

# Selbstreparatur von Logik-Baugruppen in hochintegrierten Schaltungen – Möglichkeiten und Grenzen

René Kothe<sup>1</sup>, Sven Habermann<sup>2</sup>, Heinrich T. Vierhaus<sup>1</sup>

<sup>1</sup> Lehrstuhl Technische Informatik

<sup>2</sup> Philips Semiconductors, Hamburg

## Kurzfassung

Aktuelle Planungen zur weiteren Entwicklung der Halbleiter-Technologie mit Nanostrukturen sagen steigende unvermeidbare Fehlerraten durch statische Schwankungen wichtiger Parameter voraus. Damit ist bei „Systems on a Chip“ mit Millionen von Transistoren eine wirtschaftliche Ausbeute auf der Basis vollständig funktionierender Chips zunehmend unwahrscheinlich. Die mit kleineren Strukturgrößen steigende spezifische Belastung durch hohe Stromdichten und hohe Feldstärken führt andererseits auch zu einer höheren Wahrscheinlichkeit zu Ausfällen von ICs im Zielsystem. Damit werden Techniken notwendig, welche SoCs nach der Fertigung und sogar „im Feld“ die Fähigkeit zur Selbstreparatur verleihen. Erste Ansätze dazu werden vorgestellt.

## Abstract

Recent publications predict that IC manufacturing technologies based on nano-scale devices will have an insufficient production yield due to fluctuations of device parameters. Furthermore, such devices are expected to have higher failure rates in the field of application. Essentially, such integrated circuits require a technology of self-repair, both for economical yield and for applicability in dependable electronic systems. While self-test and even self repair technologies are „state of the art“ semiconductor memory circuits, logic self repair is by far an open problem. First investigations on logic self repair are presented here.

## 1 Einführung

Seit etwa 2004 gibt es ernst zu nehmende Publikationen, welche eine zunehmende Dichte von Fehlern bei hochintegrierten Schaltungen vorhersagen, die mit minimalen Strukturgrößen weit unter 100 nm gefertigt werden [1-5]. Ursache sind nicht nur strukturelle Defekte wie Leitungsunterbrechungen und Kurzschlüsse durch Isolierschichten, sondern relativ zunehmende Auswirkung statistischer Streuungen wie z. B. die der Dotierungsdichten mit Auswirkung auf die Schwellenspannungen von Transistoren. Damit besteht die Notwendigkeit, hochintegrierte Systeme entweder trotz multipler Fehler noch betreiben und beherrschen zu können oder durch Verzicht auf entsprechend klei-

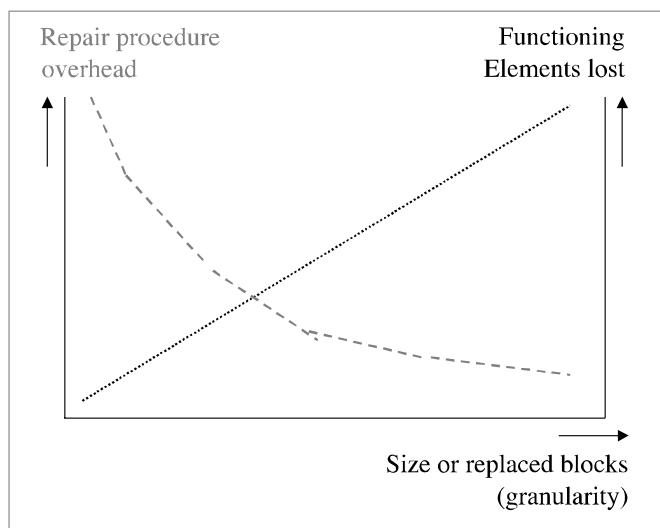
ne und kritische Bauelemente bestimmten Fehlermechanismen auszuweichen. Letztere Strategie bedeutet aber signifikante Nachteile bezüglich der Integrationsdichte und der maximalen Schaltfrequenzen. Kleinere Strukturgrößen haben zusätzlich ein höheres Risiko des vorzeitigen Ausfalls „im Feld“ [4], das durch gängige Mechanismen der Fehlererkennung und -korrektur wie der Dreifach-Auslegung von Baugruppen nur unzureichend und zu hohen Kosten zu beherrschen ist. In den Vordergrund des Interesses rücken deshalb Verfahren des eingebauten Selbsttests (BIST) und der Selbstreparatur (BISR) für hochintegrierte Schaltungen. Während solche Verfahren für „eingebettete“ Speicher-Baugruppen auf SoCs seit einigen Jahren bekannt sind und recht gut beherrscht werden [10, 11], ist die Selbstreparatur irregulärer Logik ein bisher weitgehend ungelöstes Problem. Die „Roadmap“ der Halbleiter-Industrie von 2005 [8] schätzt BIST und BISR für Speicher als „bekannt und benötigt“ an 2006 ein, dagegen wird BISR für Logik spätestens ab 2012 benötigt und als „unbekannt bezüglich praktisch anwendbarer Lösungen“ gekennzeichnet.

Damit ist die weitere Entwicklung zu Strukturgrößen weit unter 100 nm für Logik wahrscheinlich nur dann sinnvoll, wenn zuverlässig funktionierende ICs und SoCs auf der Basis nicht hochzuverlässiger Hardware aufgebaut werden können. Ansätze zu selbstreparierenden Systemen sind auf der Basis konfigurierbarer Logik-Baugruppen mit Feldprogrammierbaren Gate-Arrays (FPGAs) versucht worden [9]. Sie zeigen aber einige ganz signifikante Basis-Probleme auf. Andere Ansätze arbeiten mit funktionalen Makros, die in einer Struktur mehrfach in gleicher Form vorhanden sind [12]. Ein spezielles Problem stellen die Verbindungsstrukturen da [6].

## 2 Das Problem der Granularität

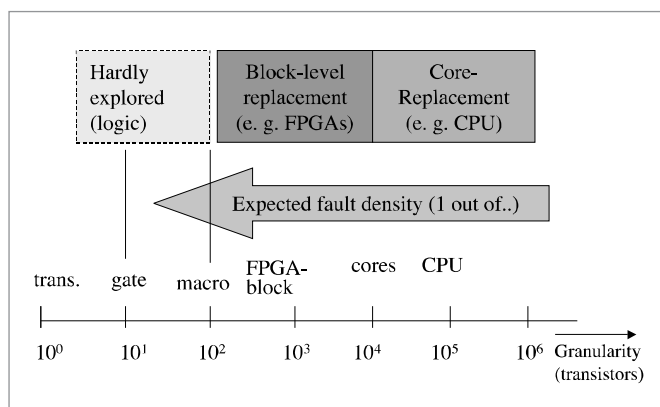
Mechanismen der Selbstreparatur Basis des Austauschs ausgefallener Funktionseinheiten auf der Basis des Ersatzes ausgefallener Makro-Komponenten durch redundant vorhandenen Bausteine sind seit Jahrzehnten z. B. aus der Großrechner-Technik bekannt. In neueren Publikationen werden auch kleinere Baugruppen wie z. B. Multiplizierer durch redundante Bausteine ersetzt [12, 13]. Typisch ist dabei, dass nur ein oder sehr wenige Typen sehr komplexer Basis-Baugruppen ersetzt werden. Für diesen Fall ist der notwendige Aufwand für die Diagnose des Fehlers in der Regel gering. Damit wird auch die eigentliche Funktion der „Selbstreparatur“ sehr einfach. Nachteilig ist, dass

z. B. für einen defekten Transistor in einem Rechner-Kern, der als defekt ermittelt worden ist, bis zu 1Mio. oder mehr eigentlich funktionsfähige Transistoren desselben Kerns ebenfalls abgeschaltet werden müssen. Bei regulären Baugruppen wie Multiplizierern oder den konfigurierbaren Kernen von FPGAs (configurable logic blocks – CLBs) liegt die Komplexität immer noch bei tausend Gattern oder mehr. Damit wird die mögliche „Vorratshaltung“ auf nur recht wenige Ersatzbaugruppen beschränkt, oder insgesamt ist nur eine geringe Anzahl von Fehlern tatsächlich reparierbar, weil die Effizienz der Reparaturfunktion unzureichend ist. Einer Reduzierung der „verschwendeten“ Bauteile bei der Reparatur durch den Übergang auf eine feinere Granularität der ausgetauschten Baugruppen steht ein wesentlich höherer Aufwand für die Reparaturfunktion gegenüber (Abb. 1).



**Abbildung 1:**  
Aufwand für Reparaturfunktionen und Effizienz der Reparatur

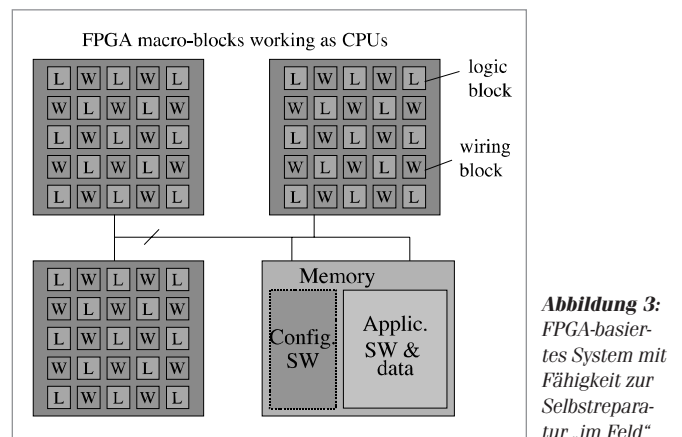
Den Stand der Technik gibt Abb. 2 wieder. Für relativ grobe Granularitäten sind Ersatzfunktionen entwickelt und erprobt worden. Sie reichen jedoch sicher nicht aus, um die für Nanometer-Technologien vorhergesagten Fehlerraten von bis zu z. B. 0,1 bis 1 Prozent, bezogen auf die Zahl der Transistoren, beherrschen zu können. Deshalb sind hier grundlegend neue Ansätze und Methoden notwendig.



**Abbildung 2:**  
Granularitäten der Selbstreparatur von Logik

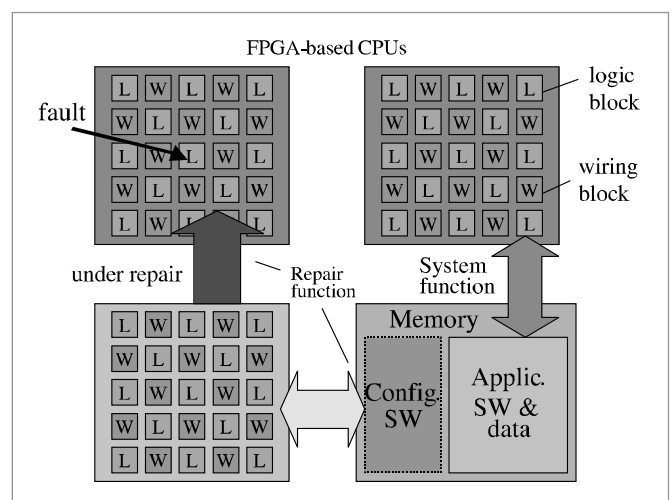
### 3 Reparaturfunktionen auf der Basis von FPGAs

Für den Fertigungstest von Schaltungen und Systemen sind potenziell externe Ressourcen verfügbar, die z. B. eine Fehlerdiagnose ganz wesentlich unterstützen können. Wegen der hohen Kosten von IC-Testern gibt es aber schon hier eine eindeutige Tendenz dazu, die Testfunktion auf den zu testenden Chip zu verlagern, so dass ein externer Tester nur noch Koordinationsfunktionen zu leisten hat, die selbst nicht Echtzeit-kritisch sind. Soll dagegen die Reparatur-Funktionen unabhängig im Zielsystem erfolgen, so müssen alle notwendigen Ressourcen „on-chip“ oder „on-board“ bereit-gestellt werden.



**Abbildung 3:**  
FPGA-basiertes System mit Fähigkeit zur Selbstreparatur „im Feld“

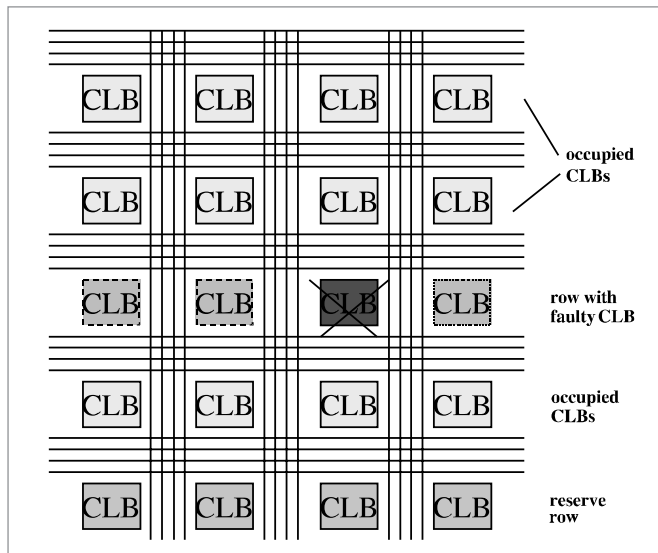
Ein zur Selbstreparatur „im Feld“, im speziellen Fall der Weltraum, fähiges FPGA-basiertes System wurde 2004 von einer Arbeitsgruppe der Stanford-Universität vorgestellt [9]. Voraussetzung ist die Verfügbarkeit von mindestens 3 unabhängigen FPGA-Blöcken, die jeweils Prozessor-Funktionen übernehmen können. Dann ist für die Fähigkeit der Selbstreparatur „im Feld“ unter Aufrechterhaltung der notwendigen Systemfunktion eine Dreifach-Auslegung notwendig. Während ein Prozessor die Systemfunktion übernimmt, wird ein zweiter Block als Prozessor benötigt, welcher die Reparaturfunktion übernimmt, die auf dem dritten (defekten) Block ausgeführt wird (Abb. 4).



**Abbildung 4:**  
Reparaturfunktion bei Dreifach-Auslegung

Im System der Stanford-Universität wird nur ein recht einfacher Prozessor-Typ (Mikrocontroller Intel 8051) mittels des FPGAs nachgebildet. Dadurch bedingt verfügt das System nur über eine relativ geringe Rechenleistung für die Aufgabe der Rekonfiguration, die wiederum keine sehr effiziente Ersatzstrategie erlaubt. Die für eine (Re-)Programmierung zugreifbaren Partitionen eines FPGAs sind in der Regel die konfigurierbaren Logik-Blöcke (CLBs), welche auch lokale Verdrahtungen umfassen. Innerhalb der CLBs existieren als reguläre Sub-Module sogenannte „Slices“ [14], welche aber für die logische Konfigurierung nicht zugreifbar sind.

Es ist möglich, gezielt bestimmte CLBs von der Verwendung innerhalb einer initialen Programmierung auszuschließen, um gezielte Redundanzen zu erzeugen.

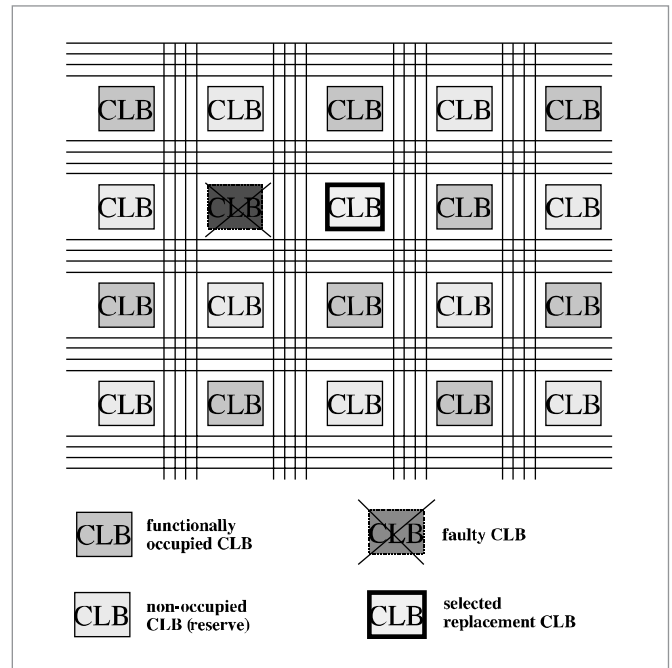


**Abbildung 5:**  
Reparatur durch Reihenverschiebung auf der FPGA-Struktur

Der einfachste bei FPGAs mögliche Ansatz ist, angelehnt an die bei Speicher-Feldern verwendeten Methoden, die Abschaltung einer kompletten Spalte oder Zeile im FPGA-Feld.

Damit ist eine reguläre Verschiebung der lokalen Verdrahtung verbunden (Abb. 5). Diese einfache Verschiebungsfunktion ist mit eingebetteten Werkzeugen und relativ leistungsarmen Prozessoren möglich [9]. Nachteilig ist, dass nicht nur ein fehlerhafter CLB-Baustein geopfert wird, sondern alle CLBs derselben Spalte, auch wenn sie vollfunktionsfähig sind.

Wir haben deshalb ein alternatives Verfahren untersucht, bei dem nur der fehlerhafte CLB-Block aus der Verdrahtung genommen und durch einen möglichst günstig gelegenen Ersatzblock ersetzt wird (Abb. 6). Im Feld re-programmierbare FPGAs benutzen häufig bereits „vorbereitete“ Rekonfigurationsmuster, die in kompakterer Form in einem Rekonfigurationsspeicher abgelegt sind. Diese Möglichkeit ist bei der Selbstreparatur nicht mehr anwendbar. Hier erfordert die große Zahl möglicher Rekonfigurationen für den Fehlerfall eine „eingebettete“ Berechnung der neuen Verdrahtungskonfiguration.



**Abbildung 6:**  
Reparatur durch Ersatz eines einzelnen CLBs

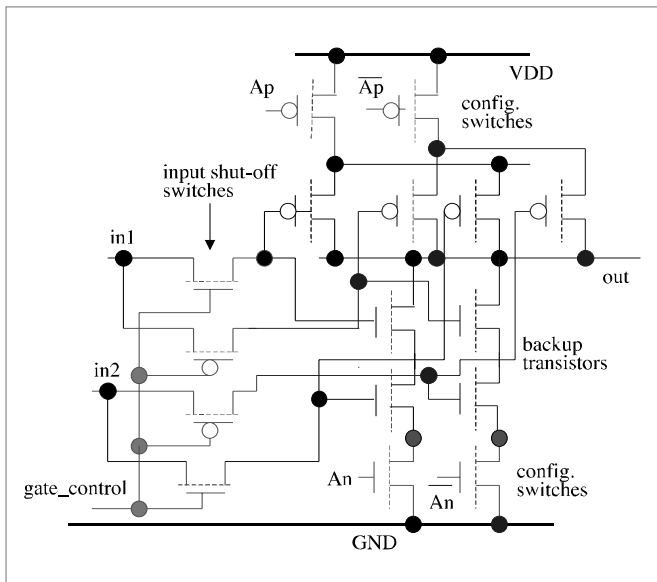
Diese Form der eingebetteten Entwurfsautomatisierung erweist sich als wesentlicher Engpass, da sie selbst die Leistungsfähigkeit eines „eingebetteten“ 32-Bit-Prozessors benötigt, um in Zeiten von Sekunden neue Verdrahtungsmuster zu erzeugen. Lediglich für sehr einfache Reparaturfunktionen ist mit Einfachprozessoren auszukommen. Nach ersten Erfahrungen erfordert die „eingebettete“ Neuberechnung der Verdrahtung für ein reales FPGA (Xilinx Vertex II) eine Code-Größe von ca. 250 Kbyte. Bei einer regulären, mit Redundanzen ausgestatteten CLB-Strukturierung, etwa in Schachbrett-Form, erweist sich die Verschiebung eines einzelnen Blocks als relativ problemlos möglich. Dann beträgt die Rechenzeit auf einem „eingebetteten“ Power PC PPC 405-Prozessor einige Sekunden. Ist der Ersatzblock nicht in direkter Nachbarschaft verfügbar, dann steigt die Rechenzeit beträchtlich an und beträgt bis zu einigen Minuten. Im realen System müsste also selbst für eine solch einfache und noch recht grob-granulare Reparaturfunktion eine „lokale“ eingebettete Rechenleistung vorgesehen werden, die nur ein eingebetteter „ausgewachsener“ 32-Bit RISC-Prozessor liefern kann.

#### 4 Fein-granulare Redundanz

Auch zu Reparaturfunktionen auf der Basis feingranularer Redundanz wurden Untersuchungen durchgeführt. Als Objekt wurde ein 2-fach NAND-Gatter in statischer CMOS-Logik mit 4 Transistoren ausgewählt. Analysiert man nur alle möglichen lokalen Fehlerfälle der Transistor-Ebene (Transistor stets nicht-leitend, stets leitend, Gate-Eingang gegen Masse kurzgeschlossen), so ergibt sich schon ein erheblicher Overhead, wenn alle Einzelfehler behoben und die betroffenen Gatter von der Stromversorgung abgetrennt werden sollen. Die Zahl der

Transistoren steigt um den Faktor 4, die Fläche des Gatters etwa um den Faktor 3. Darin noch nicht enthalten ist der zusätzliche Overhead für die Administration der Redundanz und die Implementierung von Reparatur-Funktionen. Als besonders problematisch erweist sich die Abtrennung von Transistoren mit kurzgeschlossenem Eingang durch Durchbrüche von Gate-Oxid-Schichten von parallelen Eingängen.

Es könnte sich als günstiger erweisen, alle Basis-Zellen gleich zu verdoppeln und ein solches Paar von als Doppel-Block zu administrieren. Auch hier muss eine Teilzelle im Fehlerfall wiederum von VDD und GND trennbar sein, wenn beide Zellen auf einen gemeinsamen Ausgang wirken, da sonst ein (VDD/GND)-Schluss eines internen Knotens einer Zelle gegen den Ausgangsknoten, etwa durch einen stets leitenden Transistor, die parallele Ersatzzelle ebenfalls belasten würde.



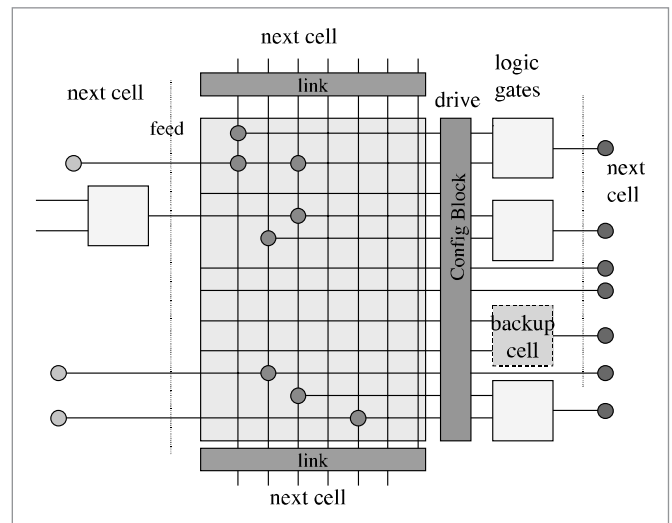
**Abbildung 7:**  
NAND-Zelle mit Zusatztransistoren für funktionalen Ersatz und Abtrennung im Fehlerfall

Die Verdopplung bringt sogar für den normalen Betrieb signifikante Vorteile, wenn einstellbar wäre:

- Einzelbetrieb des einen oder anderen Gatters mit Abschaltung des anderen Gatters (Sparfunktion)
- Paralleler Betrieb (verbesserte Treiber-Leistung, High-Speed-Betrieb)
- Komplette Abschaltung (Ruhefunktion).

Dazu benötigt jede Zelle mit zunächst 4 Transistoren (wieder für das 2-fach NAND) insgesamt 2 zusätzliche Schalttransistoren für die Trennung von Zu- und Abschaltung von VDD bzw. GND. Dies bedeutet eine lokale Verdreifachung der Transistorzahl. Da die Schalttransistoren keine zeitkritischen dynamischen Zustandsübergänge schalten müssen, kann man diese speziell robust auslegen., also z. B. größere als die minimalen Kanallängen verwenden. Ein neuralgischer Punkt bleibt dann die Ansteuerung, da natürlich auch mit transienten Fehlern auf Signalleitungen gerechnet werden muss.

Eine Redundanz, die für jedes Gatter genau eine dedizierte Ersatzzelle bereitstellen kann, trägt das Risiko einer Erschöpfung der Redundanz unter multiplen Fehlerbedingungen oder über eine lange Einsatzzeit mit sich. Es ist deshalb notwendig, auf der Gatter-Ebene weitere „freie“ Redundanzen vorzusehen. Leider ist eine solche systematische Bereitstellung weiterer Redundanz mit den in digitalen Logik-Schaltungen häufig verwendeten irregulären Verdrahtungen nicht kompatibel. Benötigt wird deshalb ein Übergang zu regulären Strukturen. Eine erste mögliche Basis bildet eine Basis-Zelle nach Abb. 8. Sie enthält jeweils die funktional benötigten Grundgatter, eine konfigurierbare Verdrahtung plus eine Ersatzzelle.

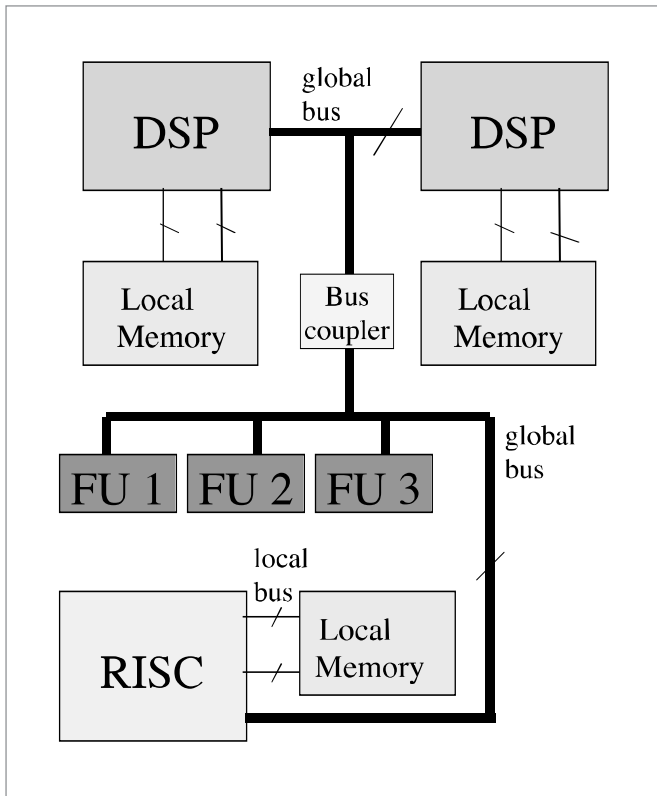


**Abbildung 8:**  
Konfigurierbare Basis-Zelle mit flexibel einsetzbarer Redundanz

Derzeit können wir den Aufwand für die Implementierung und die Administration solcher Zellen noch nicht abschätzen.

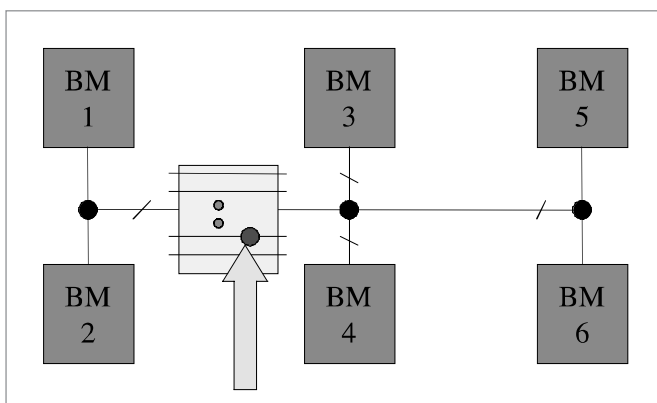
## 5 Bus-Strukturen

Ein bisher wenig beachtetes Problem für die Zuverlässigkeit von Schaltungen und Systemen stellen die globalen Leitungsverbindungen dar. Ein hochintegriertes System (System on a Chip-SoC) enthält heute stets mehrere Prozessoren, Speicher-Baugruppen, Logik-Baugruppen und oft auch noch analoge, digital-analoge und Hochfrequenz-Bausteine. Das Verbindungsnetz ist fast ein kleines Internet und wird oft auch in ähnlicher Form organisiert (Abb. 9). Das gesamte Verbindungsnetzwerk auf einem IC, das heute in bis zu 11 Ebenen übereinander abgelegt wird, kann eine Gesamtlänge von bis zu ca. 30 km aufweisen. Leider sind auch die Verbindungsleitungen auf den ICs mit kleineren Abmessungen zunehmend sowohl störanfällig gegenüber transienten (vorübergehenden) Fehlern aus auch gegenüber permanenten Ausfällen durch Fertigungsfehler oder Überlastung. Die Korrektur transienter Fehler ist seit längerer Zeit Stand der Technik. Mit einigen Zusatzleitungen ist eine Fehlererkennung und sogar Fehlerkorrektur möglich und Stand der Technik. Ein solche Fehlerkorrektur findet sich bereits im PC auf dem Leitungsbündel vom Rechnerkern zum Hauptspeicher.



**Abbildung 9:**  
Struktur eines „System on a Chip“ (SoC)

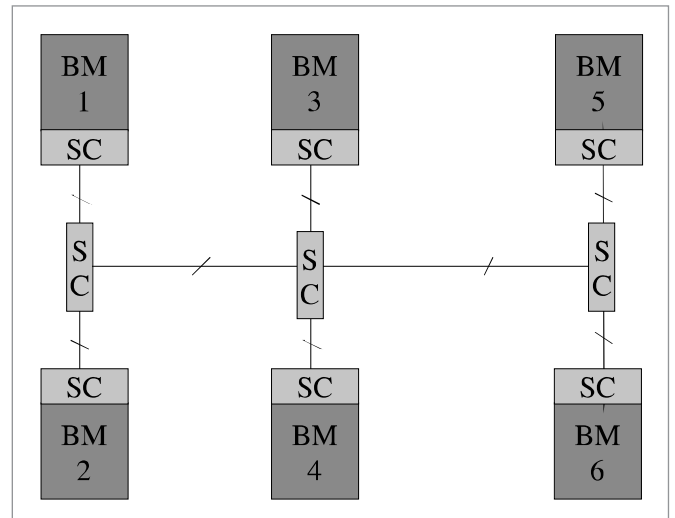
Leider kann eine solche Fehlertoleranz- Architektur nur eine geringe Anzahl permanenter Fehler verkraften, bevor sie versagt. Das Problem ist in Abb. 10 dargestellt. Ein Netzwerk, als Bus bezeichnet, wird von mehreren Baugruppen (Bus-Mastern) gemeinsam genutzt.



**Abbildung 10:**  
Bus mit permanentem Fehler

Eine einzige, in einer Sektion der Bus-Struktur unterbrochene Leitung hat nun leider globale Auswirkungen auf das gesamte Netzwerk. Das heißt, der Fehler pflanzt sich ungehindert fort.

Techniken, die ein solches Verhalten intelligent verhindern, sind bisher nur ansatzweise vorhanden. Einen möglichen Ausweg zeigt Abb. 11.



**Abbildung 11:**  
Bus-Struktur mit einzelnen Segmenten

Der Bus ist in einzelne Segmente aufgeteilt, welche durch intelligente Segment-Koppler einzeln verwaltet und auch repariert werden können. Dazu stehen in jedem Segment neben den normalen Leitungen Zusatzleitungen zur Verfügung.

Benötigt wird nun ein Schema zur Umgruppierung der Leitungen in einem einer Sektion, das folgenden Anforderungen genügt:

- Reparatur der wichtigsten Fehlertypen (Unterbrechung, Überbrückung, Kopplung).
- Minimaler Aufwand bezüglich der Zahl der benötigten Ersatzleitungen, der Schalter zur wahlweisen Kopplung der Leitungen in den SC-Elementen und der Zusatz-Logik für die Administration.
- Unabhängigkeit der Segmente voneinander, insbesondere Beibehaltung der Anschluss-belegungen eines Segmentes unabhängig von der (Re-) Konfiguration.

Erste Ergebnisse zeigen, dass es geschickte Möglichkeiten zur Vertauschung von Leitungen gibt, die solche Anforderungen erfüllen. Dazu werden zunächst die Leitungen in Bündel von z. B. Leitungen organisiert. Davon werden 2 Leitungen als Ersatzleitungen benötigt. Damit ist es dann möglich, mit nur 4 einstellbaren Varianten der Kopplung und jeweils 32 Schaltern alle dynamischen Kopplungsfehler und alle Einzelfehler innerhalb eines Segmentes zu reparieren. Will man auch alle Doppelfehler reparieren, so werden vier von 8 Leitungen als Ersatz benötigt.

Einen nicht geringen Zusatzaufwand erfordert wie in allen Fällen die Verwaltung der Fehler und die Administration der Reparatur. Für solche Zwecke eignet sich der am Lehrstuhl Technische Informatik entwickelte Test-Prozessor. Er kann darüber hinaus die Tests zur Auffindung von Fehlerstellen ausführen.

## 6 Ungelöste Probleme

Ein derzeit allenfalls ansatzweise gelöstes Problem ist die Fehlerdiagnose in großen Logik-Schaltungen. Es ist extrem aufwändig bis nahezu unmöglich, beim Auftreten eines logischen Fehlers das fehlerhafte Gatter zu ermitteln. Eine fein-granulare Reparaturfunktion ist aber ohne eine fein-granulare Fehlerdiagnose nicht möglich. Auch dazu muss ein genügend leistungsfähiger Prozessor mit „eingebetteter“ Software verfügbar sein. Es ist zu erwarten, dass die minimal erreichbare Granularität durch die Fehlerdiagnose begrenzt wird. Reparaturfunktionen, welche nicht durch permanente und irreversible Aktionen wie das Durchbrennen einer Leitung permanent gemacht werden, müssen registriert, administriert und nach einem System-Neustart nochmals durchgeführt werden. Der dafür benötigte Overhead dürfte beträchtlich sein. Wir erwarten deshalb, dass reale selbstreparierende Systeme in Zukunft auf der Basis hierarchisch organisierter Redundanz funktionieren werden, bei der die Komplexität der kleinsten redundant vorhandenen Bausteine bei 50-100 Transistoren liegt und die über die Fähigkeit zur permanenten Abtrennung fehlerhafter Baugruppen verfügt.

### Literatur

- [1] BREUER, M. A.; GUPTA, S.; MAK, T. M.: „Defect and Error Tolerance in the Presence of Massive Numbers of Defects“, IEEE Design and Test of Computers, Vol. 21, No. 3, May/June 2004, pp. 216-227
- [2] ZORIAN, Y.; GIZOPOULOS, D.: „Design for Yield and Reliability“ (Guest Editor’s Introduction), IEEE Design and Test of Computers, Vol. 21, No. 3, May/June 2004, pp. 177-182
- [3] SIRISANTANA, S.; PAUL, B. C.; ROY, K.: „Enhancing Yield at the End of the Technology Roadmap“, IEEE Design and Test of Computers, Vol. 21, No. 6, November/December 2004, pp. 563-57
- [4] NARAYANYAN, V.; XIE, Y.: „Reliability Concerns in Embedded System Design“, IEEE Computer Magazine, January 2006, pp. 118-123
- [5] DEHON, A.; NAEIMI, H.: „Seven Strategies for Tolerating Highly Defective Fabrication“, IEEE Design and Test of Computers, Vol. 22, No. 4, July/August 2005, pp. 306-315
- [6] HE, C.; JACOME, M.; DE VACIANA, G.: „A Reconfiguration-Based Defect-Tolerant Design Paradigm for Nanotechnologies“ IEEE Design and Test of Computers, Vol. 22, No. 4, pp. 316-326, July/August 2005.
- [8] SIA ROADMAP, 2005 UPDATE: Semiconductor Industries Association, <http://www.itrs.net/common/2005ITRS/Test2005.pdf>
- [9] MITRA, S.; HUANG, W.-J.; SAXENA, N. R.; YU, S.-Y.; MCCLUSKEY, E. J.: „Reconfigurable Architecture for Autonomous Self Repair“, IEEE Design and Test of Computers, Vol. 21, No. 3, May/June 2004, pp. 228-240
- [10] SHOUKOURIAN, S.; VARDANIAN, V.; ZORIAN, Y.: „SoC Yield Optimization via an Embedded-Memory Test and Repair Infrastructure“, IEEE Design and Test of Computers, Vol. 21, No. 3, May/June 2004, pp. 200-207
- [11] KIM, I.; ZORIAN, Y.; KOMORIYA, G.; PHAM, H.; HIGGINS, F. P.; LEWANDOWSKI, J. L.: „Built in Self Repair for Embedded High Density SRAM“, Proc. IEEE Int Test Conf. 1998, pp. 1112-1118

[12] LA ROSA, A.; LAVAGNO, LUCIANO; PASSERONE, CLAUDIO: „Software Development for High-Performance, Reconfigurable, Embedded Multimedia Systems“, IEEE Design and Test of Computers, Vol. 22, No. 1, January/February 2005, pp. 28-38

[13] CHEN CHANG JOHN WAWRZYNEK, R. W. BRODERSEN: “BEE2: A High-End Reconfigurable Computing System”, IEEE Design and Test of Computers, Vol. 22, No. 2, 2005, pp. 114-125

[14] DOKUMENTATION FÜR XILING Vertex II FPGAs, Xilinx Inc., 2004



*Dipl.-Inf. René Kothe hat 2004 sein Studium der Informatik an der BTU Cottbus abgeschlossen. Seit Dezember 2004 ist er als wissenschaftlicher Mitarbeiter am Lehrstuhl Technische Informatik tätig. Seine Stelle wird gefördert durch die Deutsche Forschungsgemeinschaft (DFG) im Projekt HITSOC (Hierarchische Test-Technologie für Systems on a Chip).*



*Sven Habermann (M. Sc.) hat sein Studium der Informations- und Medientechnik 2005 mit der Master-Arbeit über Selbstreparatur-Funktionen in programmierbarer Logik am Lehrstuhl Technische Informatik abgeschlossen. Er ist seit Anfang 2006 Mitarbeiter bei der Philips Semiconductors GmbH in Hamburg.*



*Heinrich Theodor Vierhaus hat 1975 ein Studium der Elektrotechnik an der Ruhr-Universität Bochum abgeschlossen. Die Promotion zum Dr.-Ing. erfolgte 1983 an der Universität Siegen. Er wurde 1996 auf den Lehrstuhl Technische Informatik der BTU Cottbus berufen.*