Constraint erfüllt. Für diesen Fall könnte die Pfadkraft - wie in Abbildung 4.10 dargestellt - null sein.

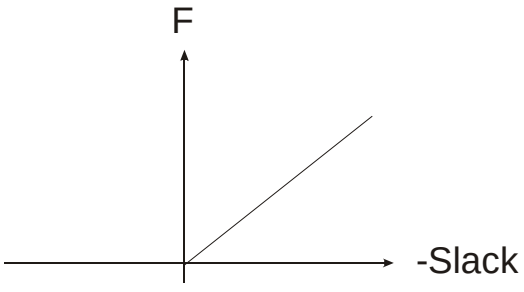

Abbildung 4.10: Ideale Pfadkraft

Es hat sich jedoch gezeigt, dass es durch die zusätzlichen anziehenden Kräfte in Pfaden mit Constraint-Verletzungen, eine Tendenz gibt, viele Zellen ins Zentrum zu verschieben. Dies führt zu vielen Zellüberlappungen und einer sehr ungleichmäßigen Belegung der Layoutfläche. Deshalb wird die Pfadkraft um eine abstoßende Komponente erweitert. Für alle Pfade, in denen die geforderte Signallaufzeit deutlich

unterschritten wird, wird eine abstoßende Pfadkraft eingeführt. Diese wirkt exakt umgekehrt zur anziehenden Pfadkraft und verteilt die Zellen eines Pfades daher auf eine größere Fläche. Der Verlauf einer anziehenden und abstoßenden Pfadkraftfunktion ist in Abbildung 4.11 dargestellt.

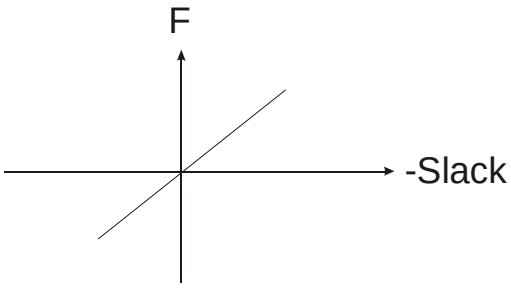

Abbildung 4.11: Pfadkraft mit anziehenden und abstoßender Kraft

Des Weiteren hat sich gezeigt, dass der Übergang von anziehender zu abstoßender Kraft nicht genau bei $t_{Delay} = t_{Soll}$, entsprechend der Nullstelle der Pfadkraft im Ursprung, liegen darf. In diesem Fall wirken auf Zellen in Pfaden, deren Constraint nur minimal unterschritten wird, abstoßende Kräfte. Dadurch werden die Zellen auseinander gerückt und die Laufzeitbedingung des Pfades wird wieder überschritten. Im darauf folgenden Iterationsschritt wirken dann wieder anziehende Kräfte und die Zellen werden wieder zusammen geschoben.

Um diese Oszillation zu vermeiden, darf die Pfadkraft - wie in Abbildung 4.12 dargestellt - erst bei einer deutlichen Unterschreitung des Constraints abstoßend wirken.

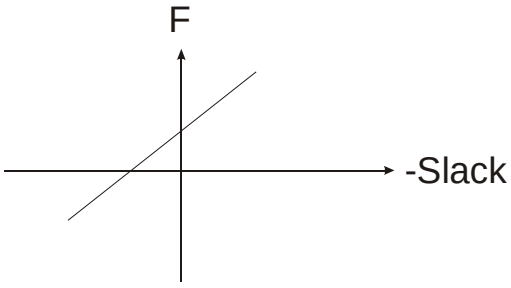

Abbildung 4.12: Verbesserter Verlauf der Pfadkraft

Erste Versuche mit dieser Pfadkraftfunktion haben gezeigt, dass die Anzahl der Pfade mit Timing-Violations durch die Verwendung der Pfadkraft zwar abnimmt, aber es trotzdem noch Pfade mit Timing-Verletzungen existieren. Es konnte herausgefunden werden, dass vorwiegend Pfade mit sehr starken Überschreitungen der Pfad-Constraints diese nach Abschluss der Platzierung immer noch verletzen. Gleichzeitig hat sich

gezeigt, dass die abstoßenden Kräfte kurz nach dem Erreichen der Constraints zu hoch waren.

Aus diesem Grund wird schließlich ein Polynom dritten Grades zur Berechnung der abstoßenden Kraft benutzt. Dieses bietet den Vorteil, bei großen Abweichungen von den Constraints große anziehende Kräfte zu erzeugen.

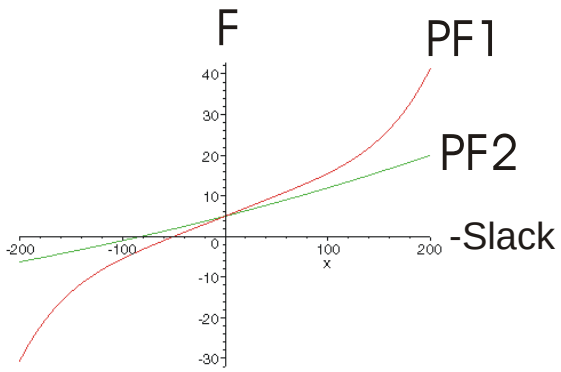

Abbildung 4.13: Lineare und polynomiale Pfadkraft

Abbildung 4.13 zeigt die lineare (PF2) und die polynomiale (PF1) Pfadkraft in einem Diagramm zum Vergleich. Aufgrund der besseren Platzierungsergebnisse kommt in dieser Arbeit nur die nichtlineare Berechnungsvorschrift zum Einsatz.

Die Pfadkraft wird in jedem Iterationsschritt der kräftebasierten Platzierung jeweils nach der abstoßenden Kraft berechnet und in das zu lösende Gleichungssystem (3.5) eingefügt. Damit ergibt sich die Gleichung

$$\underline{C} \cdot \vec{p} + \vec{d} + \vec{e} + \vec{g} = 0 \tag{4.14}$$

zur Berechnung der Platzierung, wobei

$$g_i = f_{path}(x_i, y_i) \tag{4.15}$$

ist.

4.5 Bewertung einer Platzierung

Um die Einhaltung aller Pfad-Constraints zu erreichen, wird nach jedem Platzierungsiterationsschritt die Anzahl der Pfade mit Constraint-Violations bestimmt.

Ist die Anzahl der Pfade mit nicht eingehaltenem Timing gleich null, wird die Platzierung und Globalverdrahtung beendet. Solange es noch Pfade mit Constraint-

Verletzungen gibt und die vom Benutzer vorgegebene Anzahl von Iterationen nicht überschritten ist, werden neue Platzierungen berechnet. Der Betrag der Pfadkraft kann kontinuierlich mit der Anzahl der Iterationen erhöht werden, um sicherzustellen, dass die Anzahl der Constraint-Verletzungen reduziert wird.

4.6 Behandlung von Mehrpunktnetzen

Die Modellierung von Mehrpunktnetzen wird an zwei Stellen benötigt. Zum einen müssen zur Berechnung der Pfadkräfte Mehrpunktnetze bei der Pfadlängenberechnung berücksichtigt werden und zum anderen ist eine Längenabschätzung der Mehrpunktnetze bei der Berechnung der Gesamtverdrahtungslänge erforderlich.

Für die Berechnung der Pfadkräfte wird jedes Mehrpunktnetz in mehrere Zweipunktnetze zerlegt. Dadurch existieren für das hier vorgestellte Platzierungs- und Globalverdrahtungswerkzeug innerhalb eines Pfads keine Verzweigungen, sondern jeder Pfad ist eine lineare Abfolge von Zellen. Dieses führt dazu, dass beispielsweise ein Pfad mit einem Vier-Punkt-Netz in drei Pfade zerlegt wird, die anschließend unabhängig voneinander betrachtet werden.

Um zu anderen Platzierungswerkzeugen vergleichbare Ergebnisse berechnen zu können, erfolgt die Berechnung der Gesamtverdrahtungslänge nicht auf der Basis der zuvor erzeugten Zweipunktnetze, sondern wird - wie in vielen anderen Platzierungswerkzeugen - mit dem Bounding-Box-Modell abgeschätzt.

5 Bibliotheksbasierte Detailverdrahtung

Fast alle Verdrahtungsverfahren (vgl. Abschnitt 3.1.2), die heute industriell eingesetzt werden, basieren auf einer Wegesuche. Dadurch ist es vor der Ausführung der Detailverdrahtung schwierig, sichere Aussagen über die Länge eines Netzes und damit auch über die Länge eines Pfades zu treffen. Eine Abschätzung der Verdrahtungslänge kann im Falle einer sehr hohen Verdrahtungsdichte durch lange Umwege zu den in Abschnitt 3.2 beschriebenen Problemen bei der separaten Platzierung und Verdrahtung führen. Einige kommerzielle Verdrahtungsverfahren - wie zum Beispiel der Router im Silicon Ensemble der Firma Cadence - beschränken deshalb bewusst die Suche nicht auf legale Wege, sondern erzeugen unter Umständen auch Kurzschlüsse zu anderen Leitungen. Diese müssen dann in einer folgenden - meist als Rip-Up & Reroute bezeichneten - Phase beseitigt werden.

Um den Vorteil der genauen Laufzeitmodellierung durch die simultane Platzierung und Globalverdrahtung nicht zu verlieren, ist für den leitbahnorientierten Designflow ein Detailverdrahtungsverfahren erforderlich, das Verbindungen mit gut vorsehbaren Längen und damit genau abschätzbaren Leitungsverzögerungszeiten erzeugt. Dadurch wird eine Konvergenz zwischen dem in der Platzierungs- und Globalverdrahtungsphase abgeschätzten Timing und dem Timing nach der Layoutfertigstellung erreicht.

Der neu entwickelte Detailverdrahtungsalgorithmus setzt anstelle der Wegesuche vordefinierte Verdrahtungsformen (Shapes) aus einer Bibliothek ein. Dadurch wird die Wegesuche durch eine Platzierung von Leitungselementen mit bekannter Verzögerungszeit aus einer Bibliothek ersetzt.

Ausgehend von einer zufälligen Initialverdrahtung wird eine iterative Verbesserung der Verdrahtung mit einem Optimierungsverfahren durchgeführt. Dazu wird in jedem Schritt eine Leitung auf der Layoutfläche ausgewählt und durch ein neues Verdrahtungs-Shape aus der Leitungselemente-Bibliothek ersetzt. Zur Entscheidung, ob die durchgeführte Verdrahtungsänderung akzeptiert wird, wurden zwei Optimierungsverfahren implementiert. Während die monotone Suche nur solche Verdrahtungsänderungen akzeptiert, die eine Verbesserung im Sinne der Kostenfunktion darstellen, akzeptiert das aus der Platzierung adaptierte Simulated-Annealing-Verfahren unter bestimmten Bedingungen auch eine Verschlechterung der Kostenfunktion. In Abhängigkeit von der Akzeptanzentscheidung wird die zuvor durchgeführte Verdrahtungsmodifikation akzeptiert oder wieder zurückgenommen.

Der Schwerpunkt dieser Arbeit liegt bei der simultanen Platzierung und Globalverdrahtung. Das im Rahmen dieser Arbeit eingesetzte und in diesem Kapitel beschriebene Detailverdrahtungsverfahren wurde implementiert, um zur Ergebniserzeugung einen vollständigen leitbahnorientierten Designflow zu erhalten. Daher bietet das

Detailverdrahtungsverfahren an einigen Stellen noch Erweiterungs- und Verbesserungspotenzial.

Im folgenden Abschnitt wird zunächst der Aufbau der Leitungselemente-Bibliothek beschrieben, während in Abschnitt 5.2 der Ablauf der bibliotheksbasierten Detailverdrahtung dargestellt ist. Für eine genaue Vorhersage der Verzögerungszeiten der Bibliothekselemente ist eine spezielle Leitungsanordnung erforderlich. Diese wird zusammen mit dem Verfahren der Charakterisierung der Bibliothekselemente in Abschnitt 5.3 erläutert. Im letzten Abschnitt dieses Kapitels befindet sich eine Darstellung der Vor- und Nachteile der bibliotheksbasierten Verdrahtung.

5.1 Aufbau der Leitungselemente-Bibliothek

Die Leitungselemente-Bibliothek soll alle Grundstrukturen und die dazugehörigen Verzögerungszeiten beinhalten, die zur Verdrahtung eingesetzt werden können. Dazu ist eine Menge von Leitungsformen (Shapes) festzulegen, die dem Verdrahtungsalgorithmus zur Verfügung stehen. Während sich die Verdrahtbarkeit einer Schaltung verbessert, wenn die Anzahl der unterschiedlichen Verdrahtungsformen zunimmt, steigt gleichzeitig der Rechenaufwand für den Verdrahtungsalgorithmus. Unter Verdrahtbarkeit versteht man im Allgemeinen den Anteil der Netze, die durch den Verdrahtungsalgorithmus erzeugt werden können, ohne dass Entwurfsregeln verletzt werden. Eine Verdrahtbarkeit von 100% entspricht dementsprechend der vollständigen Verdrahtung einer platzierten Schaltung, ohne dass Kurzschlüsse durch Überlappungen auftreten. In dem hier vorgestellten Verfahren werden stets für alle Netze Verbindungen erzeugt. Dabei können Kurzschlüsse auftreten. Deshalb beschreibt die Verdrahtbarkeit für diesen Algorithmus den Anteil der Netze, die ohne einen Kurzschluss erzeugt werden können.

Um die Anzahl der Verdrahtungsformen in der Leitungsbibliothek auf einem sinnvollen Niveau zu halten, erfolgt die Speicherung aller Leitungs-Shapes in der so genannten Einheitslänge. Die Einheitslänge ist dabei auf die Gesamtlänge eines Verdrahtungs-Shapes bezogen. Die Anpassung der Ausdehnung der Leitungs-Shapes erfolgt später gleichzeitig mit der Platzierung der Shapes auf der Layoutfläche.

Zur Begrenzung der Anzahl von Verdrahtungs-Shapes beinhaltet die Bibliothek in dieser Implementierung nur drei unterschiedliche Typen von Leitungselementen: Gerade Leitungselemente (G-Shapes), Leitungselemente mit einem Knick (L-Shapes) und Leitungselemente mit zwei Knicken in unterschiedliche Richtungen (Z-Shapes). Die verschiedenen Typen der Bibliothekselemente sind - nicht in Einheitslänge - in Abbildung 5.1 dargestellt.

Die Auswahl, der in der Bibliothek zur Verfügung stehenden Verdrahtungs-Shapes, kann auch die möglichen Leitungslängen der Detailverdrahtung beeinflussen. G-, L- und Z-Shapes verbinden zwei Punkte in Manhattan-Geometrie auf dem kürzestmöglichen Weg.

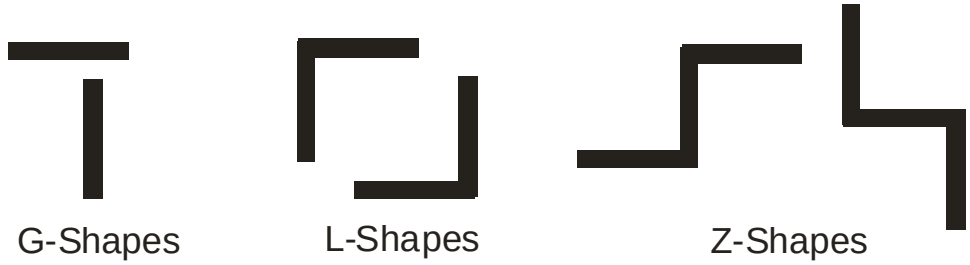

Abbildung 5.1: Typen der Verdrahtungs-Pattern

Fügt man der Bibliothek U-förmige Elemente mit zwei Knicken in gleicher Richtung hinzu, wird eine Verdrahtung um ein Hindernis ermöglicht. Dadurch steigt die Verdrahtbarkeit an, während gleichzeitig längere Umwege auftreten können. Abbildung 5.2 zeigt U-förmige Verdrahtungselemente.

Abbildung 5.2: U-förmige Verdrahtungselemente

Im Rahmen dieser Arbeit kommen nur Leitungsbibliotheken ohne U-förmige Shapes zum Einsatz, um Umwege zu verhindern. Durch das Einfügen zusätzlicher Globalverdrahtungspunkte in das Layout kann auch mit der Beschränkung auf G-, L- und Z-Shapes eine gute Verdrahtung erzeugt werden.

Neben der Form sind für jedes Element auch die Anzahl und die Positionen der Vias festgelegt. Dabei gibt es Bibliothekselemente mit null bis vier Vias. Die Beschränkung auf vier Vias stellt keine Einschränkung dar, da die Verdrahtungselemente zwischen den Globalverdrahtungspunkten relativ kurz sind. Deshalb entspricht eine Anzahl von vier Vias für die durchschnittliche Länge der Leitungs-Shapes einer eher hohen Via-Anzahl.

Um auch an dieser Stelle die Anzahl der unterschiedlichen Bibliothekselemente auf eine handhabbare Anzahl zu beschränken, sind die Vias jeweils an den Symmetriepunkten der Verdrahtungs-Shapes bzw. an den Symmetriepunkten der geraden Teile eines Verdrahtungs-Shapes positioniert. Die Positionen der Vias für Shapes mit einem und zwei Vias sind exemplarisch in Abbildung 5.3 dargestellt.

Die Vias legen ebenfalls fest, über wie viele Verdrahtungslagen der Ebenenwechsel stattfindet. Dabei wird in dieser Arbeit von einer Technologie ausgegangen, die Stacked-Vias erlaubt.

Abbildung 5.3: Via-Positionen der Verdrahtungselemente

Unter Stacked-Vias wird die Platzierung von mehreren Vias unmittelbar übereinander verstanden, was einem mehrfachen Ebenenwechsel z.B. von der ersten zur dritten Metalllage entspricht. Zu jedem Shape mit einem Via wird in der Leitungselemente-Bibliothek die Anzahl der Ebenen, über die der Wechsel erfolgt, gespeichert.

Als weitere Restriktion erfolgt im Rahmen dieser Arbeit die Verdrahtung auf einer Verdrahtungsebene jeweils nur in einer Richtung, d.h. entweder nur horizontal oder nur vertikal. Dabei kann der Benutzer die Richtung der untersten Metalllage festlegen. Im Regelfall verläuft diese parallel zur Power-Verdrahtung der Standard-Zellen in der gleichen Metalllage wie die Power-Verdrahtung. Anschließend werden die darüber liegenden Ebenen jeweils orthogonal zur vorangegangenen Ebene verdrahtet. Diese Art der Verdrahtung wurde ausgewählt, weil sich durch Experimente gezeigt hat, dass für die bibliotheksbasierte Verdrahtung die Verdrahtbarkeit bei orthogonaler Verdrahtung höher als bei der allgemeinen Verdrahtung ist. Zusätzlich führt die orthogonale Verdrahtung zu einer einfacheren Datenstruktur zur Speicherung der Daten (vgl. Abschnitt 5.2.2.4) und vereinfacht die in Abschnitt 5.2.2.5 beschriebene Berechnung der Überlappungen erheblich.

Jedem Verdrahtungselement der Bibliothek wird ein Verzögerungszeit-Wert zugewiesen. Diese Verzögerungszeit kann mit der Länge des Leitungssegments skaliert werden. Während die Bestimmung der Verzögerungszeit für die Leitungssegmente in Abschnitt 5.3 dargestellt ist, wird im folgenden Abschnitt zunächst der Ablauf der bibliotheksbasierten Verdrahtung detailliert erläutert.

5.2 Ablauf der Detailverdrahtung

Die Idee der bibliotheksbasierten Verdrahtung liegt in der Einschränkung der Freiheitsgerade des Verdrahtungsalgorithmus, um auf diese Weise eine bessere

Vorhersagbarkeit der Verdrahtungslänge während der Platzierungs- und Globalverdrahtungsphase treffen zu können.

Im allgemeinen Fall der Verdrahtung wird eine bibliotheksbasierte Verdrahtung mit der in Abschnitt 5.1 beschriebenen Anzahl von Verdrahtungsformen nicht möglich sein, da komplexere Verdrahtungsformen erforderlich wären. Aufgrund der während der Platzierung positionierten Globalverdrahtungspunkte (vgl. Kapitel 4) sind aber die Distanzen zwischen jeweils zwei Globalverdrahtungspunkten bzw. einer Zelle und einem Globalverdrahtungspunkt relativ klein. Dadurch kann in den meisten Fällen eine Verbindung mit einem der einfachen Shapes aus der in Abschnitt 5.1 beschriebenen Bibliothek realisiert werden.

Die Schritte der Detailverdrahtung sowie die im Rahmen dieses Verfahrens verwendete Datenstruktur sind in den folgenden vier Abschnitten beschrieben.

5.2.1 Initialverdrahtung

Die bibliotheksbasierte Detailverdrahtung basiert, wie schon in Abschnitt 5.1 beschrieben, auf der Optimierung einer Initialverdrahtung. Diese Initialverdrahtung kann dabei nicht mit einem konventionellen Verdrahter erzeugt werden, da dieser beliebige Verdrahtungsformen verwenden würde. Um einen hohen Rechenaufwand für die Überführung solcher Verdrahtungsformen auf die Elemente der Leitungselemente-Bibliothek zu verhindern, wird eine zufällige Initialverdrahtung unter Verwendung aller in der Bibliothek befindlicher Leitungs-Shapes erzeugt.

Die Zufallskomponente kommt dabei bei allen drei wesentlichen Entscheidungen der Initialverdrahtung zum Einsatz:

Im ersten Schritt der Initialverdrahtung wird das zu verdrahtende Netz zufällig ausgewählt.

Im zweiten Schritt erfolgt die Auswahl des Verdrahtungs-Shapes. Zwei horizontal oder vertikal auf einer Linie liegende Pins werden immer mit einem geraden Shape (G-Shape) verbunden. Für alle anderen Anordnungen von Pins wird zufällig ein L-oder ein Z-Shape ausgewählt.

Der dritte zufällige Entscheidungsschritt legt die Anzahl der Vias fest. Dabei wird die Anzahl der Vias für ein Netz zufällig aus dem Intervall [0..4] (vgl. Abschnitt 5.1) gewählt.

Die Initialverdrahtung wird aufgrund der zufällig selektierten Verdrahtungs-Shapes im Regelfall nicht legal sein, d.h. es gibt eine größere Anzahl von Kürzschlüssen im Layout. Während der folgenden Optimierungsschritte müssen alle Überlappungen zwischen mehreren Leitungen, die elektrisch gesehen Kurzschlüssen entsprechen, entfernt werden, um eine legale Verdrahtung zu erreichen. Eine Optimierung des Laufzeitverhaltens der Schaltung ist in dieser Phase des Entwurfs nur noch eingeschränkt möglich, da die Verdrahtungspunkte der Globalverdrahtung die Detailverdrahtung stark einschränken und mit L- oder Z-Shapes keine Umwege verdrahtet werden können. Damit beeinflusst lediglich die Anzahl der Vias, die bei der Verdrahtungsoptimierung festgelegt wird, die Leitungsverzögerungszeit.

5.2.2 Optimierung der Verdrahtung

Für die Optimierung der Verdrahtung wurden zwei unterschiedliche Verfahren implementiert. Zum einen kann ein Verfahren zur monotonen Suche verwendet werden, welches nur solche Modifikationsschritte akzeptiert, die zu einer Verbesserung der Verdrahtung im Sinne der Kostenfunktion führen. Das zweite Verfahren basiert auf einem Simulated-Annealing-Ansatz. Dieses - bei der Platzierung (vgl. Abschnitt 3.1.1.3) weit verbreitete Verfahren - akzeptiert unter bestimmten Voraussetzungen auch Verdrahtungsänderungen, die den Wert der Kostenfunktion verschlechtern. Für beide Optimierungsverfahren sind vergleichbar zur Platzierung folgende Schritte erforderlich:

Im ersten Schritt erfolgt die Selektion eines oder mehrerer Shapes zur Modifikation. Im zweiten Schritt wird ein neues Element aus der Verdrahtungselemente-Bibliothek ausgewählt, das das bisher verwendete Shape ersetzen soll. Diese beiden Schritte entsprechen bei der Platzierung der Selektion der zu vertauschenden Zellen, wenn man zur Platzierungsmodifikation Zellen jeweils paarweise vertauscht (vgl. Abschnitt 3.1.1.3). Im dritten Schritt muss das neu aus der Bibliothek ausgewählte Element längenmäßig an die Pins, die es verbinden soll, angepasst werden, da alle Bibliothekselemente Einheitslänge haben. Für diesen Schritt gibt es keine Entsprechung bei der Platzierung. Abschließend erfolgen die Bestimmung der Kosten der Verdrahtungsänderung und die daraus resultierende Akzeptanzentscheidung der Verdrahtungsmodifikation. Dabei hängt die Akzeptanzentscheidung vom gewählten Optimierungsverfahren - monotone Suche oder Simulated Annealing - ab. Die einzelnen Schritte der Verdrahtungsoptimierung werden im Folgenden genauer beschrieben.

5.2.2.1 Selektion eines Shapes zur Modifikation

Die Selektion des zu modifizierenden Verdrahtungselements erfolgt zufallsgesteuert. Dazu wird eine Zufallszahl aus dem Intervall [1..Netzanzahl] erzeugt und das entsprechende Netz zur Modifikation ausgewählt.

Bei der im Rahmen dieser Arbeit durchgeführten Implementierung wird auf die gleichzeitige Selektion mehrerer Pattern zur Modifikation verzichtet, da sich gezeigt hat, dass dieses die Ergebnisqualität nicht signifikant beeinflusst, aber den Rechenaufwand deutlich erhöht.

5.2.2.2 Selektion eines neuen Verdrahtungs-Shapes aus der Bibliothek

Die Selektion des neuen Verdrahtungs-Shapes aus der Bibliothek erfolgt ebenfalls zufällig in zwei Stufen. In dem ersten Schritt erfolgt die Auswahl der Verdrahtungsform - G-, L- oder Z-Form - und im zweiten Schritt die Auswahl der Anzahl der Vias. Diese zweistufige Auswahl wurde gewählt, da für gerade Shapes nur die Anzahl der Vias verändert werden kann, nicht aber die Verdrahtungsform. Für diese Elemente entfällt also die erste Selektionsstufe.

Bezüglich der L- und Z-förmigen Elemente können durch den ersten Modifikationsschritt prinzipiell folgende Modifikationen auftreten: Wird ein L-förmiges Element aus dem Layout entfernt und ein neues L-Shape mit anderer Ausrichtung dem Layout hinzugefügt, entspricht das einer Drehung um 180 Grad. Bei Z-förmigen Shapes entspricht die Drehung der Änderung der Vorzugsrichtung. Dabei wird durch die Vorzugsrichtung bei Z-förmigen Leitungssegmenten die Verdrahtungsrichtung bezeichnet, in der zwei der drei Segmentteile verlaufen. Zusätzlich kann ein L-Shape durch ein Z-Shape, bzw. ein Z-Shape durch ein L-Shape ersetzt werden. Der zweite Modifikationsschritt kann die Anzahl der Vias für alle Shapes verändern. Beispielhaft sind die Änderungen der Verdrahtung durch die Selektion eines neuen Verdrahtungs-Shapes mit der gleichen Form aber einer anderen Ausrichtung in Abbildung 5.4 dargestellt.

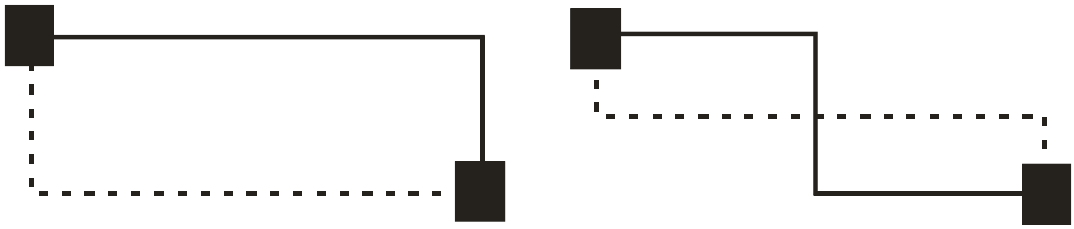

Abbildung 5.4: Modifikation durch Selektion eines Leitungs-Shapes gleicher Form

5.2.2.3 Sizing des selektierten Verdrahtungs-Shapes

Nach der Selektion eines neuen Shapes aus der Leitungselemente-Bibliothek muss dieses von der Länge her an die zu verbindenden Zellen angepasst werden, da die Leitungselemente-Bibliothek alle Elemente in Einheitslänge enthält.

Dieses Sizing wird unter Erhaltung der Symmetrieeigenschaften der Verdrahtungs-Shapes durchgeführt. Ein Verdrahtungs-Shape mit Einheitslänge, das beispielsweise ein Via in seiner Mitte hat, wird nach dem Sizing auf die Länge l das Via an der Position $l/2$ haben. Auf die gleiche Weise werden Leitungsknicke behandelt. Ein punktsymmetrisches Z-förmiges Shape mit Einheitslänge wird nach dem Sizing wieder punktsymmetrisch sein. Das Sizing ist beispielhaft für ein Z-Shape mit drei Vias in Abbildung 5.5 dargestellt.

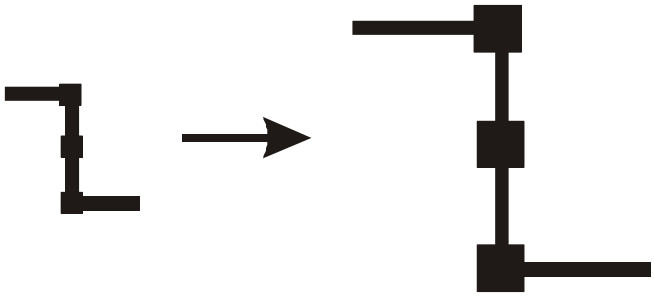

Abbildung 5.5: Shape-Sizing

5.2.2.4 Datenstruktur und Kostenfunktion der Optimierung

Durch die zufällige Initialplatzierung sowie die zufällige Modifikation oder Selektion von Leitungssegmenten im Verlauf der Detailverdrahtung werden häufig Überlappungen zwischen zwei oder mehreren Leitungen auftreten. Diese Kurzschlüsse müssen im Laufe der Detailverdrahtungsoptimierung beseitigt werden und werden dazu in der Kostenfunktion des Optimierungsverfahrens berücksichtigt. Zur Erkennung dieser Kurzschlüsse ist eine Datenstruktur erforderlich, die ein effizientes Auffinden von Überlappungen von Verdrahtungs-Shapes ermöglicht, da nach jedem Iterationsschritt der Detailverdrahtung eine Berechnung der Änderung der Verdrahtungskosten durchgeführt werden muss.

Eine rasterbasierte Datenstruktur, bei der jede Leitung auf eine Vielzahl von Rasterpunkten abgebildet wird, erlaubt eine effiziente Erkennung von Kurzschlüssen. Dazu werden in einer matrixähnlichen Darstellung jedem Rasterelement die Nummern der auf dem Rasterelement liegenden Netze zugewiesen. Sobald auf einem Rasterelement mehr als ein Netz positioniert ist, liegt eine Überlappung und damit ein

Kurzschluss vor. Allerdings ist der Speicherbedarf für eine rasterbasierte Datenstruktur sehr hoch. Geht man von einer Chipfläche von 2,5x2,5 cm^2 und einer Leitbahnbreite von 0,25 µm sowie einem Mindestabstand zwischen zwei Leitungen von ebenfalls 0,25 µm sowie vier Metalllagen aus, ergibt sich folgender Speicherbedarf:

$$\begin{aligned} M &= Anzahl\ \ Layer \cdot Anzahl\ \ \ Rasterelemente \cdot Speicherbedarf\ \ \ je\ \ \ Rasterelement \\ &= Layer \cdot \left(\frac{b_{Chip}}{b_{Leitbahn} + b_{Abstand}} \right)^2 \cdot mem(Raster) = 4 \cdot \left(\frac{0,025\ m}{0,5 \cdot 10^{-6}\ m} \right)^2 \cdot 32\ bit\ \approx\ 38\ GB \end{aligned} \quad \textbf{(5.1)}$$

Dieser Speicherbedarf ist so hoch, dass eine rasterbasierte Speicherung der Daten nicht sinnvoll ist.

Daher erfolgt die Speicherung der Daten in diesem Ansatz linienbasiert. Dazu werden nur der Anfangs- und der Endpunkt jedes Leitungssegmentteils gespeichert. Ein Leitungssegmentteil beschreibt dabei ein gerades Stück eines Verdrahtungs-Shapes, das auf einer Metallebene liegt. Damit beginnen und enden Leitungssegmentteile an den Start- und Endpunkten der Verdrahtungs-Shapes, an Vias und an Knicken. Experimente haben gezeigt, dass ein Leitungssegment durchschnittlich aus ca. vier geraden Teilen besteht. Da jeder Leitungssegmentteil mit zwei Punkten in der Datenstruktur gespeichert wird, ergibt sich ein Speicherbedarf von durchschnittlich acht Integerwerten pro Leitungssegment. Dies entspricht einem Speicherbedarf von 24 Byte je Leitungssegment, wenn man voraussetzt, dass ein Integerwert durch vier Bytes codiert wird. Auch bei einer hohen Anzahl von Netzen, ist der Speicherbedarf dieser linienbasierten Datenstruktur deutlich geringer als der Speicherbedarf der rasterbasierten Struktur.

Die Speicherung der Start- und Endpunkte der Leitungssegmentteile erfolgt in einer Sparse-Matrix. Diese bietet eine rasterartige Struktur, ohne dass die unbenutzten Elemente der Matrix Speicherplatz belegen. Zusätzlich erlaubt eine Sparse-Matrix ein effizientes Einfügen und Löschen der Verdrahtungspunkte.

Da in einer Sparse-Matrix jeweils nur ein Wert an einer Position gespeichert werden kann, erfordert die Speicherung von übereinander liegenden Start- und Endpunkten mehrerer Verdrahtungssegmente eine Sonderbehandlung. Zur Berechnung der Überlappungskosten muss ein effizienter Zugriff auf die Netze über die Koordinaten auf dem Layout möglich sein. Deshalb werden die übereinander liegenden Netzpunkte in einem Hash gespeichert. Ein Hash erlaubt ein effizientes Einfügen, Suchen und Löschen von Elementen mit einer Laufzeitkomplexität von $O(n \cdot \log n)$. Dabei besteht ein Hash jeweils aus Wertepaaren: Einem Schlüssel, nach dem schnell gesucht werden kann, und dem zum Schlüssel zugehörigen zu speichernden Wert. Bei der Detailverdrahtung wird der Hash-Schlüssel aus der Position auf der Layoutfläche berechnet. Dieses stellt sicher,

dass für jede Position auf der Layoutfläche ein eindeutiger Schlüssel berechnet wird. Bei dem zu speichernden Wert handelt es sich um die Liste aller Netz-Nummern, die an der betrachteten Position auf der Layoutfläche beginnen oder enden. Die Berechnung des Hash-Schlüssels für ein Element an der Position (x, y) auf Layer l kann wie folgt erfolgen:

$$Key = (l-1)\cdot(W+H)+W\cdot y+x \qquad \textbf{(5.2)}$$

W und H beschreiben in Gleichung (5.2) die Breite bzw. die Höhe der Layoutfläche, l die Nummer der Verdrahtungsebene.

Um bei der Überlapp-Berechnung nicht an jedem Rasterelement im Hash nach einer eventuell vorhandenen Liste von Netzen zu suchen, wird der Hash-Schlüssel in der Verdrahtungsstruktur wie ein Netz gespeichert. Damit entspricht eine null der Verdrahtungsmatrix einem Verdrahtungsrasterelement ohne Verdrahtungsstart- oder Verdrahtungsendpunkt. Zur Unterscheidung zwischen Netzen und Hash-Schlüsseln in der Verdrahtungsdatenstruktur werden alle prinzipiell positiven Hash-Schlüssel mit einem negativen Vorzeichen in der Datenstruktur gespeichert. Dadurch kann bei der Überlapp-Berechnung bereits aus dem Vorzeichen des gefundenen Elements in der Matrix der Verdrahtungspunkte erkannt werden, ob es sich um den Start- oder Endpunkt eines geraden Verdrahtungssegments handelt oder um einen Hash-Schlüssel, bei dem die Anzahl der Netze und ggf. die Netz-Nummern aus dem Hash gesucht werden müssen. Der prinzipielle Aufbau der Datenstruktur ist in Abbildung 5.6 dargestellt.

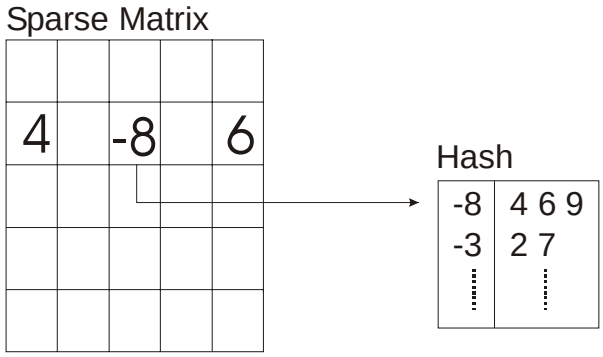

Abbildung 5.6: Datenstruktur zur Speicherung der Detailverdrahtung

Die Erkennung von Überlappungen ist aufgrund der gewählten orthogonalen Verdrahtung einfach möglich. Betrachtet man den Start- und den Endpunkt eines Verdrahtungssegmentteils innerhalb einer Zeile bzw. einer Spalte der Verdrahtungsmatrix, dürfen sich zwischen den beiden Matrixeinträgen des Segments keine weiteren Start- oder Endpunkte von anderen Verdrahtungselementen befinden, wenn keine Überlappung auftreten soll. Werden andere Einträge zwischen den Punkten des Netzes gefunden, wurde ein Kurzschluss erkannt.

5.2.2.5 Bewertung eines Detailverdrahtungsschrittes

Für das Optimierungsverfahren ist zur Bewertung jedes Detailverdrahtungsschritts eine Kostenfunktion erforderlich. Die Kostenfunktion berücksichtigt zwei unterschiedliche Kostenkomponenten.

Überlappungskosten:

Die Überlappungskosten sind ein Maß für die Anzahl der Kurzschlüsse im Layout. Dabei wird nicht nur die Zahl der Überlappungen (Kurzschlüsse) zwischen zwei oder mehr Leitungen gezählt, sondern es wird die Länge der Überlappungen bestimmt. Dadurch kann beispielsweise eine Shape-Änderung von einem L-Shape zu einem Z-Shape zu einer Verbesserung der Kostenfunktion - wie in Abbildung 5.7 dargestellt - führen, auch wenn es weiterhin einen Kurzschluss zwischen den beteiligten Netzen gibt.

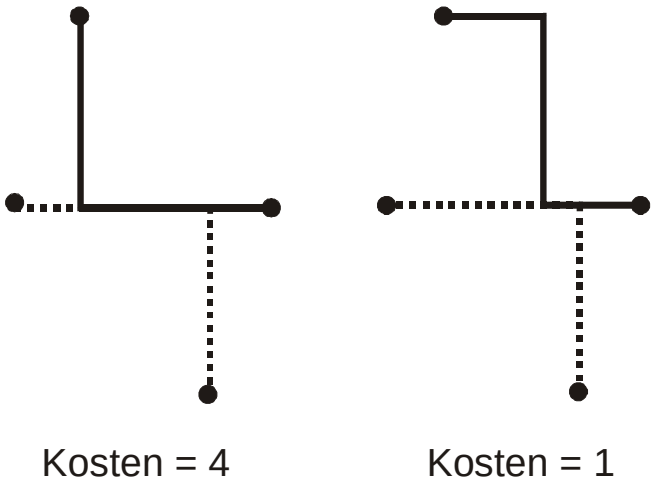

Abbildung 5.7: Überlappungskostenänderung durch Shape-Modifikation

Timing-Violation-Kosten:

Das Ziel des im Rahmen dieser Arbeit entwickelten Designflows ist die Einhaltung aller Pfad-Constraints. Der zweite Term der Kostenfunktion ist daher ein Maß für die Einhaltung der Pfad-Constraints. Dabei wird - vergleichbar zu den Überlappungskosten - nicht nur die Anzahl der Pfade mit Constraint-Verletzungen gezählt, sondern es werden die Beträge aller Laufzeiten, die das jeweilige Pfad-Constraint überschreiten, aufsummiert.

Die Gesamtkosten für einen Detailverdrahtungsschritt setzen sich aus den Überlappungskosten und den Timing-Violation-Kosten gleichmäßig zusammen. Eine unterschiedliche Gewichtung der beiden Anteile ist nicht sinnvoll, da erst eine

verwertbare Detailverdrahtung erreicht wird, wenn sowohl die Überlappungs- als auch die Timing-Violation-Kosten null sind.

Zur Berechnung der Kosten wurden zwei unterschiedliche Kostenfunktionen implementiert. Die erste Kostenfunktion berechnet die Kosten der gesamten Detailverdrahtung und wird einmalig nach der Initialplatzierung eingesetzt. Die zweite Kostenfunktion berechnet die Überlappungskosten, die durch ein einzelnes Netz verursacht werden. Diese inkrementelle Kostenfunktion wird jeweils vor und nach einer Shape-Modifikation aufgerufen. Die Differenz der Kosten entspricht der mit der Shape-Änderung verbundenen Kostenänderung.

Die inkrementelle Kostenfunktion führt im Vergleich zur Berechnung aller Überlappungen auf der Layoutfläche in jedem Iterationsschritt zu einer deutlichen Reduktion der Laufzeit. Da die Kostenfunktion zwischen 10^6 und 10^{10} mal aufgerufen wird, wäre eine Berechnung der Kosten mit der Kostenfunktion für das gesamte Layout wenig effizient, da nur geringfügige Änderungen in jedem Schritt an der Verdrahtung vorgenommen werden.

Die inkrementelle Kostenfunktion baut auf der in Abschnitt 5.2.2.4 beschriebenen Datenstruktur auf. Zu jedem Verdrahtungs-Shape werden die Start- und Endkoordinaten sowie der Verdrahtungslayer von allen dazugehörigen Verdrahtungssegmentteilen gespeichert. Zur inkrementellen Kostenberechnung müssen die Kosten für ein Verdrahtungs-Shape berechnet werden. Dazu muss jeweils die Liste der Koordinaten der Verdrahtungseckpunkte abgearbeitet werden. Für jedes Punktepaar wird die Ausrichtung (horizontal bzw. vertikal) bestimmt. Anschließend werden die Überlappungen zwischen den Eckpunkten berechnet.

5.2.2.6 Akzeptanz von Verdrahtungsänderungen

Der letzte Schritt einer Iteration der Detailverdrahtung ist die Entscheidung, ob die Veränderung der Verdrahtung akzeptiert wird. Die Akzeptanz der Verdrahtungsänderungen hängt von dem verwendeten Optimierungsverfahren ab.

Das hier zum Einsatz kommende und für die Verdrahtung modifizierte Simulated Annealing verwendet eine exponentielle Abkühlkurve und eine, zu der in Abschnitt 3.1.1.3 beschriebenen, vergleichbare Kostenfunktion. Der Wert der Akzeptanzfunktion

$$f_{Akzept} = \min(1.0, e^{-\frac{Verdrahtungskostenänderung}{Temperatur}}) \qquad \textbf{(5.3)}$$

wird mit einer Zufallszahl aus dem Intervall [0..1] verglichen. Ein Verdrahtungsschritt, der die Kosten verschlechtert, wird akzeptiert, wenn die Zufallszahl kleiner als der Wert der Akzeptanzfunktion ist.

5.3 Leitungsanordnung und Charakterisierung

Die Abschätzung der Leitungsverzögerungen zur Berechnung der Pfadkräfte während der Platzierung erfolgt wie in Abschnitt 4.2 beschrieben mit mehreren Bounding-Boxen. Diese Modellierung setzt voraus, dass die Verzögerungszeit einer Leitung proportional zum Quadrat ihrer Länge ist. Dieses ist im Allgemeinen nur in erster Nährung richtig, da die Verzögerungszeit auch von den Kopplungskapazitäten zu den benachbarten Leitungen und damit auch von den Signalwechseln auf diesen Leitungen abhängt.

Um eine längenabhängige Abschätzung der Verzögerungszeit zu ermöglichen, wird in diesem Verdrahtungsansatz eine spezielle Leitungsanordnung angenommen. In der in Abbildung 5.8 dargestellten Anordnung liegt eine Signalleitung stets zwischen einer Masse- und einer Versorgungsleitung. Die Verdrahtung auf übereinander liegenden Metallebenen erfolgt jeweils orthogonal.

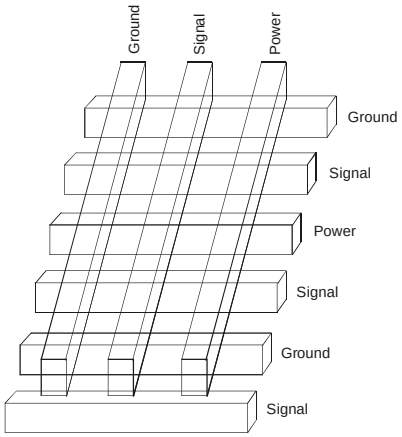

Abbildung 5.8: Leitungsschema

Dieses Verdrahtungsschema von [Khatri99] erlaubt eine längenproportionale Abschätzung der Leitungsverzögerung ohne einen signifikanten Fehler. Die Kopplungskapazitäten zu den Nachbarleitungen sind vor der Verdrahtung bekannt und die Signale der Nachbarleitungen sind statisch. Die Kopplungskapazitäten zwischen zwei Signalleitungen können vernachlässigt werden, da diese um den Faktor 10-15 kleiner sind, als die Kopplungskapazitäten zu den direkten Nachbarleitungen.

Die Charakterisierung der Leitungssegmente erfolgte mit einem 3D-Fieldsolver, der nach der Boundary-Element-Methode [AutoBEM] arbeitet. Dazu wurden

unterschiedliche Leitungsanordnungen mit einem Layouteditor entworfen und die Kapazitäten und Widerstände der Anordnungen mit dem Fieldsolver bestimmt. Zur Untersuchung des Fehlers bei der Skalierung der Leitungs-Shapes wurden unterschiedlich lange Anordnungen gleicher Form untersucht. Dabei hat sich gezeigt, dass für Ecken und Vias näherungsweise eine konstante Verzögerungszeit angenommen werden kann und die Verzögerungszeit der geraden Teile eines Verdrahtungs-Shapes für die verwendete Leiteranordnung proportional zur Leitungslänge ist. Deshalb können die Verdrahtungs-Shapes und ihre dazugehörigen Verzögerungszeiten ohne nennenswerten Fehler skaliert werden.

5.4 Vor- und Nachteile der bibliotheksbasierten Detailverdrahtung

Im Gegensatz zu allen Verdrahtungsalgorithmen, die auf einer Wegesuche basieren, können Umwege bei der hier zum Einsatz kommenden bibliotheksbasierten Detailverdrahtung nur begrenzt auftreten, da beispielsweise keine U-förmigen Verdrahtungselemente zur Verfügung stehen. Da im Allgemeinen keine Verdrahtung mit den hier verwendeten einfachen Verdrahtungsformen möglich ist, erfordert die bibliotheksbasierte Detailverdrahtung einen Vorbereitungsschritt, der lange Netze in kürzere Abschnitte zerlegt. Dieser erfolgt für diese Arbeit während der Platzierung und Globalverdrahtung durch die Platzierung der Leitungspunkte.

Während einige Algorithmen zur Detailverdrahtung auf Heuristiken basieren, kommt bei der bibliotheksbasierten Detailverdrahtung ein Optimierungsverfahren zum Einsatz. Bei dem Einsatz von Heuristiken können in vielen Fällen gute Ergebnisse erzeugt werden, aber es gibt keine Garantie für die Ergebnisqualität. Während einige Optimierungsverfahren lokale Minima nicht mehr verlassen, erlaubt der hier verwendete Simulated-Annealing-Ansatz auch das Verlassen von lokalen Minima. Dieses setzt allerdings gut gewählte Parameter voraus, die beim Simulated Annealing in der Regel experimentell zu bestimmen sind.

6 Implementierung und Ergebnisse

Das vorgestellte Verfahren wurde im Rahmen dieser Arbeit prototypisch in Matlab implementiert. Matlab ist ein Mathematiksystem, welches eine Programmierung mittels einer C-ähnlichen Programmiersprache ermöglicht. Der Programmcode wird jedoch nicht kompiliert, sondern zur Ausführungszeit interpretiert. Dieses führt im Vergleich zu anderen, in Maschinencode übersetzten Programmen, zu einer deutlich höheren Programmlaufzeit. Außerdem existieren in Matlab keine dynamischen Datenstrukturen. Dadurch können komplexe Datenstrukturen nur als Felder realisiert werden, was zu höheren Laufzeiten für Einfüge- und Löschoperationen im Vergleich zu dynamischen Datenstrukturen wie z.B. verketteten Listen führt.

Zur Untersuchung der Eigenschaften des neuen leitbahnorientierten Designflows wurden verschiedene Testdesigns untersucht. Dabei wurden die Ergebnisse der Platzierungs- und Globalverdrahtungsphase unabhängig von der Detailverdrahtung untersucht. Die Verdrahtungsergebnisse bauen dagegen immer auf der vorangegangen Platzierungs- und Globalverdrahtungsphase auf, da die speziellen Globalverdrahtungs-Informationen (vgl. Abschnitt 4.1) für die bibliotheksbasierte Detailverdrahtung erforderlich sind. Zum Vergleich der Ergebnisse wurden die gleichen Testdesigns mit dem Platzierer „Plato“, [Eisenmann99] platziert, der als anerkannt guter Platzierungsalgorithmus gilt. Die Ergebnisse der Platzierung und Globalverdrahtung werden in Abschnitt 6.3.1 im Vergleich zu den Ergebnissen von Plato vorgestellt. Die Detailverdrahtungsergebnisse werden in Abschnitt 6.3.2 vorgestellt. Die Veränderungen der Verzögerungszeiten während des leitbahnorientierten Designablaufs zwischen den geschätzten Verzögerungszeiten der Platzierung und den Verzögerungszeiten nach der Verdrahtung werden in Abschnitt 6.3.3 beschrieben. Eine Bewertung des neuen Designflows erfolgt in Abschnitt 6.4.

6.1 Implementierung

Das im Rahmen dieser Arbeit entwickelte Platzierungs- und Verdrahtungswerkzeug besteht - wie in Abbildung 6.1 dargestellt - aus sieben Komponenten:

Die Eingangsschnittstelle liest Netzlisten im Yal-Format [YAL] sowie Netzlisten in einem proprietären Pfadformat ein. Dieses Pfadformat wird von einem zusätzlichen, nicht im Rahmen dieser Arbeit entwickelten, C-Programm (VMAP) aus strukturellen Verilog-Netzlisten erzeugt. Für das Yal-Eingabeformat erfolgt gleichzeitig mit dem Einlesen die Bestimmung aller Pfade mit einer Tiefensuche innerhalb der durch einen Graphen repräsentierten Netzliste. Die Pfad-Constraints werden aus der - durch den Benutzer festzulegenden - maximalen Taktfrequenz berechnet. Zur Erzeugung der Pfade ist eine Textdatei erforderlich, die die Zellnamen aller Flipflops beinhaltet.

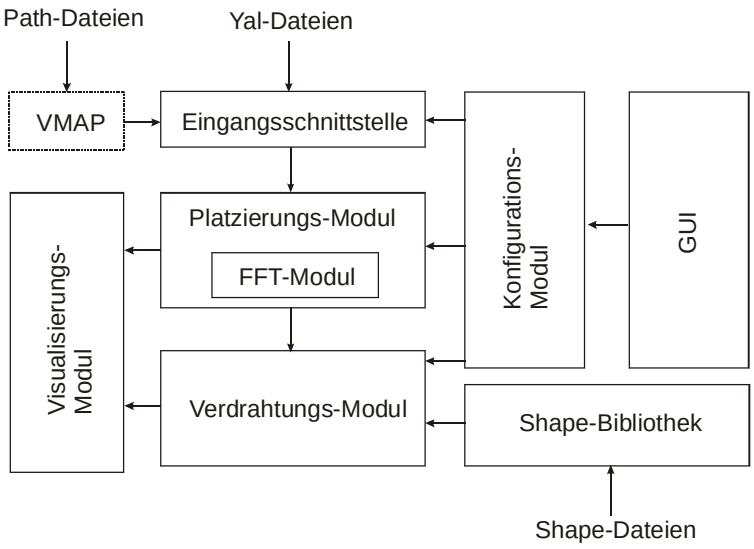

Abbildung 6.1: Programmstruktur

Das Platzierungsmodul berechnet die simultane Platzierung und Globalverdrahtung. Dabei erfolgt die in Abschnitt 4.4 beschriebene Berechnung der abstoßenden Kräfte mit Hilfe einer FFT, welche durch das im Platzierungs-Modul befindliche FFT-Modul berechnet wird. Das Platzierungs-Modul beinhaltet außerdem ein Ausgangsinterface, welches die Platzierung im Physical-Design-Exchange-Format (pdef) in eine Textdatei ausgibt. Die Ausgabe der virtuellen Zellen kann dabei aktiviert oder deaktiviert werden. Eine Deaktivierung ist für die Weiterverarbeitung der Platzierungsergebnisse mit einem kommerziellen Werkzeug notwendig.

Das Verdrahtungsmodul führt die bibliotheksbasierte Detailverdrahtung durch. Dazu wird eine Bibliothek verwendet, die Informationen über die zur Verfügung stehenden Verdrahtungs-Shapes sowie die Verzögerungszeiten der Shapes enthält. Diese Bibliothek wird aus einer Textdatei, die die Spezifikation aller Shapes enthält, aufgebaut.

Das Visualisierungsmodul sorgt für die graphische Darstellung der Platzierungs- und Verdrahtungsergebnisse auf dem Bildschirm. Da die Grafikausgabe unter Matlab relativ langsam ist, kann ausgewählt werden, wie viele Platzierungs- bzw. Verdrahtungsiterationsschritte visualisiert werden sollen.

Das Konfigurationsmodul beinhaltet alle Platzierungs- und Verdrahtungsparameter, die durch den Benutzer festgelegt werden können. Dazu zählen beispielsweise die Anzahl der Platzierungs-Iterationsschritte und die Start- und Endtemperatur des Simulated Annealings der Detailverdrahtung. Eine Übersicht aller Parameter befindet sich im Anhang dieser Arbeit.

Das Modul GUI realisiert eine grafische Benutzeroberfläche zur Eingabe der Benutzerparameter und zum Starten des Programms.

6.2 Testdesigns

Zur Untersuchung der Ergebnisse des leitbahnorientierten Designflows wurden sechs unterschiedliche Designs der MCNC-Benchmark-Suite [Madden02] verwendet. Bei der MCNC-Benchmark-Suite handelt es sich um verschiedene Netzlisten, die auf unterschiedlichen Standardzell-Bibliotheken aufbauen. Die Netzlisten der Benchmark-Schaltungen liegen jeweils im Yal-Format [YAL] vor.

Eine Übersicht über die Eigenschaften der Testbeispiele kann der folgenden Tabelle entnommen werden.

Design	Anzahl der Zellen	Anzahl der Netze	Anzahl der Ein- und Ausgänge
Primary1	833	904	81
Struct	1.952	1.920	64
Biomed	6.515	7.052	97
Industry1	3.085	2.594	814
Industry2	12.637	1.3419	495

Tabelle 6.1: Eigenschaften der Benchmark-Schaltungen

Für das Design Primary1 beträgt die Höhe der Standardzellen 150 µm, während die Minimalbreite der Leitungen auf allen Metallebenen 4 µm beträgt. Für die Beispiele Struct, Biomed und Industry2 haben die Standardzellen eine Höhe von 104 µm, die Leitbahnbreite beträgt 3 µm. Im Industry1-Beispiel beträgt die Zellhöhe 68 µm. Die Abmessungen der Standardzellen dieser Benchmark-Schaltungen sind deutlich größer als die Abmessungen von aktuellen Technologien mit Strukturgrößen von weniger als 0,25 µm. Diese Schaltungen werden in dieser Arbeit trotzdem verwendet, um eine Vergleichbarkeit der Ergebnisse mit anderen Ansätzen zu ermöglichen, für die Platzierungsergebnisse nur für die MCNC-Benchmarks veröffentlicht sind. Bei neueren Technologien, für die es noch keine allgemein verwendeten Benchmark-Schaltungen gibt, ist in jedem Fall eine Verbesserung der Ergebnisse im Vergleich zu konventionellen Verfahren zu erwarten. Durch den Anstieg der Verzögerungszeiten der Leitungen pro Längeneinheit bei gleichzeitig sinkenden Gatterverzögerungen im Vergleich von älteren zu neueren Technologien ist eine stärkere Berücksichtigung der Leitbahneigenschaften notwendig. Dieses erfolgt bei dem hier vorgestellten Designflow durch die Globalverdrahtung während der Platzierung und dem Einsatz einer Leitungselemente-Bibliothek. Diese ermöglicht eine genauere Abschätzung der Verzögerungszeiten während der Platzierung und es können auf diese Weise große Differenzen zwischen den geschätzten und den realen Verzögerungszeiten – insbesondere bei neuen Technologien mit einer relativ hohen Leitungsverzögerung pro Länge - vermieden werden.

6.3 Ergebnisse

In den folgenden drei Abschnitten werden die Ergebnisse der Experimente mit den sechs Testdesigns dargestellt. Dazu werden zunächst die Ergebnisse der simultanen Platzierung und Globalverdrahtung beschrieben und mit einem konventionellen Platzierungswerkzeug verglichen. Anschließend erfolgt die Beschreibung der Detailverdrahtungsergebnisse. Abschließend werden die Ergebnisse des gesamten Designflows bestehend aus Platzierung und Globalverdrahtung sowie der Detailverdrahtung in Abschnitt 6.3.3 beschrieben.

6.3.1 Platzierung- und Globalverdrahtung

Zur Bewertung der erzeugten Platzierungen, die die Globalverdrahtung beinhalten, wurden alle fünf Testdesigns platziert und die Platzierungsergebnisse mit den Ergebnissen der aktuellen Implementierung „Plato“ von [Eisenmann99] verglichen.

Während für die Testbeispiele Struct und Biomed genaue Informationen über die Zellen der diesen Designs zugrunde liegenden Standardzell-Bibliotheken vorliegen, ist dieses für die anderen Testdesigns der MCNC-Benchmarksuite nicht der Fall. Für die zum Beispiel Primary1 gehörende Bibliothek gibt es eine Beschreibung der Zellen, während keinerlei Informationen über die Zellen der Beispiele Industry1 und Industry2 zur Verfügung stehen. Für diese Schaltungen wurden den Zellen generisch erzeugte Verzögerungszeiten zugewiesen. Die Erkennung von Flipflops in den Beispielen Industry1 und Industry2 basiert auf einer Analyse der Netzliste.

In den meisten Fällen werden Platzierungsergebnisse anhand der geschätzten Gesamtverdrahtungslänge verglichen, während hier zwei andere Kriterien eingesetzt werden. Es wird die Anzahl der Pfade mit Überschreitung der Pfad-Constraints bestimmt und zusätzlich die Länge aller Pfade verglichen. Die Länge der Pfade wird dabei als Summe der an dem jeweiligen Pfad beteiligten Netzlängen bestimmt. Die Netzlängenabschätzung erfolgt durch mehrere Bounding-Boxen wie in Abschnitt 4.2 beschrieben.

Diese beiden Kriterien ermöglichen eine Bewertung der Platzierung aufgrund von Informationen über die Einhaltung von Pfad-Restriktionen sowie die Länge aller Pfade. Ein Design, das nach der Platzierung und Globalverdrahtung keine Pfad-Constraints verletzt und bei dem die Längen der kritischen Pfade deutlich unter der zulässigen Maximallänge liegen, wird sich mit hoher Wahrscheinlichkeit unter Einhaltung aller Pfad-Constraints verdrahten lassen. Designs, die schon in dieser Designphase eine größere Anzahl von Constraint-Verletzungen aufweisen, werden auch nach der Verdrahtung die Laufzeit-Randbedingungen nicht erfüllen können. Für diese Designs kann bei dieser Art der Bewertung nach der Platzierung – ohne, dass eine Durchführung der

Verdrahtung erforderlich ist – eine Resynthese oder eine erneute Platzierung durchgeführt werden. Das Bewertungskriterium Gesamtverdrahtungslänge erlaubt dagegen keine vergleichbaren Aussagen nach der Platzierung. Auch wenn die Gesamtverdrahtungslänge eines Designs im Vergleich zu anderen Platzierungen kurz ist, kann eine durchgeführte Platzierung unter Umständen nicht mehr derart verdrahtet werden, dass alle Constraints eingehalten werden können. Deshalb ist dieses Bewertungskriterium für laufzeitkritische Schaltungen zur Bewertung einer Platzierung nicht gut geeignet und wird im Rahmen dieser Arbeit nicht verwendet.

Die Berechnung der Pfad-Verzögerungen und die Überprüfung der Überschreitung der Pfad-Verzögerungszeiten werden - wie in Kapitel 4 beschrieben - durchgeführt. Zum Vergleich der Pfadlängen werden alle Pfade in drei Kategorien eingeordnet: kurze, mittlere und kritische Pfade. Dabei werden die 20% kürzesten Pfade in die Kategorie „kurz“ und die 20% längsten Pfade in die Kategorie „kritisch“ eingeordnet. Alle anderen Pfade zählen zur Kategorie „mittel“.

Die Ergebnisse der Platzierungen beider Werkzeuge werden anhand der Summen der Pfadlängen in jeder Kategorie miteinander verglichen. Die Anzahl der Pfade jedes Designs kann der Tabelle 6.2 entnommen werden, während die Platzierungsergebnisse für „Plato“ und für den im Rahmen dieser Arbeit entwickelten Platzierungsansatz in den Tabellen 6.3 bzw. 6.4 zu finden sind.

Design	**Pfadanzahl**
Struct	7.509
Primary1	9.410
Biomed	12.015
Industry1	773
Industry2	18.525

Tabelle 6.2: Pfadzahl der Testdesigns

Design	**Länge aller „kurzen“ Pfade**	**Länge aller „mittleren“ Pfade**	**Länge aller „kritischen“ Pfade**
Struct	5.723.182	30.751.995	14.982.920
Primary1	8.866.936	82.453.176	51.554.604
Biomed	1.826.328	27.828.214	39.146.203
Industry1	22.038	514.666	599.783
Industry2	3.102.987	108.200.340	102.598.654

Tabelle 6.3: Pfadlängenzuordnung der Platzierungsergebnisse von „Plato“

Design	Länge aller „kurzen“ Pfade	Länge aller „mittleren“ Pfade	Länge aller „kritischen“ Pfade
Struct	4.581.476	29.275.027	14.889.031
Primary1	11.492.536	89.190.902	47.764.344
Biomed	7.301.084	50.612.085	29.583.622
Industry1	26.263	405.818	498.789
Industry2	3.363.479	65.190.420	99.801.867

Tabelle 6.4: Pfadlängenzuordnung der Platzierungsergebnisse für den neuen leitbahnorientierten Designflow

Die Längenabschätzung in µm in den Tabellen 6.3 und 6.4 basiert auf der Zerlegung aller Mehrpunktnetze in Zweipunktnetze. Diese Abschätzung führt in der Praxis zu einer deutlichen Überschätzung der Gesamtverdrahtungslänge. Eine genauere Berechnung der Verdrahtungslänge wäre nur möglich, wenn die Verzweigungspunkte jedes Mehrpunktnetzes bekannt wären. Der Aufwand zur Berechnung dieser Punkte ist jedoch so hoch, dass diese während der Platzierung und Globalverdrahtung nicht möglich ist.

Die hohen Verdrahtungslängen ergeben sich durch die exponentielle Zunahme der Anzahl der Pfade mit der Zellanzahl. Dadurch werden viele Netze in den Längen vieler Pfade berücksichtigt. Die exponentielle Zunahme der Anzahl der Pfade mit der Schaltungsgröße tritt jedoch nur bei flachen Schaltungen auf. Bei hierarchischen Schaltungen sind die Ein- und Ausgänge jedes Moduls im Regelfall durch Flipflops gepuffert, so dass es keine modulübergreifenden Pfade gibt und der exponentielle Zusammenhang zwischen Zellanzahl und Pfadanzahl nur innerhalb der Module gilt.

Die Veränderung der Summen der Längen der Pfade in den jeweiligen Kategorien zwischen den beiden Platzierungswerkzeugen ist in Tabelle 6.5 dargestellt.

Design	Änderung der Länge der „kurzen“ Pfade	Änderung der Länge der „mittleren“ Pfade	Änderung der Länge der „kritischen“ Pfade
Struct	- 20,0 %	- 4,8 %	- 1,0 %
Primary1	+ 29,6 %	+ 8,2 %	- 7,4 %
Biomed	+ 300,0 %	+ 81,9 %	- 24,4 %
Industry1	+ 19,2 %	-21,1 %	-16,8 %
Industry2	+ 8,4 %	- 39,7 %	- 2,7 %

Tabelle 6.5: Veränderungen der Pfadlängen

Man erkennt, dass sich die Summe der Längen der kritischen Pfade in allen Designs verkürzt hat. Gleichzeitig ist die Summe der kurzen Pfade bei einigen Designs deutlich angestiegen. Die Veränderung der Längen der Pfadlängensumme der mittleren Pfade variiert designabhängig.

Das Designbeispiel Biomed weist eine besonders starke Verkürzung der kritischen Pfadlängen und gleichzeitig einen deutlichen Anstieg der kurzen und mittleren Pfade auf. Dieses ist durch eine große Anzahl von Netzen mit einem hohen Fan-Out in den kritischen Pfaden erklärbar. Der leitbahnorientierte Designflow zerlegt zur Pfadlängenberechnung alle Mehrpunktnetze in Zweipunktnetze. Dadurch erfolgt während der Platzierung durch den Einsatz der Pfadkraft, die die Differenz zwischen aktueller Pfadlänge und Pfad-Constraint modelliert, eine Optimierung der Zweipunktnetze, die ein Mehrpunktnetz bilden und nicht eine Optimierung der Mehrpunktnetzlänge. Deshalb können kritische Pfade, die durch ein Mehrpunktnetz mit einem hohen Fan-Out verlaufen, stärker verkürzt werden als kritische Pfade, die nur durch Zweipunktnetze verlaufen. Durch die Verkürzung einer Zweipunktverbindung eines Mehrpunktnetzes steigt jedoch in der Regel gleichzeitig die Länge anderer Zweipunktverbindungen des Mehrpunktnetzes an. Gleichzeitig führt eine deutliche Reduktion der Länge der kritischen Pfade zu einem Anstieg der Pfadlängen in den anderen Pfadkategorien.

Die Verbesserung der Platzierung des Beispiels Industry1 bezüglich der Summen der Pfadlängen in den Kategorien „mittel“ und „kritisch“ fällt im Vergleich zu Plato deutlich höher als bei den anderen Designs aus. Eine mögliche Erklärung könnten Fehler bei der Erkennung der Flipflops sein.

Das Ziel des neuen Designflows ist - wie in Kapitel 4 beschrieben - die Einhaltung aller Pfad-Constraints. Die Verkürzung der Gesamtlänge aller kritischen Pfade leistet dazu einen wichtigen Beitrag. Nur lange und kritische Pfade können Pfad-Constraints verletzen, während die Länge aller anderen Pfade unter dem Gesichtspunkt der Verzögerungszeiten nicht relevant ist. Der Anstieg der Länge der kurzen und mittleren Pfade sowie die Optimierung der Verdrahtung bezüglich der Pfadlängen durch eine Zweipunktbetrachtung aller Netze führen jedoch zu dem in Tabelle 6.6 dargestellten Anstieg der Gesamtverdrahtungslänge. Der Anstieg der Gesamtverdrahtungslänge ist bei dem Einsatz des neuen Platzierungsalgorithmus prinzipiell nicht vermeidbar, sondern die Konsequenz aus der Verkürzung der langen Pfade. Die Gesamtverdrahtungslängen in Tabelle 6.6 wurden mit einem Bounding-Box-Modell berechnet. Dieses Modell wurde zum Vergleich der Ergebnisse mit anderen veröffentlichten Ergebnissen verwendet. Ein Vergleich auf Basis der Pfadlängen ist zwar möglich, erlaubt aber keine Aussagen über die reale Gesamtverdrahtungslänge, da viele Netze zu mehreren Pfaden gezählt werden.

Design	Gesamtverdrahtungslänge Plato	Gesamtverdrahtungslänge neuer Designflow	Abweichung
Struct	448.767	586.525	+ 30,7 %
Primary1	924.621	1.254.436	+ 35,7 %
Biomed	1.974.648	8.836.630	+ 347,5 %
Industry1	3.229.482	4.216.223	+ 30,5 %
Industry2	15.066.023	24.978.412	+ 65,8 %

Tabelle 6.6: Vergleich der Gesamtverdrahtungslänge

Man erkennt in Tabelle 6.6 dass der Anstieg der Gesamtverdrahtungslänge von der Netzliste abhängt. Insbesondere bei einer starken Verkürzung der Summe der kritischen Pfade aufgrund der bereits beschriebenen Optimierung von Teilen von Mehrpunktnetzen steigt die Gesamtverdrahtungslänge deutlich an. Durch die höhere Gesamtverdrahtungslänge steigt bei gleicher Fläche die Belegung der Verdrahtungsressourcen an. Dieses kann dazu führen, dass in der Detailverdrahtungsphase einige Netze nicht verdrahtet werden können. Dieses Problem kann umgangen werden, indem zusätzliche Metalllagen verwendet werden.

Wie viele Platzierungswerkzeuge bietet „Plato“ einen „Timing-Driven-Placement“-Modus an, bei dem die Länge des längsten Pfades speziell minimiert wird. Für den Vergleich der Ergebnisse mit diesem Vorgehen, werden die Verzögerungszeiten der fünf längsten Pfade aufbauend auf einem RC-Modell für das Designbeispiel Biomed zwischen „Plato“ und den im Rahmen dieser Arbeit implementierten Algorithmus in Tabelle 6.7 verglichen. Dabei wird die Verdrahtungslänge für beide Algorithmen mit einer Bounding-Box je Zweipunktnetz abgeschätzt.

Pfad	Plato		Leitbahnorientierter Designflow	
	Länge	Delay	Länge	Delay
1	30.272	17,51 ns	20.096	15,91 ns
2	30.272	17,51 ns	19.654	15,56 ns
3	30.272	17,37 ns	19.486	15,31 ns
4	30.272	17,37 ns	19.450	15,30 ns
5	29.968	17,37 ns	19.360	15,21 ns

Tabelle 6.7: Vergleich der fünf längsten Pfade

Man erkennt in Tabelle 6.7, dass die Länge der kritischen Pfade bei dem leitbahnorientierten Designflow im Vergleich zu Plato um ca. 9 % kürzer ist.

Die Laufzeitkomplexität des neuen Platzierungsalgorithmus lässt sich anhand der Komplexität der kräftegesteuerten Platzierung abschätzen, wenn man davon ausgeht, dass die Pfad-Constraints in einer früheren Designphase - wie z.B. der statischen Timing-Analyse (vgl. Abschnitt 2.4) - berechnet worden sind. Kräftegesteuerte Platzierung hat wie in Abschnitt 3.1.1.1 beschrieben, eine experimentell ermittelte Laufzeitkomplexität von $n^{1,5}$. Während n in der Standardimplementierung die Anzahl der Zellen ist, ist n für den leitbahnorientierten Designflow die Summe aus virtuellen und realen Zellen. Damit ist n für einen Programmlauf nicht konstant und kann nur in Abhängigkeit von der maximalen Segmentlänge abgeschätzt werden.

Die Auswertung der Testbeispiele hat gezeigt, dass eine Zelle mit durchschnittlich vier Netzen verbunden ist und jedes Netz mit durchschnittlich fünf virtuellen Zellen modelliert wird. Damit erhöht sich die Anzahl der Zellen für den neuen Platzierungs- und Globalverdrahtungsalgorithmus durchschnittlich um den Faktor 20. Dieses führt zu einem nicht unerheblichen Anstieg der Laufzeit. Dieser wird - bezogen auf die Laufzeit des gesamten Designflows - zumindest teilweise durch den Wegfall der Globalverdrahtung, die gleichzeitig mit der Platzierung durchgeführt wird, wieder kompensiert. Trotzdem wird der neue Designflow im Regelfall eine höhere Laufzeit als ein konventioneller Designflow haben.

6.3.2 Detailverdrahtung

Die implementierte Detailverdrahtung entspricht dem in Kapitel 5 beschriebenen Verfahren. Aufgrund der Beschränkung der Anzahl von Verdrahtungselementen stehen insgesamt ca. 20 verschiedene Bibliothekselemente zur Verfügung. Die Vias in allen Elementen sind - wie in Abschnitt 5.1 beschrieben - an den Symmetriepunkten der Shapes angeordnet.

Durch diese Einschränkungen kann in den meisten Layouts keine Verdrahtung mit der gleichen Anzahl von Metalllagen wie mit einem konventionellen Detailverdrahter (vgl. Abschnitt 3.1.2.2) erzeugt werden. Netze, die Kurzschlüsse verursachen, können entfernt und anschließend mit einem Maze-Router (vgl. Abschnitt 3.1.2.2.1) verdrahtet werden. Dabei kann die Einhaltung der Timing-Restriktionen jedoch nicht mehr garantiert werden.

Tabelle 6.8 zeigt die Anzahl von Kurzschlüssen, die durch die prototypische Implementierung des bibliotheksbasierten Verdrahtungsalgorithmus erzeugt werden, wenn man die Anzahl der Verdrahtungsebenen wie bei einem konventionellen Verdrahter wählt. Dabei werden die Ergebnisse der zwei verschiedenen Akzeptanzstrategien - monotone Suche und Simulated Annealing (vgl. Anschnitt 5.2.2.6) - gegenübergestellt. Als Testbeispiele werden die Beispiele Struct, Primary1, Industry1 und Biomed gewählt.

Aufgrund der hohen Laufzeiten durch die prototypische Implementierung in Matlab wurde auf die Verdrahtung der Schaltung Industry2 verzichtet.

Design	Anzahl Überlappungen Initialverdrahtung	Anzahl Überlappungen Monotone Suche	Anzahl Überlappungen Simulated Annealing
Struct	2.830	1.703	1.347
Primary1	2.577	1.317	958
Biomed	6.215	1.630	1.279
Industry1	2.612	489	445

Tabelle 6.8: Vergleich der Optimierungsverfahren

Man erkennt in Tabelle 6.8, dass das Optimierungsverfahren Simulated Annealing im Vergleich zur monotonen Suche die besseren Ergebnisse - entsprechend einer geringeren Anzahl von Überlappungen - erzeugt. Dieses liegt daran, dass durch die Akzeptanz von Verschlechterungen der Kostenfunktion lokale Minima verlassen werden können.

Ein großer Anteil der Überlappungen wird durch Mehrpunktnetze mit einer großen Anzahl von getriebenen Zellen - wie zum Beispiel dem Reset-Netz - verursacht. Bei diesen Netzen ist die Verdrahtungslänge durch die Zweipunktverdrahtung deutlich erhöht und führt so zu vielen Überlappungen. Außerdem liegen in den verwendeten Standardzellbibliotheken alle Pins auf gleicher Höhe in den Standardzellen. Dieses kann bei den zur Verfügung stehenden Verdrahtungs-Shapes zu vielen Überlappungen führen. Im Vergleich zu der Verdrahtungsrate von ca. 80% des Shape-basierten Verdrahtungsansatzes von [Kastner00] erscheint die relativ hohe Anzahl von Überlappungen dieses Ansatzes für einen prototypisch implementierten Shape-basierten Verdrahter ohne Mehrpunktnetzverdrahtung akzeptabel.

Bei der Optimierung der Verdrahtungsergebnisse wurden die Auswirkungen der Anteile von L- und Z-Shapes exemplarisch am Beispiel Primary1 untersucht. Dazu wurde der Anteil der Z-Shapes festgelegt und die restlichen Shapes als L- oder G-Shapes gewählt, wobei der Anteil der G-Shapes meistens unter 5% lag, da G-Shapes nur bei vertikal oder horizontal auf einer Linie liegenden Pins verwendet werden. Die Ergebnisse für 10.000 Modifikationsschritte bei monotoner Suche können Tabelle 6.9 entnommen werden. Dabei erkennt man, dass ein höherer Anteil von Z-Shapes zu einer Reduktion der Anzahl von Überlappungen führt.

Die in Tabelle 6.8 dargestellten Verdrahtungsergebnisse wurden aufgrund der Ergebnisse aus Tabelle 6.9 mit einem Anteil von 80% Z-Shapes erzeugt.

Anteil Z-Shapes	Anzahl Überlappungen
20%	1.643
40%	1.596
60%	1.483
80%	1.470
100%	1.542

Tabelle 6.9: Überlappungen in Abhängigkeit vom Anteil Z-Shapes

Versuchsweise wurde zur Verbesserung der Verdrahtung die Anzahl der Metallebenen erhöht und Vias über mehrere Ebenen hinweg (Stacked Vias) zugelassen. Da diese Vias aber die entsprechenden Spuren in allen dazwischen liegenden Ebenen blockieren, wurde dadurch keine deutliche Verbesserung des Verdrahtungsergebnisses erzielt.

Die Anzahl der Kurzschlüsse im Layout könnte durch verschiedene Maßnahmen deutlich verringert werden. Zunächst könnte für Spezialnetze wie Takt und Reset eine Sonderbehandlung eingeführt werden. Dieses würde einen zusätzlichen Algorithmus erfordern, der vergleichbar zu einem „Clock-Tree-Generator“ in einem konventionellen Designflow arbeitet. Zusätzlich könnte die Platzierung der Vias an beliebigen Positionen auf den Shapes erfolgen. Dazu könnten die Via-Positionen zufällig erzeugt werden. Zusätzlich würde das Einfügen von Shapes, die die Zellen nicht auf dem direkten Weg verbinden, eine Verbesserung der Verdrahtungsqualität bringen. Dazu zählen zum Beispiel U-förmige Shapes.

Zur Erzeugung der überlappfreien Verdrahtung ist daher ein zusätzlicher Optimierungsschritt erforderlich. In diesem Schritt müssen in der Nähe von Überlappungen weitere virtuelle Zellen in jedes an der Überlappung beteiligte Netz eingefügt werden. Anschließend müssen die bisher verwendeten Verdrahtungs-Shapes entfernt werden und unter Verwendung der neuen virtuellen Zellen neue Verdrahtungs-Shapes eingefügt werden. Durch die zusätzlichen virtuellen Zellen entstehen zwischen jeweils zwei realen Zellen komplexere Verdrahtungsformen, die in der Regel zu weniger Überlappungen führen können.

Zusätzlich sind Verbesserungen zu erwarten, wenn die Auswahl der Shapes und der Vias sowie der Via-Positionen nicht mehr nur zufällig, sondern heuristisch unter Berücksichtigung der Belegung der Verdrahtungsressourcen durch andere Leitungen erfolgt.

Die Laufzeitkomplexität des Verdrahtungsverfahrens orientiert sich an der Laufzeitkomplexität von Simulated Annealing. Diese hängt von verschiedenen

implementierungsabhängigen Faktoren und der Wahl der Parameter - wie beispielsweise der Abkühlkurve ab - und kann nicht allgemein angeben werden [Sechen98].

In Abbildung 6.2 ist exemplarisch für die Schaltungen der MCNC-Benchmark-Suite das Layout der Benchmark-Schaltung Primary1 dargestellt. Man erkennt im Layout eine gleichmäßige Zellverteilung und eine gleichmäßig hohe Verdrahtungsdichte über der Layoutfläche. Diese beiden Kriterien sind die einzigen, die unmittelbar aus dem Layout erkennbar sind. Eine möglichst gleichmäßige Zell- und Leitungsverteilung über der Layoutfläche entspricht dabei dem idealen Zustand.

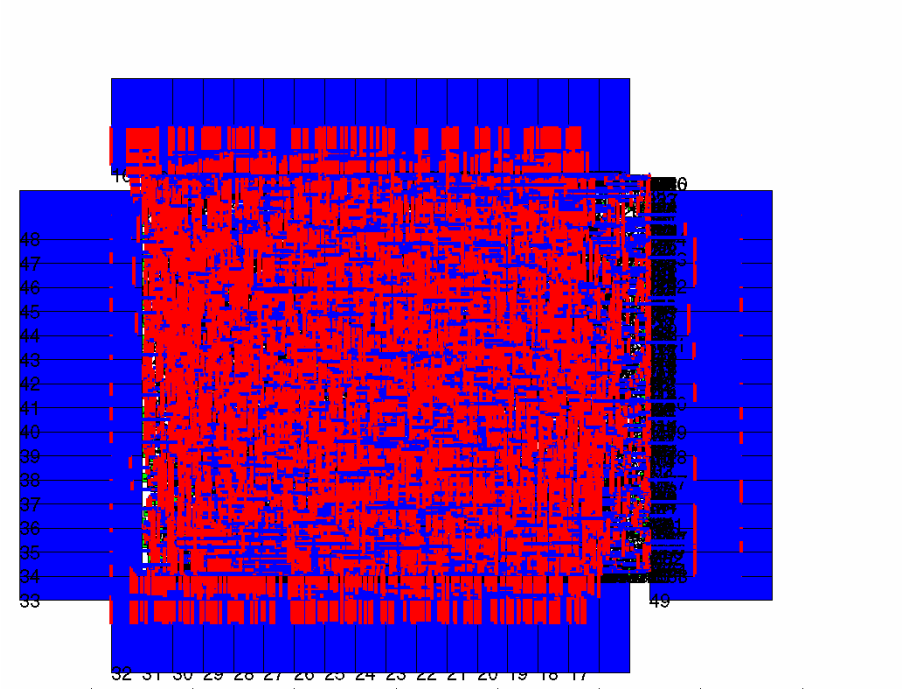

Abbildung 6.2: Exemplarisches Layout

6.3.3 Gesamt-Designablauf

Der im Rahmen dieser Arbeit entwickelte Designflow soll einen Beitrag zur Lösung des Timing-Closure-Problems leisten. Deshalb ist das primäre Ziel, alle Pfad-Restriktionen einzuhalten und die Abweichungen zwischen den in frühen Designphasen geschätzten Verzögerungszeiten und den Verzögerungszeiten des Layouts gering zu halten. Zur Überprüfung der Abweichungen der Verzögerungszeiten werden die geschätzten Pfadlängen nach der Platzierung und Globalverdrahtung mit den Pfadlängen nach der bibliotheksbasierten Detailverdrahtung verglichen. Dabei werden Vias als eine - bezüglich der Verzögerungszeit - äquivalent lange Leitung berücksichtigt.

Tabelle 6.10 zeigt die mittlere Abweichung der geschätzten Pfadlängen aus Tabelle 6.6 im Vergleich zu der Summe der Längen der verdrahteten Netze jedes Pfads für alle Testdesigns.

Design	Mittlere Abweichung
Struct	4,30 %
Primary1	5,35 %
Biomed	4,72 %
Industry1	0,61 %

Tabelle 6.10: Mittlere Abweichung der Pfadverdrahtungslängen

Die Mittelwertbildung der Ergebnisse aus Tabelle 6.10 führt dazu, dass die höheren Abweichungen einiger Pfade nicht mehr ersichtlich sind. Deshalb wird die maximale Abweichung der realen von der geschätzten Pfadlänge für jedes Design in der folgenden Tabelle 6.11 dargestellt.

Design	Max. Abweichung der Verdrahtungslänge
Struct	7,98 %
Primary1	6,59 %
Biomed	6,21 %
Industry1	8,51 %

Tabelle 6.11: Maximale Verdrahtungslängenabweichung

Man erkennt in den Tabellen 6.10 und 6.11, dass die Abweichungen zwischen den geschätzten und den realen Verzögerungszeiten in diesem Designflow sehr klein sind. Dadurch ist es nahezu ausgeschlossen, dass Iterationen zwischen der Platzierungs- und der Verdrahtungsphase erforderlich werden.

6.4 Bewertung

Insgesamt lässt sich feststellen, dass der im Rahmen dieser Arbeit implementierte Algorithmus in der Lage ist, einen Beitrag zur Lösung des Timing-Closure-Problems zu leisten.

Durch die simultane Platzierung und Globalverdrahtung erfolgt während der Platzierung eine so genaue Modellierung der Verdrahtung, dass die modellierten Verzögerungs-

zeiten nur geringfügig von den Verzögerungszeiten der Verdrahtung abweichen. Dadurch kann zwischen der Platzierung und der Verdrahtung mit hoher Wahrscheinlichkeit kein Timing-Closure-Problem mehr auftreten. Aufgrund der eingeführten Pfadkräfte treten in den Testdesigns nach der Platzierung keine oder deutlich weniger Pfad-Constraint-Verletzungen als bei der Verwendung von konventionellen Platzierungs- und Verdrahtungswerkzeugen auf. Dieses trägt zur Vermeidung von Iterationen zwischen Synthese und Platzierung bei. Die bibliotheksbasierte Verdrahtung ermöglicht eine Detailverdrahtung mit Elementen, deren Verzögerungszeiten bereits vor der Verdrahtung bekannt sind und deren Auswahl während der Verdrahtung passend zu den Constraints erfolgt.

Dieser Ansatz hat jedoch auch einige Nachteile. Zum einen steigt die Laufzeit der Platzierung aufgrund der deutlich höheren Zellanzahl im Vergleich zu anderen kräftegesteuerten Platzierungsverfahren deutlich an. Zum anderen ermöglicht die bisher nur prototypisch implementierte Detailverdrahtung keine Verdrahtung unter Einhaltung aller Design-Rules. Weiterhin fehlt diesem Ansatz noch eine bessere Behandlung von Mehrpunktnetzen. Die Zerlegung dieser Netze in mehrere Zweipunktnetze führt zu unrealistischen Verdrahtungslängen im Vergleich zu anderen Designflows und zu unter Umständen zu klein berechneten Verzögerungszeiten aufgrund von zu gering abgeschätzten Leitungskapazitäten.

Die mit dem Platzierungs- und Verdrahtungsalgorithmus erzielten Laufzeiten liegen zwischen einigen Minuten und einigen Stunden in Abhängigkeit von der Designgröße. Diese hohen Laufzeiten sind vorwiegend auf die Implementierung in Matlab zurückzuführen. Sowohl die beschränkten Möglichkeiten bei der Verwendung der Datenstrukturen als auch die Interpretation der Matlab-Befehle zur Programmlaufzeit lassen einen deutlichen Geschwindigkeitsgewinn bei einer Implementierung in C erwarten.

7 Zusammenfassung

In der vorliegenden Arbeit wurde ein neuer Designflow für einen leitbahnzentrierten Designflow vorgestellt. Dieser Entwurfsablauf besteht aus zwei neuen Algorithmen. Der erste Algorithmus ermöglicht eine simultane Platzierung und Globalverdrahtung unter Berücksichtigung von Pfad-Constraints, der zweite ist ein bibliotheksbasierter Detailverdrahter, der auf einer Bibliothek von Leitungssegmenten aufbaut.

Das implementierte Verfahren zur gleichzeitigen Platzierung und Globalverdrahtung basiert auf der kräftegesteuerten Platzierung. Die Globalverdrahtung wird dabei durch so genannte virtuelle Zellen repräsentiert. Diese werden gleichzeitig mit den realen Zellen auf der Layoutfläche platziert und definieren Punkte, durch die die Detailverdrahtung in den späteren Entwurfsschritten verlaufen soll. Zusätzliche Pfadkräfte, die proportional zur Differenz aus Taktperiode und Pfadverzögerung sind, verhindern das Auftreten von Constraint-Verletzungen.

Die Detailverdrahtung wird in dieser Arbeit nicht wie bei konventionellen Verdrahtungsverfahren durch Wegesuche erzeugt, sondern basiert auf der Platzierung von vordefinierten Verdrahtungselementen. Diese Verdrahtungselemente werden einmalig charakterisiert und in einer Bibliothek abgespeichert. Während der Detailverdrahtung erfolgt die Selektion der verschiedenen Bibliothekselemente mit einem für die Verdrahtung modifizierten Simulated-Annealing-Verfahren.

Die Ergebnisse haben gezeigt, dass diese Algorithmen einen Beitrag zur Lösung des Timing-Closure-Problems leisten können. Aufgrund der prototypischen Implementierung konnten nur kleinere Beispiele mit bis zu ca. 15.000 Zellen untersucht werden. Durch den neuen Platzierungsansatz reduziert sich sowohl die Anzahl als auch die Länge der kritischen Pfade. Gleichzeitig ermöglicht der Verdrahtungsansatz eine sehr genaue Vorhersage der Verdrahtungslängen bereits während der Platzierung. Dadurch kann die Platzierung bereits vor der Verdrahtung optimiert werden, so dass mit hoher Wahrscheinlichkeit keine Iterationen zwischen Platzierung und Verdrahtung auftreten.

Der Einsatz von Designabläufen, die die Eigenschaften der Leitbahnen bereits in früheren Phasen des Entwurfsprozesses berücksichtigen, wird in naher Zukunft eine immer größere Bedeutung erlangen. Dies ist darauf zurückzuführen, dass in Technologien mit Strukturgrößen von weniger als 0,13 µm die Verzögerungszeiten der Leitungen ungefähr genauso groß wie die Verzögerungen der Gatter werden. Während die Gatterverzögerungszeiten bereits in der Synthese berechnet werden, erfolgt zurzeit eine Berechnung der Leitungsverzögerungen mit einer ausreichenden Genauigkeit erst nach der Platzierung. Um das Auftreten von Iterationen zwischen Platzierung, Verdrahtung und Synthese vermeiden zu können, müssen Entwurfsverfahren wie das hier vorgestellte, in Zukunft eingesetzt werden.

Literaturverzeichnis

[Adler01] T. Adler, *Algorithmen zur optimierten Verdrahtung integrierter Analogschaltungen*, Dissertation, Fortschritts-Berichte VDI, Reihe 20, Nr. 336, 2001

[Antreich82] K. J. Antreich, F. M. Johannes, F. H. Kirsch, *A New Approach for Solving the Placement Problem Using Force Models*, Proceedings International Symposium on Circuits and Systems, 1982

[AutoBEM] Coyote Sytems Inc., Online Help, 2000

[Bodapi00] S. Bodapati, F. N. Najm, *Pre-Layout Estimation of Individual Wire Lengths*, Proceedings System Level Interconnection Prediction, S. 91-96, 2000

[Brück93] R. Brück, *Entwurfswerkzeuge für VLSI-Layout*, Hanser-Verlag, 1993

[Caldwell99] A. E. Caldwell, A. B. Kahng, S. Mantik, A. Zelikovsky, *On Wirelength Estimations for Row-Based Placement*, IEEE Transactions on Computer-Aided Design, Bd. 18, Nr. 9, S. 1265-1278, 1999

[Chaudary93] K. Chaudary, A. Onozawa, E. S. Kuh, *A Spacing Algorithm for Performance Enhancement and Crosstalk Reduction*, Proceedings International Conference on Computer Aided Design, S. 697-702, 1993

[Chen91] H. Y. Chen, S. M. Kang, *iCoach: A Circuit Optimization Aid for CMOS High Performance Circuits*, Integration, the VLSI Journal, Bd. 10, S. 185-212, 1991

[Chou01] Y.-C. Chou, Y.-L. Lin, *A Performance-Driven Standard-Cell Placer Based on a Modified Force-Directed Algorithm*, Proceedings International Symposium on Physical Design, S. 24-29, 2001

[Chou02] Y.C. Chou, Y.-L. Lin, *Effective Enforcement of Path-Delay Constraints in Performance-Driven Placement*, IEEE Transactions on Computer Aided Design, Bd. 21, Nr. 1, S. 15-21, 2002

[Cong00] J. Cong, J. Fang, K. Khoo, *DUNE: A Multi-Layer Gridless Routing System with Wire Planning*, Proceedings International Symposium on Physical Design, S. 12-18, 2000

[Cong01] J. Cong, J. Fang, Y. Zhang, *Multilevel Approach to Full-Chip Gridless Routing*, Proceedings International Conference on Computer-Aided Design, S. 396-403, 2002

[Cong97a] J. Cong, L. He, C.-K. Koh, Z. Pan, *Interconnect Design for Deep Submicron ICs,* Proceedings International Conference on Computer-Aided Design, S. 478-485, 1997

[Cong97b] J. Cong, L. He, C.-K. Koh, Z. Pan, *Global Interconnect Sizing and Spacing with Consideration of Coupling Capacitance*, Proceedings International Conference on Computer-Aided Design, S. 628-633, 1997

[Eisenmann98] H. Eisenmann, F. M. Johannes, *Generic Global Placement and Floorplanning*, Proceedings Design Automation Conference, S. 269-274, 1998

[Eisenmann99] H. Eisenmann, *Ein universelles Plazierungsverfahren für integrierte Schaltungen*, Dissertation, Technische Universität München, 1999

[Elmore48] W.C. Elmore, *The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifier*, Journal of Applied Physics, Bd. 19, S. 55-63, 1948

[Gajski83] D. Gajski, R. H. Kuhn, *New VLSI-Tools*, IEEE Computer, Bd. 16, Nr. 12, S.11-14, 1983

[Gosti01] W. Gosti, S. Khatri, A. Sangiovanni-Vincentelli, *Adressing the Timing Closure Problem by Integrated Logic Optimization*, Proceedings International Conference on Computer-Aided Design, S. 224-231, 2001

[Hart68] P. Hart, N. Nilson, B. Raphael, *A Formal Basis for the Heuristic Determination of Minimum Cost Paths*, IEEE Transactions on Systems, Science, Cybernetics, Bd. SCC-4, S. 1100-107, 1968

[Hightower69] D. W. Hightower, *A Solution to Line Routing Problems on the Continuous Plane*, Design Automation Workshop, S. 1-24, 1969

[Hu00] J. Hu, S. S. Sapatnekar, *A Timing-Constrained Algorithm for Simultaneous Global Routing of Multiple Nets*, Proceedings International Conference on Computer-Aided Design, S. 99-103, 2000

[Hur99] S. Hur, A. Jagannathan, J. Lillis, *Timing Driven Maze Routing*, Proceedings International Symposium on Physical Design, S. 208-213, 1999

[ITRS02] International Technology Road Map for Semiconductors, http://public.itrs.net/Files/2002Update/Home.pdf

[Jagannathan00] A. Jagannathan, S.-W. Hur, J. Lillis, *A Fast Algorithm for Context-Aware Buffer Insertion*, Proceedings Design Automation Conference, S. 368-373, 2000

[Jhang96] K. Jhang, S. Ha, C.S. Jhon, *COP: A Crosstalk Optimizer for Gridda Channel Routing*, IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, Vol. 15, S. 424-429, 1996

[Karypis99] G. Karypis, V. Kumar, *Multilevel k-way Hypergraph Partitioning*, Proceedings Design Automation Conference, S. 343-348, 1999

[Kastner00] R. Kastner, E. Bozorgzadeh, M. Sarrafzadeh, *Predictable Routing*, Proceedings International Conference on Computer-Aided Design, S. 110-113, 2000

[Keutzer87] K. Keutzer, *DAGON: Technology Mapping and Local Optimization*, Proceedings Design Automation Conference, S. 341-347, 1987

[Keutzer97] K. Keutzer, A. R. Newton, N. Shenoy, *The Future of Logic Synthesis and Physical Design in Deep-Submicron Process Geometries*, International Symposium on Physical Design, S. 218-224, 1997

[Khatri99] S. Khatri, A. Mehtrota, R. Brayton, A. Sangiovanni-Vincentelli, *A Novel VLSI Layout Fabric for Deep Sub-Micron Applications*, Proceedings Design Automation Conference, S. 491-496, 1999

[Kirkpatrick83] S Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Optimization by Simulated Annealing*, Science 220(4598), S. 671- 680, 1983

[Kleinhans88] J. M. Kleinhans, G. Sigl, F. M. Johannes, *GORDIAN: A New Global Optimization / Rectangle Dissection Method for Cell Placement*, Proceedings International Conference on Computer-Aided Design, S. 506-509, 1988

[Kleinhans91] J. M. Kleinhans, G. Sigl, F.M. Johannes, K. J. Antreich, *GORDIAN: VLSI Placement by Quadratic Programming and Slicing Optimization*, IEEE Transactions on CAD, Bd. 10, Nr. 3, S. 356-365, 1991

[Kong02] T. Kong, *A Novel Net Weighting Algorithm for Timing-Driven Placement*, Proceedings International Conference on Computer-Aided Design, S. 172-176, 2002

[Lalgudi94] L. N. Kannan, P. R. Suaris, H.-G. Fang, *A Methodology and Algorithms for Post-Placement Delay Optimization*, Proceedings Design Automation Conference, S. 327-332, 1994

[Lee61] C. Y. Lee, *An Algorithm for Patch Connections and Its Applications*, IRE Transactions on Electronic Computers, Bd. EC-10, S.364-365, 1961

[Lin02] S. Lin, Y. Chang, *A Novel Framework for Multilevel Routing Considering Routability and Performance*, Proceedings International Conference on Computer-Aided Design, S. 44-50, 2002

[Lou97] J. Lou, A. H. Salek, M. Pedram, *An Exact Solution to Simultaneous Technology Mapping and Linear Placement Problem*, Proceedings International Conference on Computer-Aided Design, S. 671-675, 1997

[Madden02] Patrick H. Madden, "*Reporting of Standard Cell Placement Results*", IEEE Transactions on Computer Aided Design, Bd. 21, Nr. 2, 2002

[MAGMA02] Magma Design Systems, *The Fixed Timing Methodology*, White Paper, 2002

[Mikami68] K. Mikami, K. Tabuchi, *A Computer Program for Optimal Routing of Printed Circuit Connectors*, Proc. IFIPS, Bd. H-47, S. 1475-1478, 1968

[Mo01a] F. Mo, A. Tabbara, R. K. Brayton, *A Timing-driven Macro-Cell Placement Algorithm*, Proceedings International Conference on Computer Design, S. 322-327, 2001

[Mo01b] F. Mo, A. Tabbara, R. K. Brayton, A *Force-Directed Maze Router*, Proceedings International Conference on Computer-Aided Design, S. 404-407, 2001

[MONTEREY01] Monterey Design Systems, *System-Driven Physical Design*, White Paper, 2001

[Moore59] E. F. Moore, *The Shortest Path through a Maze*, Annals of the Harvard Computation Laboratory, S. 185-292, 1959

[Ohtsuki86] T. Ohtsuki, *Maze-Running and Line-Searching Algorithms*, Layout Design and Verification, Bd. 4, Elsevier Science Publishers, 1986

[Onodera91] H. Onodera, Y. Taniguchi, K. Tamaru, *Branch-and-Bound Placement for Building Block Layout*, Proceedings Design Automation Conference, S. 433-439, 1991

[Ottmann90] T. Ottmann, P. Widmayer, *Algorithmen und Datenstrukturen*, BI Wissenschaftsverlag, 1990

[Ou00] S. Ou, M. Pedram, *Timing-driven Placement Based on Partitioning with Dynamic Cut-Net Control*, Proceedings Design Automation Conference, S. 472-476, 2000

[Ou99] S. Ou, M. Pedram, *Timing-Driven Bipartitioning with Replication Using Iterative Quadratic Programming*, Proceedings Asia and South Pacific Design Automation Conference, S 105-108, 1999

[Paskalev02] K. Paskalev, *Wire Delay Models for Global Placement of ASICs*, Bachelor Thesis, Massachusetts Institute of Technology, 2001

[Pedram89] M. Pedram, B. T. Preas, *Interconnection Length Estimation for Optimized Standard Cell Layouts*, Proceedings International Conference on Computer-Aided Design, S. 390-393, 1989

[Pedram91] M. Pedram, N. Bhat, *Layout Driven Technology Mapping*, Proceedings Design Automation Conference, S. 99-105, 1991

[Rohfleisch95] B. Rohfleisch, B. Wurth, K. Antreich, *Logic Clause Analysis for Delay Optimization*, Proceedings Design Automation Conference, S. 668-672, 1995

[Rubinstein83] J. Rubinstein, *Signal Delay in RC Tree Networks*, IEEE Transactions on Computer-Aided Design, Bd. 2, Nr. 3, S. 202-210, 1983

[Salek98] A. H. Salek, J. Lou, M. Pedram, *A DSM Design Flow: Putting Floorplanning, Technology-Mapping, and Gate-Placement Together*, Proceedings International Conference on Computer-Aided Design, S. 128-133, 1998

[Salek99] A.H. Salek, J.Lou, M. Pedram, *An Integrated Logical and Physical Design Flow for Deep Submicron Circuits*, IEEE Transactions on Computer-Aided Design, Bd. 18, Nr. 9, S. 1305-1315, 1999

[Sarrafzadeh97] M. Sarrafzadeh, M. Wang, *NRG: Global and Detailed Placement*, International Conference on Computer-Aided Design, S. 532-537, 1997

[Saxena99] P. Saxena, C. L. Liu, *Crosstalk Minimization Using Wire Perturbations*, Proceedings Design Automation Conference, S. 100-103, 1999

[Sechen98] C. Sechen, K. Lee, *VLSI Placement and Global Routing Using Simulated Annealing*, Kluwer Academic Publishers, 1998

[Scheffer01] L. Scheffer, *Timing Closure Today*, Firmenpräsentation Cadence Design Systems, Asia and South Pacific Design Automation Conference, 2001

[Sherwani95] N. A. Sherwani, *Algorithms for VLSI Physical Design Automation*, Second Edition, Kluver Academic Publishers, 1995

[Sigl91] G. Sigl, K. Doll, F. M. Johannes, *Analytical Placement: A Linear or a Quadratic Objective Function?*, Proceedings Design Automation Conference, S. 427-431, 1991

[Spiro90] H. Spiro, *Simulation integrierter Schaltungen*, Oldenbourg Verlag, 1990

[Steiner] www.tik.ee.ethz.ch/tik/education/lectures/AFK/S01_02/Steiner.pdf

[Stenz97] G. Stenz, B. M. Riess, B. Rohfleisch, F. M. Johannes, *Timing Driven Placement in Interaction with Netlist Transformations*, International Symposium on Physical Design, S. 36-41, 1997

[Stroobandt01] D. Stroobandt, *A Priori Wire Length Estimates for Digital Design*, Kluver Academic Publishers, 2001

[Sundararajan00] V. Sundararajan, S. S. Sapatnekar, K. K. Parhi, *Minflotransit: Min-Cost Flow Based Transistor Sizing Tool*, Proceedings Design Automation Conference, S. 649-654, 2000

[Swartz95] W. Swartz, C. Sechen, *Timing Driven Placement for Large Standard Cell Circuits*, Proceedings Design Automation Conference, S. 211-215, 1995

[Synopsys_PC] Synopsys Inc., *Physical Compiler*, Handbuch, 2002

[Tseng98] H. Tseng, L. Scheffer, C. Sechen, *Timing and Crosstalk Driven Area Routing*, Proceedings Design Automation Conference, S. 378-381, 1998

[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

[Yang02] X. Yang, B. Choi, M. Sarrafzadeh, *Timing-Driven Placement Using Design Hierarchy Guided Constraint Generation*, Proceedings International Conference on Computer-Aided Design, S. 177-180, 2002

[Yannakakis85] M. Yannakakis, *A Polynomial Algorithm for the Min-Cut Linear Arrangement of Trees*, Journal of the Association for Computer Machinery, Bd. 32, Nr. 4, S. 950-988, 1985

[Zhou96] H. Zhou, D. F. Wong, *An Optimal Algorithm for River Routing with Crosstalk Constraints*, Proceedings International Conference on Computer-Aided Design, S. 310-315, 1996

[Zhou99] H. Zhou, D. F. Wong, I.-M. Liu, A. Aziz, *Simultaneous Routing and Buffer Insertion with Restrictions on Buffer Locations*, Proceedings Design Automation Conference, S. 96-99, 1999

Anhang

Parameter für das Einlesen von Daten

Parameter.InputFormat	Legt Eingabedatenformat fest 0 = Pfadformat, 1=Yal-Format
Parameter.Width	Breite der Layoutfläche (nur für Pfadformat)
Parameter.Height	Höhe der Layoutfläche (nur für Pfadformat)
Parameter.Rows	Anzahl Zellreihen (nur für Pfadformat)
Parameter.Sites	Anzahl der Sites (nur für Pfadformat)
Parameter.RowHeight	Höhe der Zellreihen (nur für Pfadformat)
Parameter.SiteWidth	Breite einer Site (nur für Pfadformat)
Parameter.RowDistance	Abstand der Zellreihen (nur für Pfadformat)
Parameter.CellHeight	Höhe der Standardzellen (nur für Pfadformat)
Parameter.WidthOrArea	Library enthält Breite oder Fläche der Zellen Breite = 0, Fläche = 1
Parameter.ReadNetlist	Netzliste einlesen, wenn 1
Parameter.GenerateGraph	Datenstruktur aufbauen, wenn 1
Parameter.GenerateCostmatrix	Kostenmatrix aufbauen, wenn 1
Parameter.GeneratePath	Pfade bestimmen, wenn 1

Parameter zur Ablaufsteuerung

Parameter.Place	Platzierung durchführen, wenn 1
Parameter.ReadPlacement	Platzierung einlesen, wenn 1
Parameter.PlacementFormat	Format der einzulesenden Platzierung Neuer Designflow = 0; Dragon Format = 1, Plato = 2
Parameter.NumberofIterations	Anzahl der Platzierungsiterationen
Parameter.Route	Verdrahtung durchführen, wenn 1
Parameter.Legalize	Legalisierung durchführen, wenn 1

Parameter zur Steuerung der Platzierung

Parameter.ForceRatio	Skalierungsfaktor für alle Kräfte
Parameter.WeightRatio	Skalierungsfaktor für die abstoßenden Kräfte. Je größer WeightRatio, desto größer sind die abstoßenden Kräfte im Vergleich zu den anziehenden.
Parameter.NetWeight	Netzgewicht für die Kostenmatrix

Parameter.HorizontalGrid	Horizontale Diskretisierung für die Berechung der abstoßenden Kräfte
Parameter.VerticalGrid	Vertikale Diskretisierung für die Berechung der abstoßenden Kräfte

Parameter zur Steuerung der simultanen Platzierung und Globalverdrahtung

Parameter.SimultaneousPR	Gleichzeitiges Platzieren und Globalverdrahten aktivieren (0 = aus, 1 = an)
Parameter.FirstInsertionStep	Erster Schritt, in dem virtuelle Zellen eingefügt werden-
Parameter.InsertNodesStepCount	Anzahl von Schritten, nach denen wieder neue virtuelle Zellen eingefügt werden, mind. 3
Parameter.WireSegmentLength	Max. Entfernung zwischen 2 Kraftangriffspunkten
Parameter.NodeweightRatio	Gewichtsverhältnis für virtuelle und normale Knoten in Kostenmatrix
Parameter.VNodeSize	Breite und Höhe der virtuellen Knoten für die Dichtematrix
Parameter.ReadCellDelay	Zellverzögerungszeiten einlesen
Parameter.MaxPathlength	Max. Länge eines Pfads
Parameter.UsePathForce	Pfadkräfte ein- und ausschalten (0= aus, 1=ein)
Parameter.CalculatePathlength	Pfadlängen berechnen

Parameter zur Steuerung der Verdrahtung

Parameter.Layer	Anzahl Verdrahtungslayer
Parameter.AcceptMode	Monotone Suche = 0, Sim. Annealing = 1
Parameter.RoutingSteps	Anzahl Iterationsschritte für monotone Suche
Parameter.StartTemp	Starttemperatur für Sim. Annealing
Parameter.EndTemp	Endtemperatur für Sim. Annealing
Parameter.ChangeTempSteps	Anzahl der Schritte, nach denen die Temperatur reduziert wird.
Parameter.Alpha	Faktor, um den die Temperatur reduziert wird.

Parameter zur Steuerung der Ausgabe

Parameter.Debuglevel	Bestimmt Menge der anzeigten Informationen über den Programmablauf (keine = 0 – alle = 3).
Parameter.GenerateVerilog	Schreibt Verilog-Netzliste
Parameter.Psplot	Ausgabe der Ergebnisse als eps-Datei

Parameter.Figplot	Ausgabe der Ergebnisse als fig-Datei
Parameter.VisualizeRepellingF.	Abstoßende Kraft durch Pfeile visualisieren (an=1, aus = 0)
Parameter.VisualizeForcefield	Komplettes Kraftfeld anzeigen, wenn 1
Parameter.DrawFlylines	Verbindungen zwischen Zellen einzeichnen, wenn 1

Lebenslauf

Name:	Carsten Malonnek
Geboren:	06.02.1973 in Hannover
Staatsangehörigkeit:	deutsch
Familienstand:	ledig

Schulausbildung:

1979 – 1983	Grundschule Bennigsen, Bennigsen
1983 – 1985	Orientierungsstufe Nord, Springe
1985 – 1992	Otto-Hahn-Gymnasium, Springe

Zivildienst:

1992 – 1993	Zivildienst in Springe

Studium

1993 – 1998	Studium der Elektrotechnik mit Studienrichtung Technische Informatik, Universität Hannover

Beruflicher Werdegang:

1998 - 2003	Wissenschaftlicher Mitarbeiter am Institut für Mikroelektronische Systeme, Universität Hannover
seit 1.11.2003	Entwicklungsingenieur bei der ADIT GmbH, Hildesheim