

# Propagierung und Transformation von Randbedingungen für den AMS-IC-Entwurf

Andreas Krinke\*, Maximilian Mittag†, Göran Jerke†, Jens Lienig\*

\*Technische Universität Dresden, Institut für Feinwerktechnik und Elektronik-Design, Dresden

†Robert Bosch GmbH, Reutlingen

E-Mail: {krinke, lienig}@ifte.de, {maximilian.mittag, goeran.jerke}@de.bosch.com

## Kurzfassung

Die Berücksichtigung einer immer größeren Anzahl von Randbedingungen stellt den Entwurf integrierter Analog- und Mixed-Signal-Schaltungen vor wachsende Probleme. Auch ist sie ein wesentliches Hindernis für die dringend benötigte, aber immer noch ausstehende weitere Automatisierung in diesem Bereich. Zur Lösung dieses Problems stellt der vorliegende Beitrag eine neuartige Methodik für die Propagierung und Transformation von Randbedingungen vor. Mittels dieses Verfahrens können sowohl Schaltungsentwickler als auch Softwarewerkzeuge sämtliche relevanten Randbedingungen jederzeit berücksichtigen, unabhängig davon, an welcher Stelle sie ursprünglich erzeugt wurden. Die Methode wurde in einen industriellen Entwurfsablauf integriert. Ihre praktische Relevanz konnte durch die erstmalige Realisierung einer elektrischen Randbedingung, die eine Begrenzung des Widerstandes  $R_{ON}$  von Endstufen in Smart-Power-ICs erlaubt, nachgewiesen werden.

## 1 Einleitung

Der Entwurf integrierter Analog- und Mixed-Signal-Schaltungen (AMS-ICs) ist von manuellen, nicht automatisierten Tätigkeiten geprägt. Dieser Umstand wird insbesondere im Vergleich mit der weitreichenden Automatisierung des Entwurfs digitaler Schaltungen deutlich. Die Diskrepanz ergibt sich im Wesentlichen durch die große Anzahl von Randbedingungen, die beim Analogentwurf zu berücksichtigen sind [9].

Randbedingungen beschreiben verbindliche Anforderungen an die Werte von Designparametern. Eine Anwendung ist die Beschreibung von Designzielen, wie z. B. die Weitergabe von Positionen und geplanten Flächen von Zellen nach dem Floorplanning. Ein zweites Einsatzgebiet ist die Festlegung von Parametergrenzen, z. B. eines maximalen Spannungsabfalls auf einer Leiterbahn, der für die korrekte Funktion notwendig ist. Lange Zeit waren analoge Entwurfsrandbedingungen implizit gegeben, entweder als sogenanntes „Expertenwissen“, oder im besten Fall als informelle Anmerkungen in Stromlaufplänen [5]. In den letzten Jahren wurde das Konzept von Randbedingungen für den Entwurf von AMS-ICs formalisiert und so zunehmend in Entwurfswerkzeuge integriert (z. B. [2, 10]).

Derartige Software unterstützt den hierarchischen Entwurf, bei dem der Schaltkreis in Zellen mit festgelegter Funktionalität aufgeteilt wird. Von jeder Zelle können mehrere Instanzen erzeugt werden. Während des Entwurfs lassen sich in der aktuell bearbeiteten Zelle Rand-

bedingungen definieren.

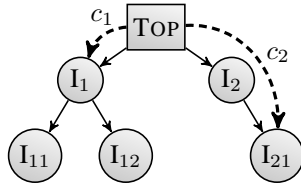
Der wesentliche Nachteil dieser Vorgehensweise ist die begrenzte Sicht- und Verifizierbarkeit der Randbedingungen. Zurzeit sind sie in der Regel nur in der originalen Zelle sichtbar und verifizierbar, also dort, wo sie erstellt wurden. Es gibt jedoch sogenannte hierarchische Randbedingungen (siehe folgender Abschnitt), die sich auf Elemente in anderen Zellen beziehen (z. B. auf Netze oder Instanzen) und die dadurch auch außerhalb der originalen Zelle relevant sind. Gleiches gilt für Randbedingungen, die nicht die Designhierarchie zur Informationsweitergabe verwenden, wie z. B. die im zweiten Absatz erwähnte verdrahtungsabhängige Beschränkung des Spannungsabfalls. In beiden Fällen fehlen sowohl Entwicklern als auch in Algorithmen Informationen darüber, ob lokale Designentscheidungen einen Einfluss auf Randbedingungen haben, die in anderen Zellen definiert wurden.

Die *Propagierung* löst dieses Problem durch Ausbreitung der Randbedingungen innerhalb des Designs. Als Erweiterung dieses Konzepts ermöglicht die *Transformation* eine Umwandlung von Randbedingungen, so dass zusätzliche Informationen abgeleitet werden können.

### 1.1 Stand der Technik

Zur Bewältigung der Komplexität moderner integrierter Schaltungen wird ein hierarchischer Entwurstil angewendet. Die resultierende *Designhierarchie* ist ein Graph, dessen Knoten den Zell-Instanzen entsprechen. Jede zugrundeliegende Zelle enthält: (a) Instanzen anderer Zellen, (b) Terminals, die zu Instanzen gehören und elektrische Verbindungen zu Elementen tiefer in der

Diese Arbeit wurde durch Mittel des Bundesministeriums für Bildung und Forschung (BMBF) unter der Fördernummer 01M3195B unterstützt.



**Bild 1** Die Randbedingung  $c_1$  ist lokal: Die referenzierte Instanz  $I_1$  gehört zur selben Zelle wie  $c_1$ . Die Randbedingung  $c_2$  ist hierarchisch: Sie gehört zur Zelle TOP und referenziert die Instanz  $I_{21}$ , die nicht zu TOP gehört. Veränderungen der Zelle von  $I_2$  können  $c_2$  beeinflussen.

Hierarchie erlauben, (c) Pins, die Verbindungen zu in der Hierarchie übergeordneten Elementen ermöglichen, und (d) Netze, die Terminals und Pins verbinden. Die Knoten des Graphen werden entsprechend den zugehörigen Instanzen durch Kanten verbunden.

Moderne Software-Werkzeuge für den Entwurf von AMS-ICs (z. B. das Cadence Design Framework II [2]) bieten ein System zur Verwaltung von Randbedingungen. Damit kann der Anwender neue Randbedingungen erzeugen, die zu einer bestimmten Zelle gehören. Sie beziehen sich auf eine Menge von Designelementen aus der obigen Aufzählung, den sogenannten *Zielelementen* (engl. *member*). Jedes dieser Elemente kann entweder lokal oder hierarchisch referenziert sein. Während lokale Elemente zur Zelle der Randbedingung gehören, befinden sich hierarchische Elemente in Zellen, die tiefer in der Designhierarchie liegen (**Bild 1**).

Weiterhin besitzt jede Randbedingung einen bestimmten Typ, der die Parameter aller abgeleiteten Randbedingungen sowie den Algorithmus zu ihrer Überprüfung (Verifikation) festlegt (siehe Abschnitt 2). Ein Beispiel ist der Randbedingungstyp zur Beschränkung des Widerstands von Leiterbahnen. Alle Randbedingungen dieses Typs besitzen je einen maximalen und minimalen Widerstandswert als Parameter und werden mittels eines Widerstandsmodells verifiziert.

Aktuelle Systeme zur Verwaltung von Randbedingungen, z. B. das in [2] integrierte, unterstützen nur netzbasierte Propagierung. Dabei existiert jedoch keine Möglichkeit zum Überspringen bestimmter, mit einem Netz verbundener Bauelemente. Weiterhin ist die Transformation von Randbedingungen nicht möglich. Heutige Verifikationswerkzeuge, wie z. B. Mentor Graphics' Calibre PERC, unterstützen topologische Randbedingungen. Geprüft wird die Existenz von Schaltungsstrukturen (z. B. ESD-Schutzschaltungen), die von Designregeln gefordert werden [7]. Dabei handelt es sich jedoch um einen Verifikationsschritt, der keine Erleichterung während des Entwurfsprozesses bietet.

## 1.2 Eigener Beitrag

In diesem Beitrag werden zwei Erweiterungen von Systemen zur Verwaltung von Randbedingungen vorgestellt: (1) Die *Propagierung* verteilt Randbedingungen innerhalb der Designhierarchie sowie basierend auf elektrischen Verbindungen und ermöglicht so den Zugriff auf

diese Information in allen relevanten Zellen.

(2) Die *Transformation* erzeugt automatisch neue Randbedingungen in Abhängigkeit vom Typ der originalen Randbedingung.

In Kombination erlauben beide Methoden die Sicht- und Verifizierbarkeit von Randbedingungen in allen Zellen, wo sie möglicherweise Einfluss auf Designentscheidungen haben. Dies erlaubt ihre kontinuierliche Berücksichtigung, unabhängig davon, wo sie in der Designhierarchie ursprünglich erzeugt wurden.

Ein neuer Typ von Randbedingung zur Beschränkung des Widerstands zwischen Gehäusepins dient zur Demonstration der praktischen Relevanz dieses neuartigen Ansatzes. Derartige Randbedingungen werden abhängig von der Verdrahtung zwischen den Pins transformiert und propagiert. Die experimentellen Ergebnisse für industrielle Designs zeigen eine Verringerung der Anzahl an Designiterationen, da ein bisher vor dem Tape-Out manuell durchgeführter Verifikationsschritt durch die Definition, Anzeige und Verifikation von Randbedingungen ersetzt werden konnte. Dadurch können bereits in frühen Stadien des Entwurfs getroffene Entscheidungen alle zu diesem Zeitpunkt existierenden Randbedingungen berücksichtigen.

## 2 Problembeschreibung

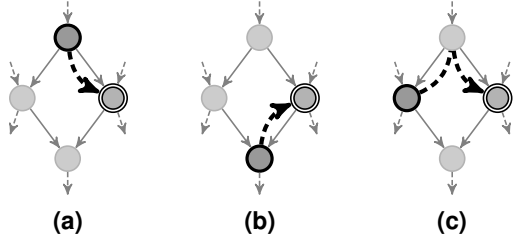
Eine Randbedingung  $c \in C(\text{cell})$  gehört zum Kontext einer bestimmten Zelle (*cell*), repräsentiert durch die Menge  $C$  aller Randbedingungen. Sie werden als Tupel  $c = (t(c), M(c), P(c))$  definiert, mit  $t(c) \in T$  als Typ der Randbedingung,  $M(c) = \{m_1, \dots, m_j\}$  als Menge der Zielelemente und  $P(c)$  als Tupel  $(p_1, \dots, p_k)$  der Werte aller Parameter von  $t(c)$ . Jedes Zielelement kann entweder *lokal* oder *hierarchisch* sein. Während lokale Elemente zur selben Zelle wie auch  $c$  gehören, befinden sich hierarchische Elemente in einer anderen Zelle tiefer in der Hierarchie (siehe Bild 1). Hierarchische Elemente können mittels eines Pfades  $i_1/i_2/\dots/i_n/e$  identifiziert werden. Dieser Pfad startet in der Zelle der Randbedingung, passiert Instanzen entsprechend der Sequenz  $i_1, i_2, \dots, i_n$  und erreicht schließlich das Designelement  $e$ , das ebenfalls eine Instanz sein kann. Eine Randbedingung ist *hierarchisch*, sobald sie mindestens ein hierarchisches Zielelement besitzt.

Für jeden Typ von Randbedingung existiert eine Funktion, die alle abgeleiteten Randbedingungen verifiziert. Diese Funktion legt implizit sämtliche Eigenschaften des Typs fest. Sie ist definiert als

$$\text{verif}_t : \{c \in C : t(c) = t\} \rightarrow \{\text{TRUE}, \text{FALSE}\}.$$

Für eine Randbedingung vom Typ  $t$  gibt  $\text{verif}_t$  nur dann TRUE zurück, falls die Randbedingung erfüllt ist.

Das Problem der Propagierung von Randbedingungen besteht in der garantierten Sicht- und Verifizierbarkeit hierarchischer Randbedingungen in *allen* Zellen, wo sie Designentscheidungen beeinflussen können.



**Bild 2** Die drei Arten der Propagierung von Randbedingungen. Top-Down-Propagierung (a), Bottom-Up-Propagierung (b) und Bottom-Up-Top-Down-Propagierung (c) [5]. Zellen mit hierarchischen Randbedingungen werden als  $\bullet$  dargestellt, die aktuelle Zelle ist mit  $\odot$  markiert.

### 3 Verwaltung von Randbedingungen

Während des Entwurfs eines integrierten Schaltkreises vorgenommene Änderungen an einer Zelle (z. B. das Einfügen eines Bauelements) sollen möglichst zu jedem Zeitpunkt sämtliche relevanten Randbedingungen (z. B. den maximalen Widerstand) einhalten. Erschwerend kommt hinzu, dass sich diese in all jenen Zellen befinden können, die eine hierarchische Verbindung zur aktuellen Zelle besitzen. Zwei Zellen sind hierarchisch verbunden, falls ein Pfad zwischen Instanzen der Zellen im Graph der Designhierarchie existiert (vgl. Bild 1). Die hier vorgestellte Methode propagiert diese Randbedingungen, indem in allen relevanten Zellen neue erzeugt und somit sicht- und verifizierbar werden.

Die Suche nach relevanten Randbedingungen in hierarchisch verbundenen Zellen erfordert Kenntnis davon, wo jede Zelle instanziiert ist. Diese Information wird entweder vom Entwurfswerkzeug bereitgestellt oder muss vor der Propagierung durch Verarbeitung der Instanzliste aller Zellen gesammelt werden. Die Verknüpfung jeder Zelle  $c$  mit einer Liste elterlicher Zellen, die Instanzen von  $c$  beinhalten, bildet als Datenstruktur  $H$  das Ergebnis dieses Prozesses.

#### 3.1 Top-Down-Propagierung

**Bild 2** zeigt neben der Top-Down-Propagierung, die hierarchische Randbedingungen in die Tiefe der Hierarchie verteilt, noch zwei weitere Arten der Propagierung. Die Bottom-Up-Propagierung unterstützt z. B. Randbedingungen für die Position von Zellen im Layout. Diese wird durch übergeordnete Zellen beeinflusst und somit müssen die Randbedingungen propagiert werden. Die Bottom-Up-Top-Down-Propagierung erlaubt schließlich die beliebige Ausbreitung von Randbedingungen in der Hierarchie, z. B. entlang der Verdrahtung.

Diese Methode realisiert die Propagierung in zwei Stufen: (a) Top-Down-Propagierung und (b) Allgemeine Propagierung. Ihre Kombination unterstützt u. a. alle drei beschriebenen Arten der Propagierung.

Für die Top-Down-Propagierung ist die Suche nach relevanten Randbedingungen auf in der Hierarchie übergeordnete Zellen beschränkt. Zum Beispiel werden in Bild 1 während der Propagierung zur Zelle von  $I_2$

```

1: procedure TOPDOWNPROPAGATION( $H, currentCell$ )
2:    $Visited \leftarrow \emptyset$ 
3:    $Queue \leftarrow H[currentCell]$ 
4:   while  $Queue \neq \emptyset$  do
5:      $parentalCell \leftarrow DEQUEUE(Queue)$ 
6:     if not  $Visited[parentalCell]$  then
7:        $Visited[parentalCell] \leftarrow TRUE$ 
8:       for all  $c \in C(parentalCell)$  do
9:         ▶ Prüfung, ob ein Zielelement von der aktuellen
10:        ▶ Zelle erreichbar ist.
11:        if  $\exists m \in M(c) : m \in currentCell$  then
12:          ▶ Erzeugung einer neuen Randbedingung.
13:           $c' \leftarrow MOVE(c, currentCell)$ 
14:           $C(currentCell) \leftarrow C(currentCell) \cup \{c'\}$ 
15:        end if
16:      end for
17:       $ENQUEUE(Queue, H[parentalCell])$ 
18:    end if
19:  end while
20: end procedure

```

**Bild 3** Die Top-Down-Propagierung von Randbedingungen zur aktuellen Zelle erfordert die Untersuchung aller übergeordneter Zellen. Es wird jede Randbedingung propagiert, die mindestens ein von der aktuellen Zelle erreichbares Zielelement besitzt. MOVE (Zeile 13) passt die Pfade der Zielelemente an und fügt neue Parameter hinzu.

die von TOP und  $I_2$  untersucht. Der in **Bild 3** gezeigte Algorithmus speichert Zellen, die relevante Randbedingungen enthalten könnten, zur späteren Untersuchung in einer Warteschlange (engl. Queue). Sobald ein hierarchisches Zielelement einer solchen Randbedingung von der aktuellen Zelle aus erreichbar ist, erfolgt die Propagierung in diese Zelle. Neu erstellte Randbedingungen haben die Form  $c' = (t', M', P')$ . Der abgeleitete Typ  $t'$  besitzt zusätzliche Parameter, die ein späteres Auffinden der originalen Randbedingung ermöglichen. Für  $M'$  werden die Pfade aller Zielelemente angepasst, um in der aktuellen Zelle zu beginnen. Von dort nicht erreichbare Elemente werden aus der Menge entfernt. Führt man diesen Algorithmus in der aktuell editierten Zelle aus, bewirken alle relevanten hierarchischen Randbedingungen in übergeordneten Zellen die Erzeugung neuer Randbedingungen in dieser Zelle, wodurch sie schließlich sichtbar werden.

#### 3.2 Allgemeine Propagierung

Zur Realisierung der Bottom-Up- und Bottom-Up-Top-Down-Propagierung wird eine neue Art von Funktion eingeführt. Diese ist spezifisch für einen Typ  $t$  von Randbedingung und ist definiert als:

$$propagate_t : \{c \in C : t(c) = t\} \rightarrow C'$$

mit  $C'$  als Menge aller möglichen Randbedingungen. Die Funktion  $propagate_t$  erhält eine Randbedingung  $c$  des Typs  $t$  als Argument und gibt eine einzelne propagierte Randbedingung  $c' = (t', M', P')$  zurück, wobei  $t'$  eine erweiterte Variante des originalen Typs  $t$  ist. Durch  $propagate_t$  können neue Randbedingungen in allen übergeordneten Zellen erzeugt werden. Dies

ermöglicht direkt die Bottom-Up-Propagierung. Durch eine anschließend ausgeführte Top-Down-Propagierung wird schließlich auch die Bottom-Up-Top-Down-Propagierung realisiert.

Für die Erstellung eines propagierbaren Typs von Randbedingung ist demnach die Implementation der entsprechenden  $\text{propagate}_t$ -Funktion zusätzlich zur Verifikationsfunktion  $\text{verif}_t$  (vgl. Abschnitt 2) notwendig. Zur Unterstützung aller drei Propagierungsarten wird der Algorithmus zur Top-Down-Propagierung aus Bild 3 erweitert. Die Suche nach relevanten Randbedingungen untersucht nun nicht nur übergeordnete Zellen, sondern alle, die über hierarchische Pfade erreichbar sind. Für jede Randbedingung  $c$  mit einer zugehörigen Funktion  $\text{propagate}_{t(c)}$  wird diese ausgeführt, was in einer propagierten Randbedingung  $c'$  resultiert. Der anschließende Top-Down-Propagierung erzeugt die eigentliche Randbedingung in der aktuell editierten Zelle.

### 3.3 Transformation

Die Transformation ermöglicht die Veränderung einer Menge von Randbedingungen hinsichtlich ihrer Typen, Zielelemente und Parameterwerte. Sie gestattet dadurch die Ableitung neuer Randbedingungen aus existierenden. Zum Beispiel ist bei Kenntnis des maximal zulässigen Spannungsabfalls und des maximalen Stroms einer Leiterbahn eine Randbedingung für ihren maximalen Widerstand herleitbar.

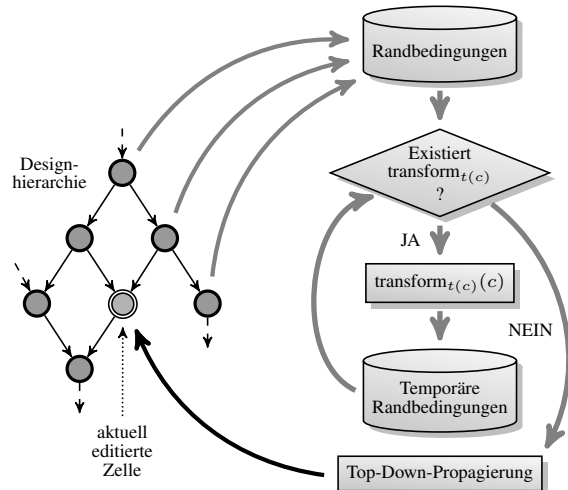
Zur Integration der Transformation in den Entwurfsablauf wird die im letzten Abschnitt beschriebene  $\text{propagate}_t$ -Funktion verallgemeinert. Dies führt zur Definition der Funktion  $\text{transform}_t$  für jeden transformierbaren Typ  $t$  von Randbedingung. Sie ist definiert als:

$$\text{transform}_t : \{c \in C : t(c) = t\} \rightarrow C'$$

mit  $C'$  als Menge aller möglichen Mengen von Randbedingungen. Im Unterschied zu  $\text{propagate}_t$  können hier mehrere Randbedingungen entstehen, die außerdem beliebigen Typs sein können. Sie kann die Funktion  $\text{propagate}_t$  ersetzen und für einen Typ  $t$  sowohl Propagierung als auch Transformation realisieren.

Es ist wichtig hervorzuheben, dass zwar die Kenntnis der Designhierarchie zum Auffinden aller relevanten Randbedingungen notwendig ist, ihre anschließende Transformation jedoch beliebige Informationen des Designs verwenden kann. Ein Beispiel ist die Transformation netzbasierter Randbedingungen unter Verwendung der Netzliste des Designs.

**Bild 4** zeigt den finalen Ablauf der Propagierung und Transformation. Der Prozess beginnt in der aktuell editierten Zelle mit der Untersuchung der Randbedingungen aller hierarchisch verbundenen Zellen. Falls keine Transformationsfunktion für den Typ  $t$  existiert, erfolgt nur die Top-Down-Propagierung. Andernfalls wird  $\text{transform}_t$  ausgeführt, wodurch eine oder mehrere neue Randbedingungen entstehen. Diese sind nur temporär und nicht sichtbar. Die Transformation wiederholt man



**Bild 4** Die entwickelte Methodik zur Propagierung und Transformation von Randbedingungen in eine Zelle kann einfach in den Entwurfsablauf integriert werden.

für temporäre Randbedingungen, bis nur noch eine Top-Down-Propagierung möglich ist. Erst danach wird jede beeinflussbare Randbedingung in der aktuellen Zelle sicht- und verifizierbar.

## 4 Experimentelle Ergebnisse

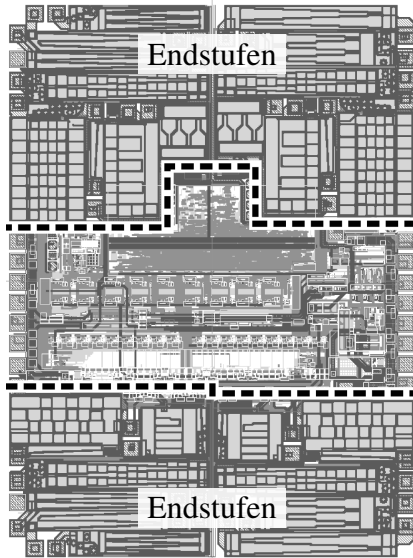
### 4.1 Implementation

Die entwickelte Methodik ist ein generisches Verfahren und somit unabhängig von der verwendeten Entwurfs-umgebung. Zur Demonstration ist sie als Erweiterung für das Cadence Design Framework II unter Verwendung der Skriptsprache SKILL++ [1] implementiert. Der in Bild 4 gezeigte Prozess wird durch das Öffnen einer Zelle aktiviert. Die Implementation ist in folgende Schritte aufgeteilt: (1) Bestimmung der Designhierarchie. (2) Ermittlung aller Randbedingungen in hierarchisch verbundenen Zellen. (3) Transformation oder Propagierung aller so gefundenen Randbedingungen wie beschrieben.

### 4.2 Komplexität

Die Laufzeitkomplexität dieses Ansatzes hängt von der Größe des Designs sowie der Anzahl und dem Typ der verwendeten Randbedingungen ab. Für ein Design mit  $n$  Zellen und  $m$  gegebenen Randbedingungen ist die Suche nach relevanten Randbedingungen  $\mathcal{O}(n + m)$ . Die Laufzeit der anschließenden Transformation und Propagierung ist von der Anzahl der Randbedingungen eines bestimmten Typs und der Komplexität der zugehörigen Transformationsfunktion abhängig. Weiterhin können diese Funktionen eine beliebige Anzahl temporärer Randbedingungen erzeugen, die u. U. wiederum transformierbar sind. Weil diese Eigenschaften durch das Design und die zur Verfügung stehenden Typen von Randbedingungen definiert werden, ist ihr Einfluss auf die Laufzeit nicht genauer charakterisierbar.

Es ist jedoch offensichtlich, dass die Implementation einer Transformationsfunktion hinsichtlich ihrer Laufzeit



**Bild 5** Layout eines industriellen Smart-Power-ICs mit einer großen Anzahl von Endstufen oberhalb und unterhalb der gestrichelten Linien. Jede Endstufe gehört zu einem einzelnen MOSFET im Stromlaufplan.

optimiert sein sollte. Zum Beispiel können notwendige Informationen zwischengespeichert werden, um langwierige Neuberechnungen zu vermeiden.

### 4.3 Demonstrator

Mit der entwickelten Methodik wurde ein neuer Typ von Randbedingung erstellt, der zur Festlegung eines maximalen Widerstands zwischen Gehäusepins eines Smart-Power-ICs dient. Derartige integrierte Schaltungen beinhalten einen oder mehrere große MOSFETs (Endstufen), deren Drain- und Source-Anschluss entweder direkt oder über Mess- und Schutzschaltungen mit Gehäusepins verbunden ist [8]. Diese Transistoren schalten große Leistungen und benötigen deshalb einen großen Teil der Chipfläche. Ein Beispiel für einen industriellen Smart-Power-IC mit einer großen Anzahl von Endstufen ist in **Bild 5** zu sehen. Die MOSFETs haben einen bedeutenden Einfluss auf die Qualität des gesamten Schaltkreises und werden deswegen mit beträchtlichem Aufwand optimiert. Ein Hauptziel ist ein möglichst geringer Widerstand zwischen Gehäusepins, die zu angeschalteten Endstufen gehören. Für diesen Widerstand  $R_{ON}$  wird eine obere Grenze  $R_{ON,max}$  spezifiziert, die direkt in eine Randbedingung überführt werden kann. Der neue Typ von Randbedingung soll direkte Rückkopplung bieten, wenn Veränderungen einer Zelle den Widerstand  $R_{ON}$  einer Endstufe beeinflussen. Die Aufgabe der zugehörigen Transformationsfunktion ist die Erzeugung von Randbedingungen für den Widerstand aller beteiligten Schaltungselemente. So werden z. B.  $R_{DS,ON,max}$ -Randbedingungen für alle Transistoren erzeugt. Die Grenzwerte berechnet man aus  $R_{ON,max}$  und den aktuellen Widerständen aller anderen Elemente zwischen den beiden Gehäusepins.

**Bild 6** zeigt eine Übersicht der Transformationsfunk-

```

1: procedure TRANSFORM $R_{ON,max}$ ( $c$ )
2:  $net_1, net_2 \leftarrow$  GetMembers( $c$ )
3:  $limit \leftarrow$  GetValue( $c$ )
4:  $netlist \leftarrow$  ExtractConnectivity( $c, net_1$ )
5:  $equations \leftarrow$  NodalAnalysis( $netlist$ )
6:  $constraints \leftarrow$  Solve( $equations, limit$ )
7: return  $constraints$ 
8: end procedure

```

**Bild 6** Übersicht über die konkrete Implementation der transform<sub>t</sub>-Funktion für den  $R_{ON,max}$ -Typs von Randbedingungen.

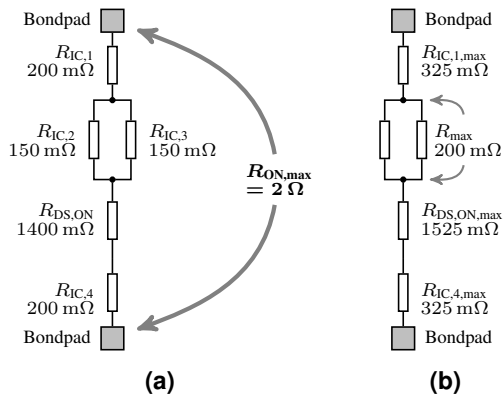
tion. Der erste Schritt ist die Extraktion der Netzliste, da die Transformation von den elektrischen Verbindungen zwischen den Gehäusepins abhängt. Dieser Prozess beginnt bei den Netzen der Pins und folgt ihnen durch die Hierarchie in andere Zellen. Bei Erreichen eines Bibliotheks- bzw. PDK-Elements (z. B. Transistoren und Widerstände), wird das Bauteil in der Netzliste gespeichert. Die Erkundung wird am zweiten Anschluss (bei Zweipolen) oder am Source- bzw. Drain-Anschluss (bei Transistoren) fortgesetzt. Im zweiten Schritt werden die aktuellen Widerstandswerte aller gefundenen PDK-Elemente ermittelt. Dazu werden entweder die Werte ihrer Widerstandsparameter oder – im Falle von Transistoren – ein Modell zur Berechnung von  $R_{DS,ON}$  anhand geometrischer Größen verwendet [6]. Bauelemente mit unbekanntem Widerstand werden ignoriert und später durch Kurzschlüsse ersetzt.

Zur Berechnung der Widerstandsbudgets wird eine Knotenspannungsanalyse durchgeführt und auf jedes Netz in der extrahierten Netzliste die Knotenregel angewendet. Die Ströme durch Bauelemente ergeben sich aus deren Widerstand und den Spannungen der angeschlossenen Netze. Das resultierende Gleichungssystem wird automatisch durch das quelloffene Computer-Algebra-System REDUCE [3, 4] gelöst. Das Ergebnis definiert den *größtmöglichen* Widerstand für jedes PDK-Element und bleibt gültig, solange die Werte aller anderen Elemente konstant bleiben. Danach generiert die Transformationsfunktion temporäre hierarchische  $R_{max}$ - und  $R_{DS,ON,max}$ -Randbedingungen für alle beteiligten Elemente. Sie befinden sich in der Zelle der originalen Randbedingung und werden nach der Top-Down-Propagierung in der aktuellen Zelle sichtbar und verifizierbar.

**Bild 7** zeigt ein vereinfachtes Beispiel einer extrahierten Netzliste. Die Endstufe wird durch  $R_{DS,ON}$  repräsentiert und ist über Verdrahtungselemente ( $R_{IC}$ ) mit Bondpads verbunden. Nach Transformation und Propagierung der  $R_{ON,max}$ -Randbedingung existieren Grenzwerte für alle relevanten Elemente. Sobald sich der Widerstandswert eines Bauelements ändert, werden diese Grenzwerte aktualisiert.

Die entwickelte Methodik und der neue Typ von Randbedingung wurde auf drei industrielle Entwürfe von Smart-Power-ICs mit 24, 27 bzw. 67 Endstufen angewendet.

**Tabelle 1** zeigt die Ergebnisse. Trotz der geringen An-



**Bild 7** Beispiel für eine Netzliste mit extrahierten Widerstandswerten und  $R_{ON,max}$ -Randbedingung (a). Transformation und Propagierung der Randbedingung erzeugt automatisch obere Schranken für die Widerstände aller beitragenden Elemente (b). Bei Veränderung eines Widerstands werden die Schranken aktualisiert.

zahl von Endstufen und  $R_{ON}$ -Randbedingungen beanspruchen die Zellen mit einem Einfluss auf diese Randbedingungen einen großen Teil der Chipfläche. Je nachdem, welche Zelle editiert wird, ist nur ein kleiner Teil aller temporären Randbedingungen sichtbar. Durch Anwendung dieser Lösung waren die Schaltungsentwickler in der Lage, lokale Modifikationen durchzuführen und gleichzeitig ihre globalen Auswirkungen im Blick zu haben.

## 5 Zusammenfassung

In diesem Beitrag wurde eine neue Methode zur Propagierung und Transformation von Randbedingungen vorgestellt. Unabhängig vom Ort der Definition einer Randbedingung erlauben diese Operationen ihre Sicht- und Verifizierbarkeit in allen Zellen, wo sie Auswirkungen auf Entwurfsentscheidungen hat. Weiterhin ermöglicht die Transformation eine Ableitung neuer Randbedingungen mit abweichendem Typ, wodurch sich der Lösungsraum des Entwurfs verkleinert.

Durch die entwickelte Methodik werden Typen von Randbedingungen unterstützt, deren Propagierung und Transformation von der Designhierarchie abhängt. Derartige Randbedingungen ergeben sich z. B. während des Floorplannings. Zusätzlich hat dieses Vorgehen für netzbasierte Randbedingungen Vorteile, wobei der Ansatz nach bisherigem Kenntnisstand als erster das Überspringen von Designelementen während der Ausbreitung entlang von Netzen ermöglicht. Der präsentierte Typ zur Beschränkung von  $R_{ON}$  demonstriert diese Fähigkeit. Darüber hinaus wird dieser Ansatz die Integration beliebiger projektspezifischer Randbedingungstypen erlauben.

Die Arbeit ist ein wichtiger Schritt zur weiteren Automatisierung des Analog- und Mixed-Signal-Entwurfs. Sie legt den Grundstein für eine automatische Berücksichtigung aller Randbedingungen, was eine wesentliche Voraussetzung für die langfristig angestrebte

**Tabelle 1** Vergleich von drei industriellen Designs in Bezug auf  $R_{ON,max}$ -Randbedingungen. Für derartige Randbedingungen relevante Zellen belegen einen großen Teil der Chipfläche.

Schaltung	design1	design2	design3
Anzahl der Zellen	278	335	184
Anz. der Endstufen = Anz. der $R_{ON}$ -Randbedingungen	24	27	67
Anzahl der Zellen mit Einfluss auf $R_{ON}$	32	23	28
Anteil aller relevanten Zellen an Gesamtfläche	42 %	58 %	67 %
Anz. temporärer Randbed.	78	93	208

Analog-Synthese ist.

Durch die entwickelte Methodik werden Entwürfe realisierbar, die zu jedem Zeitpunkt so korrekt wie möglich sind. Die Wiederverwendung von Zellen wird effizienter, da sämtliche Randbedingungen von Instanzen der Zelle bei ihrer Modifikation sichtbar sind.

Die demonstrierte Erzeugung komplett neuer, komplexer Typen von Randbedingungen verdeutlicht die praktische Relevanz. Erste Anwendungen im Rahmen des industriellen AMS-Entwurfs zeigen den Nutzen der entwickelten Methodik durch Vereinfachungen beim Verwalten von Randbedingungen sowie beim Wiederverwenden von Zellen. Auch lässt sich eine Verringerung an Designiterationen feststellen.

## 6 Literatur

- [1] Barnes, T. J.: *SKILL: A CAD System Extension Language*. In: *Proc. 27th Design Autom. Conf., DAC*, 1990, S. 266–271.
- [2] Cadence Design Systems, Inc.: *Speeding Design of Custom Silicon – The Virtuoso Custom Platform*, 2006. [http://www.cadence.com/rl/Resources/white\\_papers/Virtuoso\\_WP.pdf](http://www.cadence.com/rl/Resources/white_papers/Virtuoso_WP.pdf).
- [3] Hearn, A. C.: *REDUCE: A User-Oriented Interactive System for Algebraic Simplification*. In: Klerer, M. und J. Reinfelds (Hrsg.): *Interactive Systems for Experimental Applied Mathematics*, S. 79–90. Academic Press, New York, 1968.
- [4] Hearn, A. C.: *REDUCE User's Manual Version 3.8*, Feb. 2004. <http://www.reduce-algebra.com/docs/reduce.pdf>.
- [5] Jerke, G., J. Lienig und J. B. Freuer: *Constraint-Driven Design Methodology: A Path to Analog Design Automation*. In: Graeb, H. E. (Hrsg.): *Analog Layout Synthesis – A Survey of Topological Approaches*, S. 271–299. Springer, New York, 2011.
- [6] Kniffin, M. L., R. Thoma und J. Victory: *Physical Compact Modeling of Layout Dependent Metal Resistance in Integrated LDMOS Power Devices*. In: *Proc. 12th Int'l Symp. on Power Semiconductor Devices and ICs, ISPSD*, 2000, S. 173–176.
- [7] Marquardt, H., H. Wagieh, E. Weidner, K. Domanski und A. Ille: *Topology-Aware ESD Checking: A New Approach to ESD Protection*. In: *34th Electrical Overstress/Electrostatic Discharge Symposium*, 2012, S. 1–6.
- [8] Murari, B., F. Bertotti und G. A. Vignola (Hrsg.): *Smart Power ICs: Technologies and Applications*. Springer, Berlin, Heidelberg, 2. Aufl., 2002.
- [9] Rutenbar, R. A.: *Design Automation for Analog: The Next Generation of Tool Challenges*. In: *Proc. Int'l Conf. on CAD, ICCAD*, 2006, S. 458–460.
- [10] Subasic, P. und G. B. Arsintescu, Cadence Design Systems, Inc.: *Constraint data management for electronic design automation*, Patent US 7003749, 21.02.2004.