Constraint-geführte Floorplan-Generierung von integrierten Analog- und Mixed-Signal-Schaltungen

Ammar Nassaj, Jens Lienig, Institut für Feinwerktechnik und Elektronik-Design (IFTE), TU Dresden, Dresden
Göran Jerke, Jan Freuer, Robert Bosch GmbH, Reutlingen

Kurzfassung

Dieser Beitrag stellt ein neues constraint-geführtes Konzept zur Floorplan-Generierung (Floorplanning) für Analog- und Mixed-Signal-Schaltungen vor. Unser Verfahren ermöglicht eine zeiteffiziente Constraint-Verifikation vor der eigentlichen Floorplan-Optimierung. Dabei wird mittels einer analytischen Vorgehensweise (Linear Programming) ein Anfangsfloorplan ermittelt, der ein globales Optimum hinsichtlich der Constrainteinhaltung darstellt. Nach dieser ausschließlichen Berücksichtigung der Constraints erfolgt in einem zweiten Schritt eine Einbeziehung der Optimierungsziele. Dies geschieht mittels eines modifizierten heuristischen Verfahrens (Simulated Annealing). Dieses nutzt dabei eine adaptive Constraint-Wichtung, was die Suche in dem durch die Constraints aufgespannten Lösungsraum optimiert. Die Anwendung des hier vorgestellten Verfahrens ist insbesondere dann sinnvoll, wenn zahlreiche verknüpfte und in sich widersprüchliche Constraints bei der Layouterstellung zu berücksichtigen sind.

1 Einleitung

Beim Layoutentwurf von integrierten Schaltungen unterscheidet man zwischen Optimierungszielen und Constraints (Randbedingungen) [1]. Optimierungsziele, wie die minimale Chipfläche oder Verbindungslänge, dienen zur Verbesserung der Leistungsparameter und Zuverlässigkeit einer Schaltung. Sie drücken damit die Qualität des entwickelten Layouts aus. Gleichzeitig sind jedoch auch *Constraints* zu beachten, deren Einhaltung für die Schaltungsfunktion bzw. Realisierung des Layouts zwingend notwendig ist (**Bild 1**). Die technologische Weiterentwicklung zu kleineren Strukturabmessungen führt neben der Verschärfung bekannter Constraints auch zur Einbeziehung neuer Arten von Constraints. Damit ist die Constrainteinhaltung bei gleichzeitiger Layoutoptimierung immer schwerer kontrollierbar bzw. durchsetzbar.

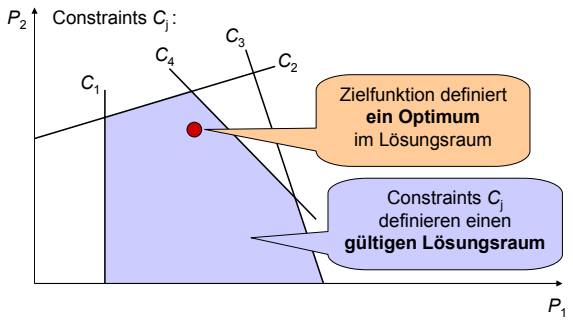

Bild 1 Die Freiheitsgrade der Constraints (Parameter P_i) spannen einen Lösungsraum auf. Während die Constraints damit die Grenzen der gültigen Layoutlösung festlegen, spezifiziert die Zielfunktion deren Qualität.

Für die Erstellung des Layouts von digitalen Schaltungen stehen derzeit verschiedene leistungsfähige Entwurfsprogramme zur Verfügung. Der automatische Layoutentwurf von analogen Schaltungen ist vergleichsweise wenig entwickelt. Dies ist auf die vielfältigen Arten von Constraints im analogen Layout, wie Übersprechen (Crosstalk), Matching, Symmetrie und viele weitere, zurückzuführen [2]. Daher wird der Layoutentwurf analoger Schaltungen fast ausschließlich per Hand durchgeführt, was ein Engpass insbesondere beim Systementwurf (SoC, System-on-Chip) mit analogen und digitalen Blöcken darstellt. Ein automatisierter Layoutentwurf ist hier nur möglich, wenn das angewendete Verfahren folgende Probleme berücksichtigt:

- Ist unter Einbeziehung aller Constraints feststellbar, ob eine gültige Lösung überhaupt existiert?
- Ist es möglich beim Vorhandensein mehrerer gültiger Lösungen, eine gute Lösung, die alle Constraints erfüllt, zeiteffizient zu finden?

1.1 Stand der Technik

In der Literatur werden verschiedene Verfahren für den Layoutentwurf von Analog- und Mixed-Signal-Schaltungen vorgestellt (z.B. [3]-[8]). Die meisten Verfahren dienen dem analogen Layoutentwurf auf Zellenebene [3]-[6]. Andere Ansätze betrachten Analog- und Mixed-Signal-Schaltungen auf Chipebene [7][8]. Hierbei werden Floorplanning, Platzierung und Verdrahtung auf Blockebene durchgeführt, d.h. es findet eine Optimierung der Blockanordnungen und Verbindungen hinsichtlich der Chipfläche statt. In beiden Fällen sind zahlreiche Arten von Constraints zu berücksichtigen.
Eines der ersten erfolgreichen analogen Layoutentwurf-Systeme wird in [4] beschrieben, wobei die Autoren in ihrem Ansatz Symmetrie sowie Device-Abutment und -Merging einbeziehen. Malvasi *et al.* stellen ein Verfahren für den sog. Performance-driven Layoutentwurf vor [5]. Die Autoren berechnen dabei die Sensitivitätsdaten und transformieren die Systemanforderungen der höchsten Entwurfsebene in Constraints auf Zellenebene. Die Layout-Generierung von analogen Schaltungen unter Beachtung der Her-

stellungsverfahren ist in [9] beschrieben. Hier werden die Layout-Tools direkt durch die Systemanforderungen der höchsten Entwurfsebene gesteuert.
Balasa *et al.* untersuchen die Platzierung analoger Zellen unter alleiniger Berücksichtigung der Symmetrie-Constraints [10]. In [11] wird ein Performance-driven Verfahren für die Platzierung analoger Schaltungen unter Berücksichtigung verschiedener parasitärer Effekte und Matching-Constraints vorgestellt. Tam *et al.* platzieren analoge Schaltungen unter Betrachtung der Symmetrie und anderer Constraints [12]. Dabei wird versucht, eine Lösung zu finden, die alle Constraints gleichzeitig erfüllt. Jedoch ist die ausschließliche Ermittlung einer solchen Lösung mit einem heuristischen Verfahren nicht immer möglich.
Die Anwendbarkeit der verfügbaren automatisierten Floorplanning- und Platzierungswerkzeuge beschränkt sich auf wenige Arten von analogen Schaltungen, wie z.B. Verstärker oder A/D Konverter. Außerdem sind sie nicht in der Lage, im Voraus zu bestimmen, ob eine gültige Lösung überhaupt existiert. Obwohl der Suchraum gültiger Layoutlösungen durch die Constraints aufgespannt ist, benutzen konventionelle Entwurfswerkzeuge gegebene Constraints nur in sehr beschränktem Maß zur Verifikation einer vorhandenen Designlösung. Die aktive Einbeziehung von Constraints in die Suchsteuerung während der Designoptimierung ist bisher nicht bekannt.

1.2 Neue Constraintberücksichtigung

Um die oben erwähnten Probleme zu überwinden, wurde ein neues Konzept für die Constraintberücksichtigung beim Layoutentwurf entwickelt und anhand eines Demonstrators für das Floorplanning von Analog- und Mixed-Signal-ICs umgesetzt.
Das Floorplanning-Konzept benutzt im ersten Schritt ein analytisches Verfahren (Linear Programming), um zum einen die Vereinbarkeit aller Constraints zu ermitteln. Damit wird zunächst untersucht, ob überhaupt eine gültige Lösung unter Constraintberücksichtigung möglich ist. Zum anderen wird hier ein Anfangsfloorplan erstellt, der optimal bezüglich der Constraints ist. Dabei erfüllt das erstellte Layout alle einzubeziehenden Constraints so, dass sie sich „mittig“ in ihren Wertebereichen, also maximal entfernt von ihren Randwerten, befinden (nachfolgend als „constraintgrenzenzentriert“ bezeichnet). Der erste Schritt findet daher eine Lösung mit maximaler Constraint-Variabilität, womit die nachfolgende Lösungsoptimierung bestmöglich vorbereitet ist. Ausgehend davon wird im zweiten Schritt durch ein heuristisches Verfahren (Simulated Annealing) eine Zielfunktion optimiert, die aus verschiedenen Optimierungszielen *und* allen gewichteten Constraints besteht. Durch die dabei adaptiv gewichteten Constraints wird die Layoutoptimierung per Simulated Annealing (SA) gesteuert, um gezielt nach der bestmöglichen Lösung zu suchen.
Während konventionelle Verfahren den durch die Constraints aufgespannten Lösungsraum mittels einer ausschließlich aus Optimierungszielen bestehenden Zielfunktion durchsuchen, lassen sich somit hier erstmals Constraints direkt in die Suchsteuerung einbeziehen.
Im nachfolgenden Abschnitt werden wichtige benutzte Begriffe erläutert. Abschnitt 3 stellt den Floorplanning-Ablauf detailliert vor. Anhand praktischer Beispiele werden die erzielten Ergebnisse im Abschnitt 4 diskutiert. Der Beitrag schließt mit einer Zusammenfassung.

2 Begriffsbestimmung

Aufgrund ihrer unterschiedlichen Berücksichtigung wird in diesem Beitrag zwischen Abstandsconstraints und Transformationsconstraints unterschieden.

- *Abstandsconstraints*: Technologische Mindestabstand-Constraints, welche garantieren, dass die einzelnen Layoutelemente einen vordefinierten Abstand einhalten.
- *Transformationsconstraints*: Constraints, die sich durch Transformationsmodelle aus den verschiedenen „konventionellen“ Constraints-Arten in geometrische Regeln abbilden lassen, werden als Transformationsconstraints bezeichnet. Diese beschreiben geometrische Randbedingungen zwischen zwei Punkten (z.B. den Zentren zweier Blöcke).

In diesem Beitrag wird vorausgesetzt, dass jedes einzelne konventionelle (und damit Transformations-) Constraint so definiert ist, dass es für mindestens einen Wert erfüllbar ist.
Verallgemeinert besitzt jedes Constraint einen Wertebereich, in dem es erfüllt ist. Der Wertebereich stellt den erlaubten Toleranzbereich dar und ist definiert zwischen einem minimalen (*min*) und einem maximalen (*max*) Wert. Ein Beispiel hierfür ist ein Constraint, das einen minimalen erlaubten Abstand zwischen einem Block und einer Wärmequelle definiert. *Min* definiert hier den vorgegebenen minimalen Abstand. *Max* ergibt sich z.B. aus den maximalen erlaubten Abmessungen des Layouts.
Da ein einziger Block oft verschiedene Constraints erfüllen soll, ist es generell nicht garantiert, dass eine Position alle Constraints optimal, d.h. constraintgrenzenzentriert, erfüllt. Sofern keine sich widersprechenden Constraints gegeben sind, ist es jedoch möglich, eine Position zu finden, die alle einzelnen Constraints so gut wie möglich erfüllt. Diese Position ist die optimale Position des Blocks unter Berücksichtigung aller zugehörigen Constraints. Unter Beachtung aller Blöcke wird ein optimaler Floorplan wie folgt definiert:

- *Optimaler Floorplan hinsichtlich der Constraints*: Jeder einzelne Block ist so angeordnet, dass alle zugehörigen Constraints so gut wie möglich erfüllt sind (d.h. so nah wie möglich zu den einzelnen optimalen Werten *opt*). Diese Anordnung besitzt maximale Constraint-Variabilität. Somit sollten geringfügige Ände-

rungen der Blockanordnung mit hoher Wahrscheinlichkeit keine Constraints verletzen.

3 Constraint-geführtes Floorplanning

Beim constraint-geführten Floorplanning erfolgt die Suche nach der bestmöglichen Floorplan-Lösung nicht durch eine ausschließliche Berücksichtigung von Optimierungszielen, sondern die Lösungssuche bezieht auch alle definierten Constraints mit ein.
Das Floorplanning-Verfahren ist in zwei Schritte aufgeteilt. Der erste Schritt benutzt ein analytisches Verfahren (Linear Programming), um einen Anfangsfloorplan zu ermitteln, der optimal bezüglich der Transformationsconstraints ist (Abschn. 3.2). Dieser erste Schritt findet daher eine Lösung mit maximaler Constraint-Variabilität. Optimierungsziele des Floorplannings werden in diesem Schritt nicht beachtet. Ausgehend davon wird im zweiten Schritt eine Zielfunktion mit einem heuristischen Verfahren (Simulated Annealing (SA)) optimiert. Die Optimierungsfunktion beinhaltet sowohl verschiedene Optimierungsziele, die Abstandsconstraints als auch alle Transformationsconstraints (Abschn. 3.3).

3.1 Ablauf des Floorplanning-Verfahrens

In **Bild 2** sind die wesentlichen Schritte des Floorplannings angegeben.

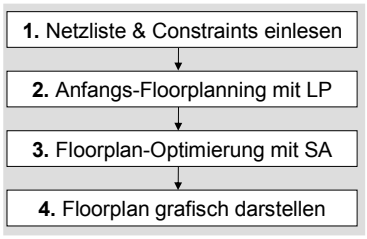

Bild 2 Abfolge der einzelnen Floorplanning-Schritte.

Schritt 2 beinhaltet das Anfangs-Floorplanning mittels Linear Programming (LP). Dabei wird unter Berücksichtigung aller Transformationsconstraints eine Anfangslösung erstellt. Hier werden lediglich die Abstandsconstraints ignoriert, d.h. das Layout zeichnet sich noch durch Überlappungen aus. Im Schritt 3 werden zur Floorplan-Optimierung mittels des SA-Algorithmus die adaptiven Constraint-Wichtungen einbezogen. Eine detaillierte Beschreibung der Schritte 2 und 3 erfolgt in den nachfolgenden Abschnitten 3.2 und 3.3.

3.2 Anfangs-Floorplanning mit Linear Programming (LP)

Beim Floorplanning mit Linear Programming [13] erfolgt die Darstellung des Floorplan-Problems in einem linearen (Un-)Gleichungssystem. Dabei werden die Transformationsconstraints als Ungleichungen beschrieben.
Die LP-Methodik besitzt folgende Vorteile:

- Es lässt sich frühzeitig feststellen, ob unter Beachtung aller Transformationsconstraints eine gültige Lösung überhaupt existiert.
- Es wird eine global-optimale Lösung bezüglich der Constraint-Einhaltung gefunden. Diese ist dadurch gekennzeichnet, dass alle Transformationsconstraints so eingehalten werden, dass sich die Lösung durch einen maximalen Abstand zu den Randwerten dieser Constraints auszeichnet. Diese Lösung wird als constraintgrenzenzentriert bezeichnet.

Eine LP-Formulierung eines Floorplans besteht aus den folgenden Komponenten:

- Zielfunktion,
- (Un-)Gleichungen für die Platzierung der Blöcke innerhalb der maximalen erlaubten Breite und Höhe eines Floorplans,
- Gleichungen für die Bestimmung der Abmessungen von Blöcken und vom gesamten Floorplan,
- (Un-)Gleichungen für die Abbildung der Transformationsconstraints.

Nachfolgend sind die verschiedenen LP-Komponenten beschrieben. Hier sind (x_i,y_i) die Blockkoordinaten, w_i und h_i die Breite bzw. die Höhe des Blockes i sowie X und Y die Floorplan-Abmessungen.

$$\text{Minimize } \sum D_k; \quad \forall k \in K; \tag{1}$$

$$x_i \geq 0; \quad y_i \geq 0 \tag{2}$$

$$x_i + w_i \leq X; \quad y_i + h_i \leq Y; \tag{3}$$

$$w_i = width(b_i); \quad h_i = height(b_i); \tag{4}$$

for each constraint c_k with $\forall k \in K$:

$$min_{ij}(c_k) = min\_allowed\_dist(b_i, b_j); \forall i \neq j, i, j \in n \tag{5}$$

$$max_{ij}(c_k) = max\_allowed\_dist(b_i, b_j); \forall i \neq j, i, j \in n \tag{6}$$

$$opt_{ij}(c_k) = optimal\_dist(b_i, b_j); \forall i \neq j, i, j \in n \tag{7}$$

$$|x_i - x_j| + |y_i - y_j| + \frac{(w_i - w_j) + (h_i - h_j)}{2} \geq \min_{ij}(c_k); \tag{8}$$

$$|x_i - x_j| + |y_i - y_j| + \frac{(w_i - w_j) + (h_i - h_j)}{2} \leq \max_{ij}(c_k); \tag{9}$$

$$|x_i - x_j| + |y_i - y_j| + \frac{(w_i - w_j) + (h_i - h_j)}{2} - opt_{ij}(c_k) = D_k; \tag{10}$$

Das Ziel ist die Minimierung aller D_k Variablen, wodurch eine optimale Blockanordnung (Platzierung) hinsichtlich der Constraints erreicht wird. D_k ist die Differenz zwischen der aktuellen und der optimalen Platzierung zweier Blöcke und soll minimiert werden, wobei K ist die Anzahl der Transformationsconstraints ist. Eine optimale Platzierung zweier Blöcke bedeutet, dass das Constraint zwischen ihnen optimal erfüllt ist.

Die Ungleichungen (2) und (3) dienen zur Platzierung der Blöcke innerhalb des Floorplans. In Gleichung (4) wird die Abmessung der Blöcke festgelegt. Die Gleichungen (5)-(7) bestimmen den minimalen, maximalen und „optimal“ erlaubten Abstand zwischen zwei Punkten. Die Gleichungen (8) und (9) garantieren, dass die Blöcke b_i und b_j so platziert werden, dass der Abstand in dem Bereich $[min_{ij}(c_k),max_{ij}(c_k)]$ liegt. Dieser Abstand wird hier durch den Manhattan-Abstand zweier Punkte berechnet (Zentrum-zu-Zentrum-Abstand zweier Blöcke). Die Gleichung (10) definiert die oben gegebenen Variablen D_k.

3.3 Floorplan-Optimierung mit Simulated Annealing (SA)

Der entwickelte SA-Algorithmus schließt in der Zielfunktion neben den Optimierungszielen auch alle Constraints (Abstands- und Transformationsconstraints) mit ein:

Zielfunktion = f (*Optimierungsziele* + *Constraints*)

Hierbei werden die Floorplan-Modifikationen immer akzeptiert, welche die Zielfunktion verbessern und alle Transformationsconstraints einhalten. Dagegen werden alle Modifikationen abgelehnt, die mindestens ein Transformationsconstraint verletzen. Wenn alle Transformationsconstraints erfüllt sind, wird die Zielfunktion nach jeder Modifikation anhand der Optimierungsziele und Constraint-Wichtungen berechnet.

Abstandsconstraints haben in der Zielfunktion eine vordefinierte hohe Wichtung, um Überlappungen zu vermeiden.

Die **Transformationsconstraints** in der Zielfunktion besitzen dagegen temperatur- und positionsabhängige Wichtungen. Während die Wertigkeit der *temperaturabhängigen* Wichtungen am Anfang denen der Abstandsconstraints entsprechen, verlieren sie im Laufe des Annealing-Prozesses an Gewicht. *Positionsabhängige* Wichtungen resultieren dagegen aus dem unterschiedlichen Erfüllungsgrad der einzelnen Transformationsconstraints und werden im Rahmen dieser Arbeit als „adaptive Constraint-Wichtung“ bezeichnet. Wie im nachfolgenden Abschnitt beschrieben, ermöglichen diese die effektive Erkundung des Lösungsraumes (Bild 3).

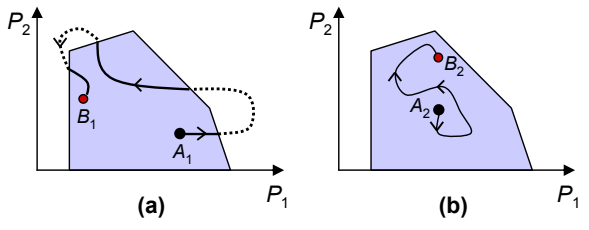

Bild 3 Beispiellauf eines konventionellen SA-Algorithmus (a), bei dem die Suchsteuerung ausschließlich durch die Zielfunktion erfolgt. Dagegen verdeutlicht (b) das Vorgehen des hier beschriebenen SA-Algorithmus mit Constraint-Führung, der sich durch eine Abfolge *gültiger* Lösungen und einer sinnvollen Lösungsraum-Durchsuchung auszeichnet.

3.3.1 Optimierung mit adaptiver Constraint-Wichtung

Die Transformationsconstraints in der Zielfunktion besitzen einen adaptiven Wichtungsfaktor, der aus einer temperaturabhängigen und einer positionsabhängigen Komponente besteht:

$$weight(c_k) = s(t) \times d_k$$

wobei $weight(c_k)$ der Wichtungsfaktor des Transformationsconstraints c_k ist, $s(t)$ ist die temperaturabhängige Komponente, d_k ist die positionsabhängige Komponente des Constraints c_k.

Die positionsabhängige Komponente der Wichtung ergibt sich aus der aktuellen Platzierung eines Blocks. Ist die Blockplatzierung hinsichtlich der Einhaltung seiner Transformationsconstraints nahe einem Optimum (d.h. sind die Transformationsconstraints so eingehalten, dass sie sich weit von ihren Randwerten entfernt befinden), sinkt die positionsabhängige Komponente des Wichtungsfaktors. Dagegen wird diese Komponente größer, wenn die Block-Position hinsichtlich der Transformationsconstraints nicht Constraintgrenzen-zentriert ist, d.h. die Blockposition befindet sich „nahe“ einer Constraint-Verletztung.

Die positionsabhängigen Komponenten werden mit der temperaturabhängigen Komponente zur Ermittlung der Wichtung der Transformationsconstraints multipliziert, um den Einfluss der Transformationsconstraints auf die Zielfunktion temperaturabhängig zu steuern. Aufgrund der temperaturabhängigen Komponente spielt die Wichtung der Constraints mit Temperaturabnahme eine immer kleinere Rolle bei der Bewertung der Zielfunktion (**Bild 4**).

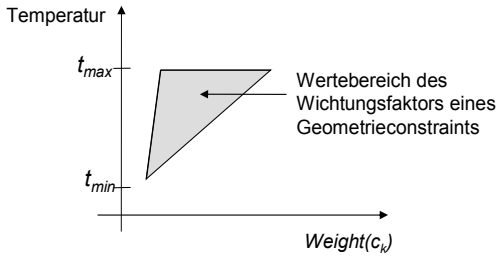

Bild 4 Veranschaulichung der Abnahme der Wichtung *weight*(c_k) eines Transformationsconstraints bei sinkender Temperatur.

Mit diesem Vorgehen erfolgt die Lösungssuche hinsichtlich der Optimierungsziele anfänglich unter Einhaltung einer starken Constraint-Variabilität, was bei der „initialen Groberkundung“ des Lösungsraums vorteilhaft ist. Während des Optimierungsprozesses verlieren die Constraints gegenüber den Optimierungszielen kontinuierlich an Bedeutung, da am Ende nur ein hinsichtlich dieser Ziele optimierter Floorplan von Interesse ist.

3.3.2 Ablauf der Floorplan-Optimierung

Die wesentlichen Schritte der Floorplan-Optimierung mittels adaptiver Constraint-Wichtung sind im **Algorithmus I** dargestellt. Der SA-Algorithmus startet mit

dem constraint-zentrierten Anfangsfloorplan, welcher mit LP ermittelt wurde. Die möglichen Floorplan-Modifikationen sind „verschieben“ und „rotieren“. Bei Verletzung mindestens eines Transformations-constraints wird die Modifikation nicht akzeptiert. Übersteigt die Anzahl diese „ungültigen Modifikationen“ anhand eines bestimmten Blocks einen temperaturabhängigen Maximalwert $k(t)$, dann ist dieser Block in seiner Position zu belassen und die Floorplan-Optimierung mit einem neuen Block fortzusetzen. Die sich ergebende Endlösung ist ein optimierter Floorplan hinsichtlich der Optimierungsziele, bestmöglicher Überlappungsvermeidung und einer garantierten Einhaltung der Transformations-constraints.

Algorithmus I: Floorplan-Optimierung

1: Read initial floorplan constructed using LP
2: REPEAT
3: REPEAT
4: Apply floorplan modification
5: IF no transformation constraint is violated THEN
6: Estimate adaptive weights
7: Evaluate cost function
8: Accept or reject the modification
9: ELSE
10: reject the modification
11: IF *count*++ > *k(t)* THEN
12: choose new module
13: UNTIL stop criterion
14: $t = \alpha * t$
15: UNTIL ($t < t_{min}$ or desired quality reached)
16: End.

4 Umsetzung und Ergebnisse

Der Floorplan-Ansatz wurde mit der Programmiersprache Java in Form eines Demonstrators umgesetzt. **Bild 5** zeigt die Benutzeroberfläche (GUI) des Floorplanners mit einem einfachen Beispiel. Dieses besteht aus 8 Chip-Pads und 16 Blöcken (b1-b16). Die Darstellungsfarben sind entweder grau (grün), wenn alle Transformationsconstraints constraintgrenzenzentriert erfüllt sind (z.B. Constraint c_1), hellgrau (gelb), wenn zugehörige Transformationsconstraints „gerade noch“ erfüllt sind, und schwarz (rot), wenn mindestens ein Transformationsconstraint des Blocks verletzt ist. Weiß dargestellte Blöcke besitzen keine zugehörigen Transformationsconstraints.
Der erste Schritt zur Erzeugung eines Anfangsfloorplans wird mit LP durchgeführt, wobei hier nur constraint-behaftete Blöcke relevant sind. LP platziert die Blöcke so, dass ihre Lagen bezüglich der Constraints zentriert sind (constraintgrenzenzentriert, d.h. möglichst „mittige“ Lage im jeweiligen Wertebereich der Constraints), wobei die Abstandsconstraints nicht berücksichtigt werden (**Bild 6 a**). In diesem Beispiel sind zwei Constraints, ein Top-Alignment-Constraint c_1 und ein Symmetrie-Constraint c_2, markiert:. Im nächsten Schritt wird der Floorplan mit SA und der adaptiven Constraint-Wichtung optimiert (**Bild 6 b**).

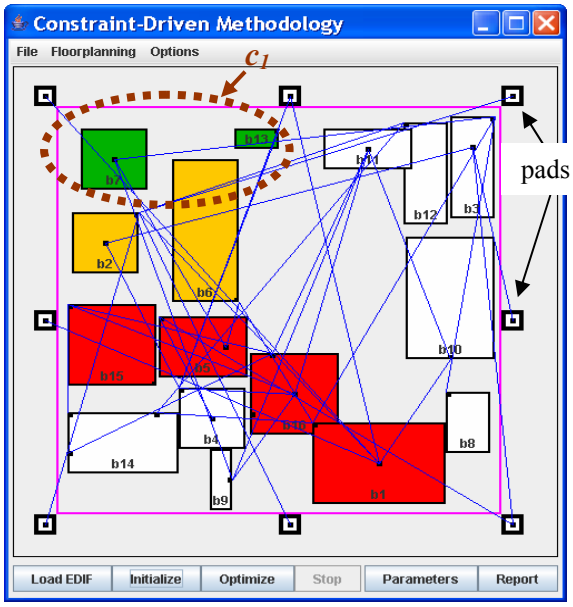

Bild 5 Floorplanning-Benutzeroberfläche mit einem noch nicht bearbeiteten Beispiel.

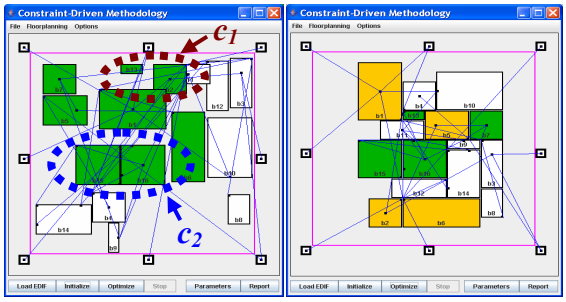

Bild 6 Der Anfangsfloorplan (links) und der optimierte Floorplan (Fläche, Verdrahtungslänge) unter Einhaltung aller Constraints (rechts) des Beispiels aus Bild 5.

Das hier vorgestellte Floorplan-Konzept ist an das kommerzielle IC-Designframework II von Cadence® angebunden. Als Eingangsdaten wird eine EDIF-Netzliste mit zusätzlichen Informationen über Blockabmessungen bzw. -fläche eingelesen. Zusätzlich werden die Constraints aus einer externen Datei importiert. Nach der Floorplan-Optimierung erfolgt eine Rückführung des dabei erzeugten Floorplans in die IC-Entwurfsumgebung. Die so erzeugte Floorplan-Anordnung lässt sich anschließend sowohl manuell als auch automatisch modifizieren oder kann direkt im weiteren Entwurfsfluss genutzt werden.
Der Floorplanner wurde innerhalb des industriellen Layoutflusses der Robert Bosch GmbH beim Floorplanning von Top-Level-Modulen analoger und Mixed-Signal-Schaltungen erfolgreich getestet. **Tabelle 1** zeigt den Zeitaufwand einiger praxisrelevanter Beispiele. Diese verdeutlichen den Effizienzgewinn gegenüber der bisher üblichen manueller Optimierung, die sich aufgrund der Vielzahl oft widersprechender Constraints über Stunden erstreckte.

Die geringen Rechenzeiten der initialen Linear Programming Optimierung ermöglichen darüber hinaus eine zeiteffiziente Constraint-Verifikation (existiert eine Lösung bei Berücksichtigung aller Constraints?) *vor* der eigentlichen Floorplan-Optimierung.

Tabelle 1 Zeitverhalten des Floorplanners (AMD Athlon 1.9 GHz, 512 MB RAM)

Circuit Name	Num. of Modules	Num. of Nets	Runtime (Sec.)		
			LP	SA	Total
CA01	32	620	0.1	15	15.1
CA02	62	1106	0.1	83	83.1
CB03	92	1398	0.2	196	196.2
CB04	101	1464	0.2	287	287.2
CB05	137	1781	0.2	405	405.2

Bild 7 veranschaulicht das sich ergebende Layout der Schaltung *CB05* unter Berücksichtigung von Verdrahtungsabständen.

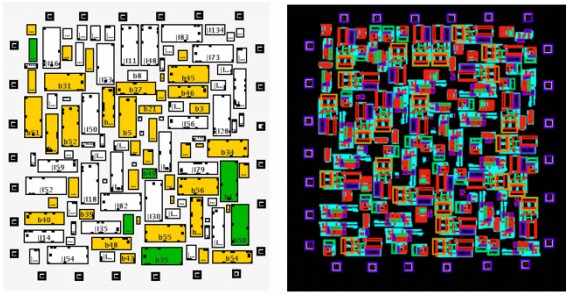

Bild 7 Ergebnis des Floorplannings der Schaltung *CB05* vor (links) und nach Überführung in die Cadence-Umgebung (rechts).

5 Zusammenfassung

Es wurde eine neue constraint-geführte IC-Entwurfsmethodik vorgestellt. Am Beispiel der Floorplan-Generierung ließ sich die prinzipielle Realisierbarkeit dieser Vorgehensweise bestätigen. Wesentliche Merkmale dieser Methodik sind eine zeiteffiziente Constraint-Verifikation und eine neuartige Einbeziehung der Constraints in die Lösungsoptimierung. Dabei ist es erstmals möglich, die zahlreichen und verknüpften Constraints vor dem zeitaufwendigen Optimierungsprozess zu verifizieren. Die Einbeziehung der Constraints in die Lösungsoptimierung erlaubt darüber hinaus eine effektive Suche im Lösungsraum.

Erste Anwendungen des vorgestellten Floorplan-Konzeptes in der IC-Entwurfsumgebung der Robert Bosch GmbH bestätigen die prinzipielle Nutzbarkeit unter industriellen Entwurfsbedingungen sowie eine beeindruckende Zeiteinsparung gegenüber dem manuellen Top-Level-Floorplanning realer Analog- und Mixed-Signal-Schaltungen.

Das hier vorgestellte Konzept soll Analog- und Mixed-Signal-Layoutdesignern helfen, Layoutprobleme mit zukünftig immer zahlreicheren, oft auch miteinander verknüpften Constraints, einer automatisierten Lösung zuzuführen.

Danksagung

Die vorliegende Arbeit wurde in Teilen gefördert vom BMBF-Verbundvorhaben LEONIDAS+ (Förderkennzeichen: 01M3074). Wir danken Dr. Jürgen Scheible und Ralf-Eckard Stephan (Robert Bosch GmbH) für die wertvollen Diskussionen.

Literatur

[1] J. Lienig: *Layoutsynthese elektronischer Schaltungen – Grundlegende Algorithmen für die Entwurfsautomatisierung*, Springer Verlag, 2006.

[2] G. Gielen and R. A. Rutenbar: Computer-Aided Design of Analog and Mixed-Signal Integrated Circuits, *Proc. IEEE*, vol. 88, No. 12, pp. 1825-1852, Dec. 2000.

[3] J. Rijmenants, J. B. Litsios, T. R. Schwarz, M. G. R. Degrauwe: ILAC: An Automated Layout Tool for Analog CMOS Circuits, *IEEE Journal of Solid--State Circuits*, vol. 24, No. 2, pp. 417-425, Apr. 1989.

[4] J.M. Cohn et al: KOAN/ANAGRAM II: New Tools for Device-Level Analog Placement and Routing, *IEEE Journal of Solid-State Circuits*, Vol. 26, No. 3, pp. 330-342, March 1991.

[5] E. Malavasi, E. Charbon, E. Felt, A. Sangiovanni-Vincentelli: Automation of IC Layout with Analog Constraints, *IEEE Trans. Computer-Aided Design*, Vol. 15, No. 8, pp. 923-942, Aug. 1996.

[6] J. M. Lin, G. M. Wu, Y. W. Chang, J. H. Chuang: Placement with Symmetry Constraints for Analog Layout Design Using TCG-S, *Proc. IEEE*, Vol. 2, pp.1135-1138, Jan. 2005.

[7] U. Choudhury, A. Sangiovann-Vincentelli: Constraint-based Channel Routing for Analog and Mixed Analog/Digital Circuits, *IEEE Trans. Computer-Aided Design*, Vol. 12, No. 4, pp. 497-510, Apr. 1993.

[8] S. Mitra, R. A. Rutenbar, L. R. Carley, D.J. Allstot: Substrateaware Mixed-signal Macrocell Placement in WRIGHT, *IEEE Journal of Solid-State Circuits*, Vol. 30, No. 3, pp. 269-278, March 1995.

[9] K. Lampaert, G. Gielen, and W. Sansen: *Analog Layout Generation Performance and Manufacturability*, Springer, 1999.

[10] F. Balasa, S. C. Maruvada, and K. Krishnamoorthy: On the Exploration of the Solution Space in Analog Placement with Symmetry Constraints. *IEEE Transactions Computer-Aided Design*, Vol. 23, No. 2, pp. 177-191, Feb. 2004.

[11] N. Jangkrajarng, L. Zhang, S. Bhattacharya, N. Kohagen and C.-J. Shi: Template-based Parasitic-Aware Optimization and Retargeting of Analog and RF Integrated Circuit Layouts. *Proc. of the 2006 ICCAD*, pp. 342-348, Nov. 2006.

[12] Y. Tam, E. F. Young, and C. Chu: Analog Placement with Symmetry and Other Placement Constraints. *Proc. of the 2006 ICCAD*, pp. 349-354, Nov. 2006.

[13] S. Sutanthavibul, E. Shragowitz, J. B. Rosen: An Analytical Approach to Floorplan Design and Optimization, *IEEE Trans. Computer-Aided Design*, Vol. 10, pp. 761-769, June 1991.