Ein formaler Ansatz zum Robustheitsnachweis

Görschwin Fey Rolf Drechsler

Fachbereich 3 – Mathematik und Informatik, Universität Bremen, 28359 Bremen
{fey,drechsle}@informatik.uni-bremen.de

Abstract

Es wird zunehmend wichtiger, dass integrierte Schaltkreise selbst beim Auftreten interner Fehlfunktionen noch ein korrektes Ein- /Ausgabeverhalten zeigen. Doch der Nachweis dieser Robustheit *gegenüber Fehlern wird bisher nicht ausreichend unterstützt.*

In dieser Arbeit wird ein Ansatz zum formalen Nachweis der Robustheit eines Schaltkreises vorgestellt. Dies wird durch eine Reduktion auf den sequentiellen Äquivalenzvergleich oder eine Reihe von Eigenschaftsbeweisen ermöglicht. Resultat der Prüfung ist eine Maßzahl für die Robustheit des Schaltkreises. Außerdem werden die Bereiche eines Schaltkreises identifiziert, die durch Architekturmaßnahmen nicht ausreichend gegen Fehler abgesichert sind und deshalb durch Fertigungstechniken gegen Fehlfunktionen abgesichert werden müssen.

1 Einleitung

Integrierte Schaltkreise werden zunehmend in sicherheitskritischen Anwendungen wie zum Beispiel dem „Stear-by-Wire" in Kraftfahrzeugen oder der Steuerung wichtiger Funktionen in Flugzeugen eingesetzt. Die Korrektheit solcher Schaltkreise wird durch den massiven Einsatz sowohl simulationsbasierter als auch formaler Verifikationsmethoden sichergestellt.

Gleichzeitig nimmt – gemäß Moore's Law – die Anzahl der Bauelemente, die in einem Gesamtsystem integriert sind, stetig zu. Währenddessen wird die physikalische Fläche, die ein einzelnes Bauelement einnimmt ständig geringer. Als Resultat werden selbst funktional korrekte Schaltkreise immer anfälliger gegenüber Fehlern, die erst nach der Produktion – also im praktischen Einsatz – auftreten. Hierzu zählen zum Beispiel transiente Fehler, die durch elektromagnetische Umweltstrahlung auftreten können und sich als *Single Event Upsets* (SEUs) äußern können, oder statische Fehler, die durch Elektromigration im Rahmen von Alterungsprozessen hervorgerufen werden. Schon während des Entwurfes wird deshalb durch eine entsprechende Schaltkreisarchitektur Sorge getragen, dass die Fehlfunktion einzelner Bauteile die funktionale Korrektheit des Schaltkreises nicht gefährdet. Stattdessen muss ein internes Fehlverhalten erkannt und signalisiert werden, während das Ein- /Ausgabeverhalten weiterhin wie im fehlerfreien Fall funktioniert. Ein einfaches Mittel hierfür ist zum Beispiel die redundante Auslegung funktionaler Einheiten.

Um nachzuweisen, dass ein Schaltkreis auch unter Fehlerannahmen noch die Spezifikation erfüllt, werden üblicherweise simulationsbasierte Methoden herangezogen. Diese sind jedoch vollkommen unzureichend in der Überdeckung aller möglichen Systemzustände sind. Es kann also Systemzustände geben, in denen interne Fehlfunktionen und resultierendes fehlerhaftes Verhalten unentdeckt bleiben.

Dagegen kann die Anwendung formaler Methoden den Nachweis liefern, dass in *jedem Systemzustand* und unter *allen Eingaben* garantiert *jede* interne Fehlfunktion erstens detektiert wird und zweitens kein Fehlverhalten verursacht. Erste Ansätze hierfür wurden in [8, 6] vorgestellt. Beide Methoden benutzen jedoch die Werkzeuge zur formalen Verifikation als „Black Box". Um die Robustheit eines Schaltkreises bezüglich eines Fehlermodelles zu prüfen, muss jeder einzelne mögliche Fehler durch einen sogenannten *Mutanten* in die Schaltungsbeschreibung injiziert werden. Der resultierende fehlerhafte Schaltkreis wird dann gegen den korrekten Schaltkreis verglichen. Es ist also eine explizite Enummerierung aller Fehler notwendig, was insbesondere für Mehrfachfehler unpraktikabel ist. Die jeweils überdeckte Fehlermenge ist durch die Funktionalität der verwendeten Mutanten limitiert. Während die Methode in [8] nur eine Ja/Nein-Antwort liefert, ob eine Schaltung bezüglich eines bestimmten Fehlers robust ist, wird in [6] der prozentuale Anteil „robuster Systemzustände" gezählt. Diese Antworten sind allerdings wenig hilfreich bei der Identifikation von Schaltungsbereichen, die entweder durch weitere Architekturmaßnahmen oder während der Fertigung robuster ausgelegt werden müssen.

In der vorliegenden Arbeit wird ein formaler Ansatz vorgestellt, der alle möglichen Fehler bezüglich dreier Fehlermodelle implizit betrachtet. Als Resultat werden die Teile der Schaltung berechnet, deren Fehlertoleranz verbessert werden muss. Zu jeder dieser Stellen kann explizit ein Fehler zusammen mit den Stimuli, die das Fehlverhalten sichtbar machen, angegeben werden. Weiterhin wird die Robustheit eines Schaltkreises bezüglich eines Fehlermodells definiert. Bei Erreichen von 100% Robustheit ist der Schaltkreis robust gegenüber allen modellierten Fehlern. Die Berechnung der Robustheit wird zunächst durch Reduktion auf den sequentiellen Äquivalenzvergleich theoretisch ermöglicht. Anschließend wird der Ansatz zur impliziten Fehlermodellierung auf Ebene der formalen Methoden beschrieben. Zur Manipulation auf Boolescher Ebene wird ein Erfüllbarkeitsbeweiser (SAT-Beweiser) eingesetzt. Schließlich werden verschiedene Techniken vorgeschlagen, die eine weitere Steigerung der Effizienz der Berechnung ermöglichen. Die praktische Anwendbarkeit wird durch experimentelle Ergebnisse nachgewiesen.

Diese Arbeit ist wie folgt strukturiert: Die Grundlagen werden in Kürze im folgenden Abschnitt erläutert. Dann wird der Begriff der Robustheit zusammen mit den zugehörigen Fehlermodellen in Abschnitt 3 eingeführt. In Abschnitt 4 wird der Ansatz zur Berechnung der Robustheit erläutert. Die Reduktion auf eine Reihe von Eigenschaftsbeweisen wird in Abschnitt 5 erklärt, außerdem werden Techniken diskutiert, die eine weitere Steigerung der Effizienz ermöglichen. Erste experimentelle Ergebnisse werden in Abschnitt 6 präsentiert. Schließlich werden die vorgestellten Forschungsergebnisse im letzten Abschnitt zusammengefasst.

2 Grundlagen

Im Folgenden werden Schaltkreise betrachtet. Ein Schaltkreis $\mathcal{C}$ besteht aus einer Menge von Komponenten. Hierzu zählen primäre Eingänge, primäre Ausgänge, Zustandsspeicher und interne kombinatorische Komponenten $g \in \mathcal{C}$. Jeder internen Komponente wird eine Boolesche Funktion zugeordnet, die Struktur des Schaltkreises ist durch einen Graphen definiert. Insbesondere sind hierdurch die Vorgänger und Nachfolger einer Komponente eindeutig gegeben.

Die Größe eines Schaltkreises ist durch die Anzahl der Komponenten gegeben, d.h. durch $|\mathcal{C}|$. Ein Teil eines Schaltkreises ist eine Teilmenge $\mathcal{S} \subseteq \mathcal{C}$ der Komponenten, deren Größe durch $|\mathcal{S}|$ gegeben ist.

Durch Struktur und Funktion wird das Ein- /Ausgabeverhalten des Schaltkreises bestimmt. Ausgehend von einem festen Startzustand, der durch eine Reset-Sequenz erreicht wird, führt eine bestimmte Eingabesequenz zu einem bestimmten Ausgabeverhalten.

Zur Manipulation Boolescher Funktionen existieren verschiedene Techniken, hierzu zählen zum Beispiel binäre Entscheidungsdiagramme [2] oder Beweiser für das *Boolesche Erfüllbarkeitsproblem* (SAT) [4, 5]. In dieser Arbeit werden SAT-Beweiser verwendet. Die Transformation eines beliebigen Schaltkreises in eine SAT-Instanz ist in linearer Laufzeit und mit linearem Speicherbedarf in Abhängigkeit von der Größe des Schaltkreises möglich [13, 7]. Die Entscheidung, ob eine gegebene SAT-Instanz erfüllbar ist, ist NP-vollständig [3]. Ungeachtet dessen können moderne SAT-Beweiser viele Probleminstanzen, die zum Beispiel aus der formalen Verifikation oder der Testmustergenerierung resultieren, sehr effektiv lösen [9, 10, 5].

3 Messung der Robustheit

In diesem Abschnitt werden zunächst Fehlermodelle angegeben und im Hinblick auf praktisch relevante Fehler motiviert. Im Anschluss wird bezüglich dieser Fehlermodelle eine formales Robustheitsmaß definiert.

3.1 Fehlermodelle

Bei Schaltkreisen treten verschiedene Typen von Fehlern auf, die die Funktion verändern. Diese Fehlertypen lassen sich zunächst in vorübergehende Fehler, zum Beispiel sogenannte SEUs durch Umgebungsstrahlung, und statische Fehler, zum Beispiel durch Elektromigration, untergliedern. Um die Robustheit eines Schaltkreises bezüglich dieser praktisch relevanten Fehlertypen zu differenzieren, werden im Folgenden verschiedene Fehlermodelle eingeführt.

Definition 1 *Gegeben sei ein Schaltkreis $\mathcal{C}$ und ein Teil $\mathcal{S} \subseteq \mathcal{C}$ dieses Schaltkreises.*

1. *Einen Fehler entsprechend des* nicht-deterministischen Fehlermodells $\mathcal{F}_N$ *zu injizieren, entspricht der Ersetzung der Ausgänge einer Komponente $g \in \mathcal{S}$ durch neue primäre Eingänge.*

2. *Einen Fehler entsprechend des* kombinatorisch deterministischen Fehlermodells $\mathcal{F}_C$ *zu injizieren, entspricht der Ersetzung einer Komponente $g \in \mathcal{S}$ durch einen neuen kombinatorischen Teilschaltkreis.*

3. *Einen Fehler entsprechend des* lokal deterministischen Fehlermodells $\mathcal{F}_L$ *zu injizieren, entspricht der Ersetzung einer Komponente $g \in \mathcal{S}$*

durch einen neuen kombinatorischen Teilschaltkreis, der die gleichen Vorgänger wie g hat.

Bemerkung 1 *Man beachte, dass die Reihe der Fehlermodelle $\mathcal{F}_N$, $\mathcal{F}_C$ und $\mathcal{F}_L$ eine zunehmende Anzahl von Bedingungen an die funktionale Veränderung des Schaltkreises darstellt. So kann zum Beispiel jede funktionale Änderung, die durch $\mathcal{F}_C$ verursacht wird, auch gemäß $\mathcal{F}_N$ erzeugt werden – aber nicht umgekehrt.*

Die Fehlermodelle entsprechen unterschiedlichen in der Praxis auftretenden Fehlertypen. SEUs können zum Beispiel durch nicht-deterministisches Verhalten gemäß $\mathcal{F}_N$ dargestellt werden.

Im Folgenden bezeichnet $\mathbb{C}_{\mathcal{C},\mathcal{S},\mathcal{F},N}$ die Menge aller Schaltkreise, die gemäß $\mathcal{F}$ aus $\mathcal{C}$ erzeugt werden können, indem N Fehler in den Teil $\mathcal{S} \subseteq \mathcal{C}$ des Schaltkreises injiziert werden.

3.2 Definition

Ein Schaltkreis wird als robust bezeichnet, wenn kein Fehler, das Ein- /Ausgabeverhalten des Schaltkreises verändert. Allerdings könnte zum Beispiel ein SEU, der an einem primären Ausgang auftritt, das Ausgabeverhalten unweigerlich verändern. Um dies zu verhindern, werden bestimmte Bereiche einer Schaltung während der Fertigung robust ausgelegt, indem zum Beispiel gröbere Strukturen verwendet werden. Diese Art der Robustheit lässt sich allerdings auf dem Entwurfsmodell nicht berechnen. Deshalb ist eine genauere Definition von Robustheit notwendig, die auch auf Teile eines Schaltkreises bezogen werden kann.

Weiterhin wird in einigen Fällen die Robustheit bezüglich einzelner Fehler nicht ausreichend sein, da oft selbst ein lokales Phänomen eine Fehlfunktion mehrerer Bauelemente verursacht. Deshalb wird hier der Begriff der Robustheit bezüglich Mehrfachfehlern definiert.

Beide Aspekte – die Betrachtung von Teilen eines Schaltkreises und Mehrfachfehler – werden durch die folgenden Definitionen abgedeckt.

Definition 2 *Gegeben sei ein Schaltkreis $\mathcal{C}$, ein Fehlermodell $\mathcal{F}$ und eine natürliche Zahl $N \geq 1$.*

Ein Teil $\mathcal{S} \subseteq \mathcal{C}$ von $\mathcal{C}$ heißt $(\mathcal{F}, N)$-robust *falls keine Injektion von N Fehlern in $\mathcal{S}$ gemäß $\mathcal{F}$ das Ein-/Ausgabeverhalten von $\mathcal{C}$ verändert.*

Auf dieser Basis lässt sich ein Maß für die Robustheit eines Schaltkreises $\mathcal{C}$ für N-fachfehler bezüglich eines Fehlermodells $\mathcal{F}$ angeben. Hierfür einfach den größten $(\mathcal{F}, N)$-robusten Teil S des Schaltkreises zu berechnen ist bei Mehrfachfehlern nicht ausreichend, da eventuell ein anderer Teil T nicht $(\mathcal{F}, N)$-robust ist, aber mit S gemeinsame Komponenten hat. Deshalb wird der größte Teil $\mathcal{S}$ von $\mathcal{C}$ bestimmt, aus dem keine Komponente in einem N-fachfehler vorkommt, der das Ein- /Ausgabeverhalten von $\mathcal{C}$ verändert.

Definition 3 *Gegeben sei ein Schaltkreis $\mathcal{C}$, ein Fehlermodell $\mathcal{F}$ und eine natürliche Zahl $N \geq 1$. Dann ist die* $(\mathcal{F}, N)$-Robustheit *von $\mathcal{C}$ gegeben durch $R_{\mathcal{F},N} = \frac{|\mathcal{S}|}{|\mathcal{C}|}$, wobei $\mathcal{S}$ eine maximale Teilmenge von $\mathcal{C}$ ist, so dass es kein $\mathcal{T} \subseteq \mathcal{C}$ gibt mit*

- $\mathcal{S} \cap \mathcal{T} \neq \emptyset$,
- $|\mathcal{T}| \leq N$,
- *$\mathcal{T}$ ist nicht $(\mathcal{F}, |\mathcal{T}|)$-robust und*
- *jede Teilmenge $\mathcal{T}' \subset \mathcal{T}$ ist $(\mathcal{F}, |\mathcal{T}'|)$-robust.*

Bemerkung 2 *Die Robustheit eines Schaltkreises bezüglich einer formalen Eigenschaft kann in analoger Weise definiert werden. Entsprechend kann auch der Algorithmus, der im nächsten Abschnitt vorgestellt wird, angewendet werden um die Robustheit bezüglich einer Eigenschaft zu bestimmen.*

4 Berechnung der Robustheit

4.1 Reduktion auf sequentielle Äquivalenz

Die Berechnung der Robustheit eines Schaltkreises kann in direkter Weise durch Abbildung auf den sequentiellen Äquivalenzvergleich erreicht werden.

Satz 1 *Gegeben sei ein Schaltkreis $\mathcal{C}$ und die Menge der fehlerhaften Schaltkreise $\mathbb{C}_{\mathcal{C},\mathcal{S},\mathcal{F},N}$. Der Teil $\mathcal{S}$ ist genau dann $(\mathcal{F}, N)$-robust, wenn jeder Schaltkreis $\mathcal{C}' \in \mathbb{C}$ zu $\mathcal{C}$ sequentiell äquivalent ist.*

Eine direkte Reduktion auf kombinatorische Äquivalenz ist trotz der direkten Abbildung der Zustandsbits im Allgemeinen nicht möglich, da ein Fehler die Menge der erreichbaren Zustände verändern kann ohne dabei das Ein- /Ausgabeverhalten zu verändern. Weiterhin ist die Anzahl der möglichen fehlerhaften Schaltkreise groß, so dass deren iterative Betrachtung sehr zeitaufwändig wäre. Deshalb wird im Folgenden ein Verfahren vorgestellt, das alle möglichen fehlerhaften Schaltkreise kompakt in einer Probleminstanz der Booleschen Erfüllbarkeit modelliert.

4.2 Implizite Enummerierung aller Fehler

Das vorgestellte Verfahren lehnt sich an Diagnoseansätze mittels Boolescher Erfüllbarkeit an [11, 12]. Während bei der Diagnose eine Modifikation des

```
1  function largestRobustPart(C, F, N)
2    create a copy C' of C;
3    foreach component g ∈ C' do
4      replace g by g'[g, f_g, F];
5    done;
6    constrain Σ f_g <= N;
7    for t = 1...t_max do
8      unroll C' and C for t cycles;
9      force at least one pair of POs
           to different values;
10     convert to SAT instance;
11     while (satisfiable) do
12       f_g = get_satisfying_assignment;
13       T := T ∪ g;
14       add constraint f_g == 0;
15     done;
16   done;
17   S := C \ T;
18   return S;
19 end function;
```

Abbildung 1. Pseudocode

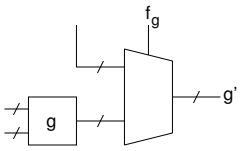

Abbildung 2. Änderung einer Komponente

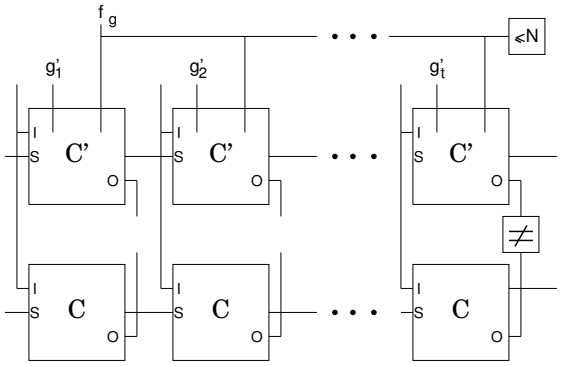

Abbildung 3. SAT Instanz

Schaltkreises gesucht wird, die zur Korrektur fehlerhaften Verhaltens führt, wird hier nach einer Modifikation gesucht, die zu fehlerhaftem Verhalten führt.

Das Verfahren wird zunächst anhand des Fehlermodells $\mathcal{F}_N$ beschrieben und im Anschluss auf die anderen Fehlermodelle erweitert. Die Erstellung der SAT-Instanz wird in Form eines Schaltkreises beschrieben, der anschließend in eine konjunktive Normalform überführt wird. Betrachtet wird weiter ein Schaltkreis $\mathcal{C}$.

Der Gesamtablauf ist in Abbildung 1 in Pseudocode dargestellt. Zunächst wird eine Kopie von $\mathcal{C}$ erzeugt (Zeile 2). Jeder Komponente $g \in \mathcal{C}'$ wird, wie in Abbildung 2 dargestellt, ein Fehlerprädikat f_g zugeordnet (Zeilen 3-4). Falls $f_g == 1$, so wird die Funktion von g modifiziert, sonst verhält g sich wie im fehlerfreien Fall. Zusätzlich wird die Anzahl der Fehlerprädikate mit dem Wert 1 auf maximal N beschränkt, um die Robustheit bezüglich N-fachfehlern zu bestimmen. Im nächsten Schritt wird nun $\mathcal{C}'$ einem sequentiellen Äquivalenzvergleich mit $\mathcal{C}$ unterzogen. Wie in Abschnitt 2 beschrieben, werden für den sequentiellen Äquivalenzvergleich beide Schaltkreise über t Zeittakte „abgerollt" (Zeile 8). Das Fehlerprädikat bleibt dabei für eine Komponente über alle Takte hinweg gleich. Für mindestens ein Paar primärer Ausgänge wird eine Abweichung der beiden Schaltkreise erzwungen (Zeile 9). Das Resultat ist in Abbildung 3 veranschaulicht. Die so erzeugte Probleminstanz ist nur dann erfüllbar, falls die Modifikation einer Komponente zu einer Abweichung im Verhalten der Schaltkreise führt. Kann keine Inäquivalenz gezeigt werden, wird die Anzahl der abgerollten Takte erhöht bis $t_{\max}$ erreicht ist. Um zu garantieren, dass tatsächlich alle Komponenten, deren Modifikation zu einer Verhaltensänderung führen kann, berechnet werden, muss $t_{\max}$ mindestens gleich der maximalen sequentiellen Tiefe eines Produktautomaten von $\mathcal{C}$ und $\hat{\mathcal{C}} \in \mathbb{C}_{\mathcal{C},\mathcal{S},\mathcal{F},N}$ sein.

Indem nun alle erfüllenden Belegungen berechnet werden (Zeilen 11-15), können alle Komponenten bestimmt werden, deren Modifikation gemäß $\mathcal{F}_N$ zu einer Änderung des Verhaltens von $\mathcal{C}$ führen kann. Diese Menge von Komponenten wird im Folgenden mit $\mathcal{T}$ bezeichnet. Die Komplementärmenge von $\mathcal{T}$ bezüglich $\mathcal{C}$ ergibt den größten $(\mathcal{F}_N, N)$-robusten Teil von $\mathcal{C}$ (Zeile 17).

Die Beschränkung des bisher dargestellten Algorithmus auf das Fehlermodell $\mathcal{F}_N$ resultiert aus der Veränderung einer Komponente g entsprechend Abbildung 2. Im Fehlerfall $f_g == 1$ kann g sich wie ein primärer Eingang nicht-deterministisch Verhalten. Die Einschränkung auf die Fehlermodelle $\mathcal{F}_C$ und $\mathcal{F}_L$ ist ebenso möglich. Hierzu werden für eine Komponente g zusätzliche Bedingungen hinzugefügt, die erzwingen, dass g sich deterministisch verhält.

Im Detail heißt das bezüglich des Fehlermodells $\mathcal{F}_C$: Ist die Belegung der Zustandsbits und der primären Eingänge zum Zeitpunkt t identisch mit der zum Zeitpunkt t', so muss der Ausgangswert von g zu beiden Zeitpunkten identisch sein.

Für das Fehlermodell $\mathcal{F}_L$ wird die Identität nur in Abhängigkeit von der Belegung der direkten Vorgänger von g gefordert.

Es resultiert der folgende Satz.

Satz 2 *Gegeben seien ein Schaltkreis $\mathcal{C}$, ein Fehlermodell $\mathcal{F}$ und eine positive Zahl N, weiterhin sei $\mathcal{S}$:=*largestRobustPart($\mathcal{C}$, $\mathcal{F}$, N). *Dann besitzt $\mathcal{C}$ eine Robustheit von $R_{\mathcal{F},N} = \frac{|\mathcal{S}|}{|\mathcal{C}|}$, falls $t_{\max}$ größer oder gleich der sequentiellen Tiefe des Produktautomaten aus $\mathcal{C}$ und $\mathcal{C}'$ ist.*

5 Diskussion

Der sequentielle Äquivalenzvergleich ist im Allgemeinen sehr resourcenintensiv. Deshalb werden in diesem Abschnitt Methoden diskutiert, um die Berechnung der Robustheit effizienter durchzuführen.

5.1 Reduktion auf Eigenschaftsbeweise

Die Funktionalität eines Schaltkreises kann genutzt werden, um die Komplexität der Berechnung deutlich zu verringern. Ein fehlertoleranter Schaltkreis hat oft zusätzliche Logik, die das Auftreten eines Fehlers signalisiert. Nach der Detektion des Fehlers kann dann durch externe Maßnahmen garantiert werden, dass der Schaltkreis wieder in einen fehlerfreien Grundzustand gebracht wird (zum Beispiel durch einen Neustart des Systems bei transienten Fehlern) oder ausgetauscht wird (bei permanenten Fehlern). Diese Funktionalität kann ausgenutzt werden, um das Problem des sequentiellen Äquivalenzvergleiches zu umgehen. Stattdessen wird ein induktiver Beweis durch mehrere formale Eigenschaften geführt. Jede Eigenschaft muss dabei nur über wenige Takte argumentieren. Als Grundlage dient dabei eine Invariante, die den fehlerfreien Systemzustand definiert. Die Robustheit der Fehlerdetektion wird dann mit dem gleichen Ansatz wie zuvor nachgewiesen. Einziger Nachteil des Verfahrens ist, dass der Ablauf nicht mehr vollautomatisch ist, da die Eigenschaften und insbesondere die Invariante von Hand für den jeweiligen Schaltkreis angepasst werden müssen.

Der induktive Beweis wird dabei wie folgt gegliedert:

1. **Voraussetzung:**

   Ausgehend vom Startzustand wird der Systemzustand im fehlerfreien Fall durch eine Invariante *Inv* erfasst.

2. **Schritt:**

   Es wird angenommen, dass bisher kein Fehler aufgetreten ist – die Invariante *Inv* ist also gültig. Dann wird unterschieden, ob im nächsten Schritt ein Fehler auftritt oder nicht.

   1. Fall: Es tritt kein Fehler auf.

      Eine Eigenschaft beweist, dass der Schaltkreis aus einem „fehlerfreien" Systemzustand in einen anderen „fehlerfreien" Zustand übergeht und dass die Logik zur Fehlerdetektion keinen Fehler signalisiert, d.h. der Nachweis der Invariante *Inv* wird geführt.

   2. Fall: Es tritt ein Fehler auf.

      Eine Eigenschaft beweist, dass der Übergang in einen Zustand, der im fehlerfreien Fall nicht erreichbar ist, durch die Logik zur Fehlerdetektion gemeldet wird, d.h. gilt *Inv* nicht mehr, so wird ein Fehler signalisiert.

Die Voraussetzung und der 1. Fall des Induktionsschrittes werden dabei mit einem herkömmlichen Eigenschaftsbeweiser überprüft. Lediglich für den 2. Fall des Induktionsschrittes ist die Verwendung des oben beschriebenen Modellierungsansatzes notwendig.

Der induktive Beweis umgeht das Problem der Erreichbarkeitsanalyse für fehlerhafte Schaltkreise. Es ist lediglich zu zeigen, dass der Übergang in einen – im fehlerfreien Fall – nicht erreichbaren Zustand detektiert wird. Die Anzahl der Takte, die hierfür betrachtet werden müssen, ist von der Funktionalität des Schaltkreises abhängig. Im einfachsten Fall wird jedes Durchlaufen von Zuständen, die im fehlerfreien Fall nicht erreichbar sind, sofort detektiert. In diesem Fall reicht es aus, einen Takt zu betrachten.

5.2 Steigerung der Effizienz

Der bisher vorgestellte Algorithmus ist vollständig, die Komplexität des sequentiellen Äquivalenzvergleiches oder des Eigenschaftsbeweises unter Fehlerannahmen ist jedoch sehr hoch. Deshalb werden im Folgenden Verbesserungen vorgestellt, durch die die Effizienz gesteigert werden kann.

Analog zum Vorgehen bei der automatischen Testmustergenerierung oder formalen Verifikation können verschiedene weitere Ansätze zur Problemlösung genutzt werden. So kann die Simulation zufälliger Testmuster helfen, um durch Fehlersimulation zu bestimmen, welche Komponenten eine Änderung verursachen können. Diese Komponenten müssen anschließend nicht weiter betrachtet werden, d.h. es müssen keine Fehlerprädikate f_g zugeordnet werden. Dadurch verkleinert sich der Suchraum. Außerdem ergibt sich so eine obere Grenze für die Robustheit der Schaltung.

Zusätzlich kann ein kombinatorischer Äquivalenzvergleich dazu dienen, die Komponenten zu bestimmen, die garantiert keine Verhaltensänderung verursachen können. Dazu wird an Stelle des sequentiel-

Tabelle 1. Laufzeiten bei Verwendung des sequentiellen Äquivalenzvergleiches für $N = 1$

		Gesamt			Fehlerhaft			
SK	$t_{\max}$	#Kmp	#FF	#Gt	$\lvert\mathcal{C}'\rvert$	SÄ	RSÄ	$R_{\mathcal{F}_N,N}$
s1269	5	624	74	1043	308	27,0s	88,4s	5%
r_s1269	5	1948	244	6514	970	14,8s	19185,3s	98%
rCounter	5	146	122	1505	70	0,2s	2,8s	97%
rCounter	10	146	122	1505	70	2,2s	21,7s	97%
rCounter	15	146	122	1505	70	13,3s	195,7s	97%

len Vergleiches ein kombinatorischer Vergleich durchgeführt. Komponenten, deren Modifikation in diesem Vergleich keine Änderung des Zustandes oder der Ausgabe bewirken, müssen im sequentiellen Fall nicht mehr betrachtet werden. Es resultiert eine – wenn auch grobe – untere Grenze für die Robustheit der Schaltung.

Eine weitere Verbesserung der Effizienz kann durch Ausnutzen der Struktur der Schaltung erreicht werden. Zunächst werden Fehler nur in Zustandsbits injiziert. Lediglich, wenn das Fehlverhalten eines Zustandsbits einen Ausgabefehler verursacht, muss die davor liegende kombinatorische Logik betrachtet werden. Diese Logik muss dann lediglich derart geändert werden, dass der betreffende fehlerhafte Zustand der Schaltung erreicht wird – die weitere Propagation muss nicht mehr betrachtet werden. In ähnlicher Weise kann, wie in [1] vorgeschlagen, die hierarchische Struktur der Schaltung genutzt werden, um zunächst die Modifikation von grob-granularen Modulen zu untersuchen und anschließend nur in den Modulen, die ein Fehlverhalten verursachen, die feinere Struktur zu betrachten.

Schließlich ist statt der exakten Berechnung des Robustheitsmaßes, die Berechnung einer oberen Schranke für $R_{\mathcal{F},N}$ möglich, indem $t_{\max}$ kleiner als die oben genannte maximale sequentielle Tiefe von $\mathcal{C}$ und $\hat{\mathcal{C}} \in \mathbb{C}_{\mathcal{C},\mathcal{S},\mathcal{F},N}$ gewählt wird. In der Praxis ist dies in den meisten Fällen auch sinnvoll.

6 Experimentelle Ergebnisse

Im Folgenden werden verschiedene robuste sowie nicht robuste Schaltkreise betrachtet. Alle Experimente werden bezüglich des Fehlermodells $\mathcal{F}_N$ durchgeführt. Die Laufzeiten wurden auf einem Rechner mit AMD Athlon 64 3500+ Prozessor, 1GB unter Linux gemessen.

Ergebnisse für die Abbildung auf den sequentiellen Äquivalenzvergleich werden in Tabelle 1 präsentiert. Es wird die $(\mathcal{F}_N, 1)$-Robustheit der Schaltkreise bestimmt. Dabei wurde $t_{\max}$ nicht analytisch bestimmt. Stattdessen wurde ein Wert gewählt, der zu einer akzeptablen Laufzeit führt. An einem Beispiel wird gezeigt, wie sich die Veränderung von $t_{\max}$ auf die Laufzeit auswirkt. In der Tabelle sind jeweils neben dem verwendeten Wert für $t_{\max}$, die Anzahl der Komponenten (#Kmp), der Zustandsbits (#FF) und der Gatter (#Gt) in der gesamten Probleminstanz angegeben. Weiterhin sind die Anzahl der Komponenten im Schaltkreis $\mathcal{C}'$ ($\lvert\mathcal{C}'\rvert$) und die Laufzeiten in CPU-Sekunden für den „normalen" sequentiellen Äquivalenzvergleich (SÄ) bzw. die Berechnung der Robustheit (RSÄ) ermittelt worden. Schließlich wird für jeden Schaltkreis die berechnete Robustheit ($R_{\mathcal{F}_N,N}$) angegeben.

Die Überprüfung der Robustheit des ISCAS-Schaltkreises *s1269* ergibt (wie erwartet) nur ein geringes Maß an Robustheit von 5%. Eine weitere Variante des Schaltkreises, die einen Mehrheitsentscheid dreier Instanzen als Ausgabe liefert, ist deutlich robuster. Nur Fehler, die direkt an primären Ein- und Ausgängen auftreten, führen zu fehlerhaftem Verhalten. Dadurch ergibt sich eine Robustheit von 98%.

Der Schaltkreis *rCounter* ist ein Zähler, dessen funktionale Einheiten redundant ausgelegt sind. Hierfür werden drei Zähler instantiiert. Durch einen Mehrheitsentscheid wird der Ausgabewert bestimmt. Weicht der aktuelle Wert eines der internen Zähler ab, so wird ein Fehler signalisiert. Somit kann eine Abweichung vom fehlerfreien Verhalten sofort festgestellt werden. Im Resultat ist der Schaltkreis $(\mathcal{F}_N, 1)$-robust. Lediglich Fehler, die an den primären Ausgängen auftreten, können nicht abgefangen werden. Hierdurch bleibt die Robustheit unter 100%.

Im Vergleich zu einem normalen sequentiellen Äquivalenzvergleich zeigt sich, dass die Berechnung der Robustheit deutlich aufwändiger ist, da sich durch die größere Anzahl primärer Eingänge der Suchraum vergrößert. Insbesondere zeigt das Beispiel *rCounter*, dass die Erhöhung des Wertes von $t_{\max}$ die Laufzeit signifikant verlängert. Insbesondere kann die maximale sequentielle Tiefe des Produktautomaten des fehlerfreien und eines fehlerhaften Schaltkreises nicht erreicht werden, um die Robustheit exakt zu berechnen. Hierzu sind Verbesserungen der Effizienz des Verfahrens notwendig.

Tabelle 2. Laufzeiten bei Verwendung des induktiven Ansatzes

SK	$\|\mathcal{C}\|$	#FF	#Gt	Vor.	Schritt 1. Fall	Schritt 2. Fall	$R_{\mathcal{F}_N,N}$
rCounter	79	25	370	<0,1s	<0,1s	0,2s	100%

Im Falle von *rCounter* ist dies durch eine Reduktion auf Eigenschaftsbeweise möglich, indem die Logik zur Signalisierung eines Fehlers ausgenutzt wird. Die experimentellen Ergebnisse dafür sind in Tabelle 2 dargestellt. Die Größe des Schaltkreises ist hier leicht erhöht (79 statt 70 Komponenten), da die Logik zur Fehlerdetektion nun ebenfalls betrachtet wird. Neben den oben schon genannten Zahlen sind die Laufzeiten für die drei Teilschritte des induktiven Beweises angegeben. Es musste jeweils nur ein Takt betrachtet werden, da jede Zustandsabweichung der drei internen Zähler detektiert wird. Im Resultat ergibt sich eine drastische Reduktion der benötigten Rechenzeit. Selbst der Gesamtaufwand zur Berechnung aller Teilschritte bleibt deutlich unter einer Sekunde. Bezüglich der Eigenschaft wird eine Robustheit von 100% erreicht, da 1-fachfehler angenommen werden. In diesem Fall ist entweder der Ausgabewert korrekt oder die Logik zur Fehlerdetektion funktioniert korrekt.

Insgesamt zeigt sich, dass die Robustheitsmessung durch implizite Enummerierung aller Fehler möglich ist. Eine deutlich Steigerung der Effizienz kann durch eine Reduktion auf Eigenschaftsbeweise erreicht werden.

7 Zusammenfassung

Es wurde ein Ansatz vorgestellt, um die Robustheit von Schaltkreisen automatisch zu berechnen. Ein vollautomatisches Verfahren wird durch die Reduktion auf sequentielle Äquivalenz ermöglicht. Der Rechenaufwand zur Berechnung der Robustheit ist in diesem Fall signifikant. Deshalb wurden Methoden zur Steigerung der Effizienz vorgeschlagen. Insbesondere lässt sich die Berechnung durch eine Reihe von Eigenschaftsbeweisen lösen. Das Verfahren hierzu ist nur halb-automatisch, da die entsprechenden Eigenschaften von Hand beschrieben werden müssen. Aber es lässt sich hierdurch eine drastische Verringerung der Rechenzeiten erreichen.

In der weiteren Arbeit sollen die Ansätze zur Steigerung der Effizienz des Verfahrens weiter ausgearbeitet, sowie die weiteren vorgeschlagenen Fehlermodelle intensiv untersucht werden.

Literatur

[1] M. Ali, S. Safarpour, A. Veneris, M. Abadir, and R. Drechsler. Post-verification debugging of hierarchical designs. In *Int'l Conf. on CAD*, pages 871–876, 2005.

[2] R. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Trans. on Comp.*, 35(8):677–691, 1986.

[3] S. Cook. The complexity of theorem proving procedures. In *3. ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[4] M. Davis, G. Logeman, and D. Loveland. A machine program for theorem proving. *Comm. of the ACM*, 5:394–397, 1962.

[5] N. Eén and N. Sörensson. An extensible SAT solver. In *SAT 2003*, volume 2919 of *LNCS*, pages 502–518, 2004.

[6] U. Krautz, M. Pflanz, C. Jacobi, H. W. Tast, K. Weber, and H. T. Vierhaus. Evaluating coverage of error detection logic for soft errors using formal methods. In *Design, Automation and Test in Europe*, pages 176–181, 2006.

[7] T. Larrabee. Test pattern generation using Boolean satisfiability. *IEEE Trans. on CAD*, 11:4–15, 1992.

[8] R. Leveugle. A new approach for early dependability evaluation based on formal property checking and controlled mutations. In *IEEE International On-Line Testing Symposium*, pages 260–265, 2005.

[9] J. Marques-Silva and K. Sakallah. Conflict analysis in search algorithms for propositional satisfiability. In *IEEE International Conference on Tools with Artificial Intelligence*, 1996.

[10] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an efficient SAT solver. In *Design Automation Conf.*, pages 530–535, 2001.

[11] A. Smith, A. Veneris, and A. Viglas. Design diagnosis using Boolean satisfiability. In *ASP Design Automation Conf.*, pages 218–223, 2004.

[12] S. Staber, G. Fey, R. Bloem, and R. Drechsler. Automatic fault localization for property checking. In *IBM Haifa Verification Conference*, volume 4383 of *LNCS*. Springer Verlag, 2006.

[13] G. Tseitin. On the complexity of derivation in propositional calculus. In *Studies in Constructive Mathematics and Mathematical Logic, Part 2*, pages 115–125, 1968. (Reprinted in: J. Siekmann, G. Wrightson (Ed.), Automation of Reasoning, Vol. 2, Springer, Berlin, 1983, pp. 466-483.).