Effiziente Mehrfach-Fehlersimulation von Analogschaltungen

DISSERTATION

zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

durch die Fakultät Wirtschaftswissenschaften der
Universität Duisburg-Essen

vorgelegt von

Eduard Weber

geboren am 23.04.1982 in Michailowka

Essen, 2018

Datum der Einreichung: 19.07.2017
Datum der mündlichen Prüfung: 05.03.2018

Erstgutachter: Prof. Dr. Klaus Echtle

Zweitgutachter: Prof. Dr. Tobias Hoßfeld

Veröffentlichungen

Die Ergebnisse der vorliegenden Arbeit wurden teilweise veröffentlicht:

- Weber, Eduard; Echtle, Klaus (2014): Simulation-Based Reliability Evaluation for Analog Applications. In: 2014 IEEE International Reliability Physics Symposium (IRPS). Waikoloa, HI, USA, 1- 5 June 2014. The Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08855, USA,

- Weber, Eduard; Echtle, Klaus (2015): Efficient Simulation of Multiple Faults for Reliability Analysis of Analogue Circuits. In: DEPEND 2015. The Eighth International Conference on Dependability. Venice, Italy, 23 - 28 August 2015. IARIA: International Academy, Research, and Industry Association.

- Weber, Eduard; Echtle, Klaus (2016): Multiple Faults Simulation of Analogue Circuits. In: International Journal On Advances in Systems and Measurements vol 9 (1&2). Online verfügbar unter http://www.iariajournals.org/systems_and_measurements.

Abstract

Fault simulation is a method for assessing fault tolerance and reliability that examines the system behaviour under defined assessment conditions in the presence of faults, which are explicitly injected. The fault models specify the faults to be injected into the components of a system, to reveal the effect on the functioning of the circuit. Fault injection can be applied to both hardware and software systems. This work is dedicated to the fault injection in analogue hardware circuits, where different types of faults are inserted into the simulation of an analogue circuit. The focus is put on the analysis of multiple faults and several faults types per component. This combination provides the desired flexibility. However, in most cases sufficiently accurate results can only be obtained with very high computational cost, because each individual simulation of an analogue circuit requires a considerable amount of time. To reduce the computational effort, an efficient method, the fault class algorithm (Fehlerklassen-Algorithmus in german, FKA), has been developed for more efficient multiple fault analysis. Using the FKA method, the fault tolerance level of a system can be determined accurately by interpolation for a substantial portion of the fault injections without conducting the respective simulations under fault injection, which saves many simulation runs. The accuracy of the method has been evaluated by simulating a sufficient number of sample circuits, that have confirmed the suitability and usefulness of the fault class algorithm. For a newly designed system to be analysed by fault simulation, representative fault models are required to express the relevant failure types of the circuit elements. Components of an analogue circuit always may always exhibit several types of faults. To simulate the faults taken from such model as simple and as flexible as possible, a script-based fault modelling approach is presented in this work. Thus, every conceivable type of fault can be described and injected into any of the simulated circuit elements. This way, you can define both detailed and abstract malfunctions. Both the efficiency of the fault classes algorithm as well as the implemented fault simulator have been validated in this thesis by various examples of analogue circuits.

Keywords:
Fault simulation; fault modelling; multiple fault injection; reliability prediction

ZUSAMMENFASSUNG

Eine Fehlersimulation ist eine Methode zur Bewertung der Fehlertoleranz und der Zuverlässigkeit, die unter definierten Prüfbedingungen das Systemverhalten beim Auftreten von Fehlern, die gezielt injiziert werden, untersucht. Fehlermodelle legen fest, welche Fehler in die Komponenten eines Systems injiziert werden, um die Auswirkungen auf die Funktionsfähigkeit der Schaltung zu prüfen. Eine Fehlerinjektion kann sowohl in Hardware- als auch in Softwaresysteme erfolgen. Diese Arbeit widmet sich der Fehlerinjektion in analoge Hardware-Schaltungen, wobei verschiedene Fehlerarten in eine Simulation der Analogschaltung eingebracht werden. Dabei liegt der Fokus auf den erweiterten Analysemöglichkeiten bei der Injektion von Mehrfachfehlern und unterschiedlichen Fehlerarten pro Komponente. Diese Kombination bietet die erstrebte Flexibilität, jedoch sind hinreichend genaue Ergebnisse in den meisten Fällen nur durch einen sehr hohen Rechenaufwand erhältlich, weil jede einzelne Simulation einer Analogschaltung einen beträchtlichen Zeitaufwand erfordert. Um den Rechenaufwand zu reduzieren, wurde in dieser Arbeit eine effiziente Methode, der Fehlerklassen-Algorithmus (FKA), für die Analyse von Mehrfachfehlern entwickelt. Mit dieser Methode kann der Toleranzgrad eines Systems für einen Teil der Fehlerinjektionen durch Interpolation ziemlich genau bestimmt werden, ohne die betreffenden Fehlerinjektionen vornehmen zu müssen, was entsprechend viele Simulationsläufe einspart. Die Genauigkeit wurde (rechenintensiv) durch einen Fehlersimulator mit einer hinreichenden Anzahl von Beispielschaltungen quantitativ bewertet, wodurch die Tauglichkeit des Fehlerklassen-Algorithmus bestätigt wurde. Damit ein neues System durch Fehlerinjektion geprüft werden kann, werden repräsentative Fehlermodelle für mehrere Ausfallarten für jede Bauelement-Art benötigt. Damit die betreffenden Fehlermodelle möglichst einfach und flexibel erstellt werden können, wird in dieser Arbeit eine skriptbasierte Fehlermodellierung vorgestellt. Damit kann jede erdenkliche Fehlerart beschrieben und in beliebige Bauelemente der simulierten Schaltung injiziert werden. Auf diese Weise können sowohl akkurate und als auch abstraktere Fehlermodelle definiert werden. Sowohl die Effizienz des Fehlerklassen-Algorithmus als auch der implementierte Fehlersimulator wurden in dieser Arbeit an Hand verschiedener Beispiele durch Simulationsergebnisse validiert.

Schlagwörter:
Fehlersimulation; Fehlermodellierung; Mehrfach-Fehlerinjektion; Zuverlässigkeitsvorhersage

DANKSAGUNG

Bei allen, die mich in den letzten Jahren, während der Entstehung dieser Arbeit begleitet haben, möchte ich mich an dieser Stelle bedanken.

Ein großes Dankeschön an Professor Klaus Echtle. Insbesondere für die mühevolle Korrektur dieser Doktorarbeit, für das stets offene Ohr zu meinen Anliegen, die Motivation und Vorschläge aus den hilfreichen Gesprächen meine Arbeit zu verbessern. Ich hätte mir keine bessere Unterstützung wünschen können.

Ebenso danke ich Prof. Dr. Tobias Hoßfeld für seine Bereitschaft, das Zweitgutachten zu erstellen.

Des Weiteren richte ich einen herzlichen Dank an meine Familie, die mir stets den Rücken gestärkt und mich während dieser Jahre unermüdlich unterstützt haben. Meine Frau, meine Eltern und meine Schwester haben mir in den letzten Jahren viel Kraft und Halt gegeben.

INHALTSVERZEICHNIS

Abstract I
Zusammenfassung II
Danksagung IV
Inhaltsverzeichnis V
Abkürzungsverzeichnis VII
Abbildungsverzeichnis VIII
Tabellenverzeichnis XI
1 Einführung 1
1.1 Fehlersimulation von analogen Schaltungen 3
1.2 Notwendigkeit der Zuverlässigkeitsbewertung 7
1.3 Zuverlässige Systeme im Betrieb 10
1.4 Zielsetzung der Arbeit 14
1.5 Aufbau der Arbeit 15
2 Grundlagen der Fehlersimulation und Zuverlässigkeit 16
2.1 Definitionen und Elemente von Zuverlässigkeit 16
2.2 Prüfen und Diagnose von analogen Schaltungen 26
2.2.1 Fehlerdiagnose bei analogen Schaltungen 26
2.2.2 Analoge Schaltungssimulation 27
2.3 Fehlerinjektion 29
2.4 Boolesche Modelle zur Zuverlässigkeitsbewertung 31
2.5 Zuverlässigkeitsbewertung bei Auftreten von mehreren Fehlerarten 39
2.6 Komplexität der Fehlersimulation bei Mehrfachfehlern und mehreren Fehlerarten 45
3 Fehlersimulation 47
3.1 Fehlersimulation von analogen und gemischten Schaltungen 47
3.2 Von physikalischen Fehlern zu adäquaten Fehlermodellen 50
3.3 Realistische Fehlermodelle auf Komponentenebene 52
3.4 Bisherige Arbeiten zu analogen Fehlersimulatoren 57
3.5 Übersicht über analoge Fehlersimulatoren 59
3.6 Lösungsansätze zur Reduktion der Simulationszeit 61
3.6.1 Zufällig ausgesuchte Stichprobe 61

3.6.2 Simulation auf höherem Abstraktionslevel 61
3.6.3 Parallele Simulation einer Schaltung 62
3.6.4 Nach Auftrittswahrscheinlichkeit von Fehlerarten 63
3.6.5 Frühzeitiger Abbruch einer Fehlersimulation 64
3.6.6 Monotonie-Regel bei Mehrfachfehlern 64
3.6.7 Kombination mehrerer Beschleunigungstechniken 66
4 Fehlerklassen-Algorithmus (FKA) 67
4.1 Effizienter Algorithmus bei Mehrfachfehlersimulation 67
4.1.1 Definitionen des Fehlerklassen-Algorithmus 68
4.1.2 Implementierung von FKA 73
4.1.3 Formale Beschreibung der Mehrfachfehlersimulation 76
4.2 Analoger Fehlersimulator 78
4.2.1 Überlegungen zum entwickelten Fehlersimulator 78
4.2.2 Entwicklungshintergrund des Fehlersimulators 80
4.2.3 Abgrenzung von anderen Fehlersimulatoren 81
4.2.4 Fehlermodellierung und Fehlerinjektion 83
4.2.5 Datenauswertung und Toleranzbereich 87
4.2.6 Ablauf der Fehlerinjektion 90
5 Bewertung 95
5.1 Simulierte Schaltungen 97
5.2 Ergebnisse der Fehlersimulation und die Bewertung des Fehlerklassen-Algorithmus (FKA) 98
5.3 Begründung der Wirksamkeit des FKA 101
6 Zusammenfassung 105
7 Ausblick 107
8 Literaturverzeichnis 109
Anhang 116
Anhang A: Grundlagen zur Zuverlässigkeitstheorie 116
Anhang B: Erweitertes Fehlermodell für die Fehlerinjektionen 124
Anhang C: Details zu untersuchten Schaltungen 125

ABKÜRZUNGSVERZEICHNIS

AFM	Analog fault modelling
APLAC	An Object-Oriented Analog/RF Circuit Simulator and Design Tool (Bezeichnung für einen objektorientierten Schaltungssimulator und Entwurfswerkzeug)
ATPG	Automatic test pattern generation
AWR	Applied Wave Research Corporation (Unternehmen)
BDD	Binary decision diagram
CDF	Cumulative distribution function
CPU	Central Processing Unit, Prozessor
DFT	Design for testability
EDA	Electronic design automation
ESP	Elektronisches Stabilitätsprogramm
FAC	Flight Augmentation Computer
FTA	Fault tree analysis
HF	Hochfrequenz
IC	Intergrated Circuits
KVF	Kumulative Verteilungsfunktion
MTBF	Mean Time Between Failures
PMU	Power management unit
RBD	Reliability Block Diagram
RF	radio frequency
RTLU	Rudder travel limiter unit
SOC	System-on-Chip
SPICE	Simulation Program with Integrated Circuit Emphasis
VB-Skript	Visual Basic Script von Microsoft (engl: VBScript oder VBS)
VHDL	Very High Speed Integrated Circuit Hardware Description Language
ZBD	Zuverlässigkeitsblockdiagramme

ABBILDUNGSVERZEICHNIS

Abbildung 1. Gliederung der Arbeit 15
Abbildung 2. Unterscheidung zwischen Fehlerursache, Fehler 20
Abbildung 3. Übersicht der Kategorien von Fehlerursachen 22
Abbildung 4. Die Ebenen der Fehlerinjektion und Fehlermodellierung . 30
Abbildung 5. Vergleich der Modellierung mittels RBD und FTA 33
Abbildung 6. Zuverlässigkeitsblockdiagramme für einfache Stromkreise 33
Abbildung 7. Beispiel einer Schaltung für die Zuverlässigkeitsanalyse .. 41
Abbildung 8. Beispiel für ein Zuverlässigkeitsblockdiagramm einer Schaltung 55
Abbildung 9. Beispiel für explizite Redundanz einer Schaltung. 56
Abbildung 10. Systemfunktion als KV-Diagramm 56
Abbildung 11. Prüfaufwand (in %) in Abhängigkeit von der Signalart 57
Abbildung 12. Wissenschaftliche Publikationen zu analogen Fehlersimulatoren 58
Abbildung 13. Komplexität der Fehlersimulation 65
Abbildung 14. Vorgehen für das Aufstellen von der MFK_3 Menge der Fehlerklassen 72
Abbildung 15. Algorithmus für die Suche der Fehlerklassen-Ketten und Bestimmung des Toleranzgrades 74
Abbildung 16. Schichtenarchitektur des Fehlersimulators 80
Abbildung 17. Grundsätzliche Struktur der Fehlersimulation für Mehrfachfehlerinjektion 85
Abbildung 18. Benutzeroberfläche der Fehlersimulation für Mehrfachfehler 86
Abbildung 19. Beispiel für ein x-y-Diagramm ohne Toleranzbereich 87
Abbildung 20. Beispiel für einen x-y Graph mit Toleranzbereich und Toleranzgrenze (rot) 88
Abbildung 21. Messreihen für die Simulation mit Parametervariationen der Komponenten 89
Abbildung 22. Befehlsbaum bzw. Aufrufgraph für die VB-Skript-basierte Fehlersimulation 90
Abbildung 23. Datenmodell des entwickelten Fehlersimulators 91
Abbildung 24. Beispielimplementierung (VB-Skript Subroutine) der Fehlerart Kurzschluss 92
Abbildung 25. Kondensatorschaltung mit Leckstrom-Fehlermodell 93
Abbildung 26. Ausgangssignale der Kondensatorschaltung 93

Abbildung 27. Beispielimplementierung (VB-Skript Subroutine) der Fehlerart Leckstrom bei Kondensatoren *94*
Abbildung 28. Beispiel für eine FK-Kette mit zunehmender (C), abnehmender (D) und gleichbleibender Fehlerart (A) *102*
Abbildung 29. Venn-Diagramme *117*
Abbildung 30. Typischer Verlauf der „Badenwannenkurve“ *122*
Abbildung 31. Beispielimplementierung (VB-Skript Subroutine) der Fehlerart Kurzschluss mit mehr als zwei Kontaktstellen *124*
Abbildung 32. Schaltung des LM741 Operationsverstärkers *126*
Abbildung 33. Ausgangssignale und Toleranzbereich des LM741 Operationsverstärkers *126*
Abbildung 34. Schaltung des VHF/UHF Verstärkers *127*
Abbildung 35. Ausgangssignale und Toleranzbereich des VHF/UHF Verstärkers *128*
Abbildung 36. Schaltung des Limiter BSP *129*
Abbildung 37. Ausgangssignale und Toleranzbereich des Limiter BSP . *130*
Abbildung 38. Schaltung des Spannungsstabilisators *131*
Abbildung 39. Ausgangssignale und Toleranzbereich des Spannungsstabilisators *131*
Abbildung 40. Schaltung des BJT_Amp_Complete: Small Signal *132*
Abbildung 41. Ausgangssignale und Toleranzbereich des BJT_Amp_Complete: Small Signal *132*
Abbildung 42. Schaltung des BJT_Amp_Complete: Large Signal *133*
Abbildung 43. Ausgangssignale und Toleranzbereich des BJT_Amp_Complete: Large Signal *133*
Abbildung 44. Schaltung der Gilbert Cell *134*
Abbildung 45. Ausgangssignale und Toleranzbereich der Schaltung Gilbert Cell *134*
Abbildung 46. Schaltung der IAM_81_Gilbert_Cell_Noise *135*
Abbildung 47. Ausgangssignale und Toleranzbereich bei IAM_81_Gilbert_Cell_Noise *135*
Abbildung 48. Schaltung Limiter_HB_Aplac *136*
Abbildung 49. Ausgangssignale und Toleranzbereich der Schaltung Limiter_HB_Aplac *137*
Abbildung 50. Schaltung des Low_Power_Mixer *138*
Abbildung 51. Ausgangssignale und Toleranzbereich der Schaltung Low_Power_Mixer *138*
Abbildung 52. Schaltung des Nonlinear Noise Switch Views *139*
Abbildung 53. Ausgangssignale und Toleranzbereich der Schaltung Nonlinear Noise Switch Views *140*
Abbildung 54. Schaltung des UHF VHF Power Amplifiers *141*

Abbildung 55. Ausgangssignale und Toleranzbereich des UHF VHF Power Amplifier..... 142
Abbildung 56. Schaltung des Impuls-Signalverstärker..... 143
Abbildung 57. Ausgangssignale P und Q des Impuls-Signalverstärkers 144
Abbildung 58. Ausgangssignale und Toleranzbereich des Impuls-Signalverstärkers..... 144

TABELLENVERZEICHNIS

Tabelle 1. *Grundlegende Zuverlässigkeitsblock-Diagramme und Zuverlässigkeitsfunktionen* *34*

Tabelle 2. *Fortsetzung: Grundlegende Zuverlässigkeitsblock-Diagramme und Zuverlässigkeitsfunktionen* *35*

Tabelle 3. *Kalkulation der Systemzuverlässigkeit anhand der Wahrheitstabelle* *37*

Tabelle 4. *Systemdiagramme und zugehörige Zuverlässigkeitsfunktionen* *40*

Tabelle 5. *Beispielrechnung für den Aufwand der Fehlersimulation* *45*

Tabelle 6. *Physikalische Defekte und elektronische Fehlermodelle* ... *50*

Tabelle 7. *Fortsetzung der physikalischen Defekte und elektronischen Fehlermodelle* *51*

Tabelle 8. *Fehlermodelle für Transistoren* *52*

Tabelle 9. *Fortsetzung der Fehlermodelle für den Widerstand, die Diode und den Kondensator* *53*

Tabelle 10. *Wichtige Parameter, die die Komponenten einer Schaltung beeinflussen* *62*

Tabelle 11. *Statistische Richtwerte für Fehleraten bei elektronischen Komponenten in %* *63*

Tabelle 12. *Bewertung mehrerer Techniken zur Beschleunigung der Fehlersimulation* *66*

Tabelle 13. *Eigenschaften der untersuchten Schaltungen* *97*

Tabelle 14. *Beispiel für die Anzahl der Simulationsläufe* *98*

Tabelle 15. *Simulationsergebnisse mit Fehlerklassen-Algorithmus (FKA)* *100*

Tabelle 16. *Vergleich der Analyseergebnisse zur T-Übereinstimmung von Fehlerklassen* *104*

Tabelle 17. *Beziehung und Zusammenhang zwischen den Funktionen* F(t), f(t), R(t) und h(t) *122*

Tabelle 18. *Unterschiedliche Formen der Zuverlässigkeitsfunktion und der Fehlerrate der am häufigsten verwendeten Distributionen* *123*

Tabelle 19. *Eigenschaften der exemplarischen Schaltungen* *125*

1 EINFÜHRUNG

Fehlersimulation (engl. fault simulation) bedeutet „Simulation unter Fehlerinjektion“. Zur Fehlersimulation gehört aber nicht nur die Fehlerinjektion selbst. Vielmehr schließt sie auch einige vor- bzw. nachgelagerte Schritte ein. Der Prozess der Fehlersimulation kann in vier aufeinander aufbauende Schritte bzw. Teilprozesse unterteilt werden: Fehlermodellierung, Fehlerinjektion, Simulation mit Fehler und die Datenauswertung nach der Simulation. Die grundsätzlichen Funktionen eines Fehlersimulators für analoge Schaltungen umfassen das Aufstellen bzw. Aktualisieren der Fehlerliste, die Simulation der fehlerinjizierten Schaltung mit Abgleich der Signale aus der fehlerfreien Schaltung und die abschließende Analyse der Simulationsergebnisse. Schaltungsfehlersimulation (engl. circuit fault simulation) ist eine Simulationstechnik, welche die Auswirkungen von Komponentenfehlern auf die Funktionsfähigkeit einer Schaltung analysiert. Die letzten Jahrzehnte sind gekennzeichnet durch eine fundierte Entwicklung der Software zur Entwurfsautomatisierung elektronischer Systeme (engl. Electronic Design Automation, EDA). So können die EDA-Tools heutzutage eine Vielzahl von Aufgaben erfüllen, angefangen von der Schaltungssimulation bis hin zum finalen Schaltungslayout. Die so entwickelten und dann in Serie hergestellten Schaltungen sind nicht als fehlerfrei anzunehmen. Schaltungen erfahren Störungen durch die Umwelt, es fallen Komponenten aus oder sie enthalten eventuell Entwurfsfehler bzw. Entwurfsschwächen (z.B. ungünstige Abstände zwischen Leiterbahnen, so dass Übersprechen möglich ist). Eine auf EDA-basierte Fehlersimulation kann auf der Grundlage der Schaltungssimulation das Verhalten der Schaltung im Fehlerfall bzw. unter Störungen simulieren und ggf. mit Hilfe von Zuverlässigkeitsmodellen bewerten. Das bedeutet in erster Linie, dass die EDA-Tools als Basis dafür genutzt werden, um auf Verhaltensmodellen und Fehlermodellen der Schaltungskomponenten (der Bauelemente) ein Zuverlässigkeitsmodell des untersuchten Systems (der Schaltung) zu erhalten. Angenommene Fehler werden durch Injektion der Komponentenfehler in das entworfene Schaltungsmodell eingefügt und durch die Simulation wird das Fehlerverhalten der Schaltung analysiert. Anschließend wird eine Bewertung der Schaltung (nach vorher festgelegten Erfolgskriterien) vorgenommen, um die Funktionsfähigkeit unter Fehlerauftritt zu beurteilen und damit auch eine quantitative Zuverlässigkeitsanalyse zu realisieren. In der Realität können mehrere Fehlerarten pro Komponente auftreten. Darüber hinaus

können in einer Schaltung entweder nur eine Komponente oder mehrere Komponenten zugleich fehlerhaft sein (Einzelfehler bzw. Mehrfachfehler). Um ein aussagekräftiges Zuverlässigkeitsmodell zu erhalten, ist es wichtig, möglichst alle Fehlerkombinationen zu berücksichtigen. Die Betrachtung mehrerer Fehler gleichzeitig ist aus den folgenden Gründen wichtig. Eine Mehrfachfehlerdiagnose ist besonders relevant, wenn ein neuer Schaltungsentwurf eingeführt wird und eine hohe Ausfalldichte vorliegen könnte. Das Auftreten von Fehlern steigt in rauer Umweltumgebung häufig stark an, was zur Entwurfszeit noch teilweise unvorhersehbar ist. Auch mehrere Parameterfehler können gleichzeitig als Folge von Alterung, Umwelteinfluss und Entwurfsfehlern vorhanden sein. Das betrifft alle Elemente gleichzeitig. Ebenso können verschiedene Fehlerarten in Bauelementen von analogen Schaltungen vorhanden sein und je nach Auftrittsort auch unterschiedliche Folgefehler auslösen, so dass de facto ein Mehrfachfehler vorliegt. Die Einschränkung der Einzelfehlersimulation und einer Ausfallart kann demzufolge zu unkorrekten Auswertungsergebnissen führen.

1.1 FEHLERSIMULATION VON ANALOGEN SCHALTUNGEN

Die Erhöhung in der Schaltungsdichte erlaubt es Schaltungsdesignern, eine immer größere Anzahl von Schaltungskomponenten (z.B. Transistoren) zu nutzen. Die Folge, nicht nur für den Fertigungsprozess, sind tendenziell mehr Fehler auf der Transistorebene. Somit wird es umso wichtiger, die Fehlermodellierung und -simulation in möglichst frühen Design-Phasen (vor Beginn der Fertigung und der Nutzungsphase) zu betreiben. Bei sicherheitsrelevanten Systemen kann man nicht auf die ersten Ausfälle in der Betriebsphase bzw. auf vergleichbare Systeme im Einsatz warten. Daher ist es wichtig, das Verhalten und die Fehlertoleranz des untersuchten Systems möglichst in den frühen Phasen der Entwicklung zu analysieren. Die Fehlersimulation ist ein sehr hilfreicher Ansatz für die Untersuchung der Fehlertoleranz-Fähigkeit einer Schaltung.

Für digitale Schaltungen sind automatisierte Prüf- und Verifikationsmethoden verfügbar und weit verbreitet. Für analoge und gemischte Schaltungen ist dies noch nicht der Fall. Die Fehlersimulation ist von Simulationen auf Transistorebene (engl. transistor level simulaiton) abhängig, da nur auf diese Weise exakte Evaluierungsergebnisse erzielt werden können. Anders als bei der digitalen Simulation sind nicht nur die beiden Signalpegel 0 und 1 zu berücksichtigen, sondern das Kontinuum der analogen Spannungspegel und Ströme, die jeweils durch ein System von (oftmals zeitabhängigen) Gleichungen beschrieben werden, die das Simulations-Tool lösen muss. Die relativ langen Simulationszeiten auf Transistorebene verhindern eine schnelle Fehlersimulation bei aufwendigen und komplexen gemischten Schaltungen. Die klassischen Zuverlässigkeitsprüfmethoden untersuchen nur Einfachfehler bzw. eine Ausfallart (engl. fault mode) pro Komponente. Schaltungen können aber auch Fehler- und Ausfallarten enthalten, die als eine Kombination (z.B. temperaturabhängige Parameterfehler) aus mehreren Fehlern (engl. multiple faults) auftreten und als solche untersucht werden müssen. Werden Mehrfachfehler und/oder mehrere mögliche Fehlerarten pro Komponente untersucht, steigt die absolute Simulationszeit wegen der Anzahl der Kombinationen von Fehlerfällen enorm an.

Mit der zunehmenden Erhöhung der Schaltungsdichte und gleichzeitigen Reduktion der Chipfläche ist das Auftreten von Mehrfachfehlern wahrscheinlicher geworden. Würde für die Kombinationen aus Mehrfachfehlern und mehreren Fehlerarten pro Komponente ein nicht optimierter Fehlersimulator genutzt werden, wären die Untersuchungen für die meisten real eingesetzten Schaltungen wegen der langen Rechendauer

oftmals praktisch nicht durchführbar. Das Prüfen der Korrektheit des Entwurfes bzw. des Auftretens von Herstellungsfehlern integrierter Schaltungen (engl. integrated circuits, IC) stellt die Hauptkosten heutiger Produktentwicklungen elektronischer Produkte dar. Wegen der zunehmenden Komplexität und der kürzer werdenden Lebenszyklen der Produkte nimmt die Analyse der Zuverlässigkeit weiter an Bedeutung zu. Die in dieser Arbeit verwendete Fehlersimulation soll die Frage beantworten, wie zuverlässig Schaltungen sind, die gemäß einem gegebenen Entwurf aufgebaut wurden. Im Rahmen der Fehlersimulation werden Fehler in ein Modell der Schaltung injiziert, das den im späteren Betrieb erwarteten Fehlern möglichst gut entsprechen soll. Besonders der hohe Kostenanteil für das Entwerfen und die Evaluation der Zuverlässigkeit analoger und gemischter Schaltungskomponenten (engl. mixed-signal components) unterstreichen die Wichtigkeit, günstige Verfahren dafür zu entwickeln. Die Ursache der deutlich höheren Kosten im Vergleich zu digitalen Schaltungen liegt im niedrigen Automatisierungsgrad analoger Schaltungsentwürfe und Schaltungstests. Bei digitalen Schaltungen erfolgt die Synthese und das Prüfen in automatisierten Schritten, während die analogen Bauelemente und Schaltungen meist von Hand und damit zeit- und kostenintensiv entworfen werden. Eine vielversprechende Methode ist die analoge Fehlermodellierung (engl. analog fault modelling, AFM), die eine sehr gute Verhaltensvoraussage bei Fehlern bietet. AFM wurde bereits vor Jahrzehnten als Methode vorgeschlagen, blieb jedoch wegen des enormen Aufwandes an Rechenkapazität ein akademisches Forschungsfeld. Der Rechenaufwand ist für die Kombination von AFM und EDA-basierter Schaltungsfehlersimulation wegen dem Simulationsaufwand (d.h. Anzahl der Simulationsläufe) als hoch anzusehen. Für den Einsatz der Fehlersimulation für praktische Anwendungen muss auch bei dem heutigen Stand der verfügbaren Rechenleistung der Rechenaufwand reduziert werden. Die Simulation von mehreren Fehlerarten pro Komponente und/oder die Einbeziehung von Mehrfachfehlern (zeitgleich) ist oftmals nur möglich, wenn der Simulationsaufwand reduziert werden kann, so dass eine hinreichend schnelle Untersuchung des Modells einer analogen Schaltung und ihrer Fehler erreicht wird.

Der Einsatz eines Schaltungssimulators mit Fehlerinjektion kann aus mehreren Gründen erfolgen. Dazu gehören insbesondere die Bestimmung des Fehlererkennungsgrads, die Bestätigung der Fehlertoleranz eines Systems und die Quantifizierung seiner Zuverlässigkeit. Zur Überprüfung (Verifikation) eines Schaltungsentwurfes ist die Feststellung, ob

eine vorgegebene Zuverlässigkeit bzw. ein vorgegebener Fehlertoleranzgrad erreicht wird, ebenfalls notwendig, weil ein Entwurf nur selten perfekt ist und der Schaltungsentwickler üblicherweise nicht im Voraus wissen kann, welche Fehlerwirkungen und Ausfallarten eintreten können. Ein Fehlersimulator unterstützt die Prüfung des Schaltungsentwurfes auf Fehlertoleranz und das Quantifizieren der Zuverlässigkeit anhand eines Zuverlässigkeitsmodelles. Auch kann mit seiner Hilfe festgestellt werden, ob alle Fehler, die ein gegebenes Fehlermodell definiert, erkannt werden (Rakowsky und Richardson 2001). Das Ergebnis einer solchen Prüfung wird durch den Fehlererkennungsgrad (FG) (engl. fault coverage, FC) in Prozent ausgedrückt. Der FG kann für die Bestimmung der Zuverlässigkeit nur dann aussagekräftig herangezogen werden, wenn alle Fehler im untersuchten System gleich wahrscheinlich sind.

$$FG = \frac{Anzahl\ der\ erkannten\ Fehler}{Gesamtanzahl\ der\ injizierten\ Fehler}\ [\%]$$

Fehlertoleranz (engl. fault tolerance) ist die Fähigkeit eines Systems, auch mit einer begrenzten Anzahl fehlerhafter Komponenten (Subsysteme) seine Spezifikation zu erfüllen. Die Fehlertoleranz kann durch einen Fehlersimulator überprüft werden. Durch die Toleranz von einzelnen Fehlern wird die Zuverlässigkeit einer Schaltung verbessert. Injizierte Fehler, die nach Prüfen aller festgelegten Spezifikationen als nicht spezifikationsverletzend eingestuft werden, können auch als toleriert betrachtet werden – obwohl sie sich einfach nicht auswirken und damit ohne negative Folgen bleiben. In Digitalschaltungen spielen solche folgenlose Fehler eine große Rolle, beispielsweise die Veränderung eines Registerbits unmittelbar bevor das Register beschrieben wird. In reinen Analogschaltungen treten folgenlose Fehler dagegen seltener auf, können aber vorkommen – meist in der Form, dass sie nur geringfügige Folgen nach sich ziehen wie etwa die Verschlechterung des Frequenzgangs eines Verstärkers.

Von einer „echten“ Tolerierung eines Fehlers wird dagegen gesprochen, wenn trotz Auftreten des Fehlers und trotz eingetretenen Auswirkungen die Spezifikationen des Systems eingehalten werden. Dabei kann die Spezifikation vollständige Funktionsfähigkeit oder, sofern es die Anwendung zulässt, nur den Wechsel in einen sicheren Zustand verlangen. Damit die Fehlertoleranz gelingt, müssen Methoden der Fehlerdiagnose und Datenanalyse sowie geeignete Fehlerbehandlungsmaßnahmen angewendet

werden. Die Fehlerdiagnose ist eine Aktivität zur Erkennung, Lokalisierung und Ursachenfeststellung von Fehlern (Rakowsky und Richardson 2001).

Die Fehlermodelle für analoge Schaltungen sind in Kapitel 3.2 und 3.3 und die konkrete Implementierung einer Fehlersimulation in Kapitel 4.2.6 zu finden.

1.2 NOTWENDIGKEIT DER ZUVERLÄSSIGKEITSBEWERTUNG

Die Zuverlässigkeit eines technischen Systems und die Bewertung der Zuverlässigkeit ist eine Thematik, mit der sich die Wissenschaft schon seit über 70 Jahren befassen. Die Zuverlässigkeit ist bereits seit Beginn der ersten konstruierten elektrotechnischen Maschinen (z.B. Dynamomaschine von Werner von Siemens (1867)) von Bedeutung. Die Zuverlässigkeit und die dazugehörigen Kenngrößen sichern das Nutzervertrauen dahingehend, dass die spezifizierte Leistung während des Betriebes erbracht wird und es keine katastrophalen Folgen für den Nutzer und die Umwelt gibt. Mit der Zeit wurden immer bessere und robustere Analysemöglichkeiten von Fehlern eines Systems entwickelt. Mit der steigenden Rechenkapazität wurden umfangreichere Software-Tools zur Bewertung der Zuverlässigkeit entwickelt.

Zuverlässigkeit ist besonders dann wichtig, wenn ein Ausfall einer Komponente bzw. eines Systems einen unverhältnismäßig hohen Schaden an Mensch und/oder der Umwelt verursacht. In diesem Fall sollte alles Erdenkliche getan werden, um den Schaden zu verhindern. Bei Produkten aus der Massenfertigung oder nicht sicherheitskritischen Produkten bzw. Dienstleistungen ist es unter Umständen nicht von hoher Relevanz, dass alle Systeme einwandfrei funktionieren, sofern deren Ausfälle keine kritischen Folgen haben.

Wir benötigen eine genaue Definition dafür, wie gut die Zuverlässigkeit eines betrachteten Systems sein muss. Um die gewünschte Zuverlässigkeit zu erhalten, ist es wichtig, sich mit der breiten Palette an Methoden zur Verbesserung der Zuverlässigkeit und zur Bewertung der erreichten Verbesserung auseinander zu setzen und diese systematisch und umfassend zu berücksichtigen. Wenn ein Schwachpunkt (z.B. eine ausfallträchtige Komponente) übersehen wird und es deshalb zu einem Systemausfall kommt, spielt es keine Rolle mehr, wie zuverlässig das restliche System funktioniert hat.

Ein großes Problem in der Entwicklung von fehlertoleranten Systemen ist die Bestimmung der erreichten Zuverlässigkeitseigenschaften. Im Gegensatz zur erbrachten Systemleistung, welche durch den Einsatz von Benchmark-Programmen bewertet werden kann, kann der Grad der erreichten Fehlertoleranz und Zuverlässigkeit nicht in einer solchen Weise bestimmt werden. Eine direkte Messmöglichkeit von Systemausfällen und Fehlerursachen bestünde darin, die Systeme über Jahre im Testbetrieb zu halten, bis das Fehlerverhalten untersucht werden kann. Diesen

„Luxus“, die Systeme über einen hinreichend langen Zeitraum, d.h. über Monate und Jahre, im Testbetrieb laufen zu lassen, kann man sich in der Regel nicht erlauben. Die allgemein akzeptierte Lösung für dieses Problem besteht darin, die Auswirkungen von Fehlern (z.B. durch Fehlerinjektion) in einem Simulationsmodell oder Prototypen zu untersuchen und das Verhalten des Systems unter Fehlerinjektion zu beobachten. Diese Vorgehensweise ist sehr flexibel, aber zeitaufwendig. Auf der anderen Seite wäre es sehr viel schwieriger, genau spezifizierte (möglichst realistische) Fehler in einen Prototyp zu injizieren und dann durch Messungen die Auswirkungen von Störungen zu beobachten.

Wenn nach dem Nutzen einer Zuverlässigkeitsanalyse gefragt wird, so sind die die folgenden Antworten zielführend. Zuverlässigkeitsanalyse kann verwendet werden für

- die Beurteilung, ob Zuverlässigkeitsziele z.B. eine vorgegebene MTBF erreichbar sind
- die Analyse alternativer Entwürfe
- die Identifikation möglicher Entwurfsschwächen
- die Bereitstellung von Daten für die Bewertung der Systemzuverlässigkeit und Verfügbarkeit
- die Vorhersage von Fehlerraten der Komponenten und des Systems
- usw.

Um eine Zuverlässigkeitsanalyse durchführen zu können, müssen anhand der folgenden Schritte genügend Information zusammengetragen werden:

(I) Definition des zu analysierenden Systems
(II) Verstehen des Gesamtsystems durch die Analyse der Komponentenstruktur
(III) Bestimmen der Betriebsbedingungen, wie Betriebstemperatur oder anderer Stressparameter
(IV) Bestimmen der Zustände bei Auftritt von Ausfällen
(V) Bestimmen der Systemstruktur für die Zuverlässigkeitsmodellierung

Die Informationen sind besonders zu Beginn der Entwicklung nicht verfügbar. Außerdem ist die Fülle der Daten auf der unteren Abstraktionsebene (Komponentenebene) nur rechnergestützt zu verarbeiten. In dieser

Arbeit wird ein Fehlersimulator verwendet, um die Systemfunktion durch Fehlerinjektion bei analogen Schaltungen zu bestimmen.

Im Folgenden wird die Frage aufgeworfen, warum eine Fehlersimulation für analoge Schaltungen benötigt wird, und welche Begrenzungen der Möglichkeiten dabei bestehen. Durch Nutzung der automatischen Fehlersimulation und Generierung von Fehlerlisten kann dafür gesorgt werden, dass der Aufwand und die Dauer einer Fehleranalyse reduziert werden. Analoger Schaltungsentwurf wird meist als ein nicht automatisierter Entwurfsprozess durchgeführt. Ein Entwickler von analogen Schaltkreisen könnte sich die folgenden Fragen stellen: Was passiert, wenn sich der Widerstandwert verdoppelt? Was passiert, wenn einzelne Komponenten mit einer bestimmenden dominanten Fehlerart ausfallen? Wie könnten sich die Parameterwerte der einzelnen Komponenten unter verschiedenen Umweltbedingungen verändern? Der in dieser Arbeit entwickelte analoge Fehlersimulator stellt eine Erweiterung der EDA-Entwicklungssoftware dar, die eine automatisierte Wenn-Dann-Analyse durchführt. Diese Wenn-Dann-Fragen können als Variationen des entwickelten Schaltungsentwurfes aufgefasst werden, die durch eine Fehlersimulation bewertet werden können. Daher ist eine effektive und effiziente Fehlersimulation sehr sinnvoll für den analogen Schaltungsentwurf, der sicherstellt, dass das System seine Funktion ausführt und dabei die Spezifikation erfüllt. Wie bereits erwähnt, nützt die Fehlersimulation ebenfalls für die Bewertung der Zuverlässigkeit. Der Preis für die Fehlersimulation besteht im hohen Rechenaufwand, weil die Zahl der benötigten Simulationsdurchläufe mit jedem Fehler und jeder Fehlerkombination ansteigt. Ohne besondere Maßnahmen ist die Anzahl der Fehlersimulationen im schlechtesten Fall gleich der Anzahl der zu untersuchenden Fehler (-kombinationen). Im Gegensatz zu digitalen Schaltungen bewirken Variationen im Fertigungsprozess, ungewisse Signalbedingungen (Rauschen) bzw. ungenaue Temperaturbereiche, dass die Parameterwerte analoger Schaltungen und Bauelemente in einem weiten Bereich streuen. Das Verhalten der analogen Schaltung kann sehr sensible auf Veränderungen der Parameterwerte reagieren. Daher kann die notwendige Anzahl der Simulationsläufe sehr hoch sein. Diese gilt es durch die Entwicklung spezieller Techniken und Methoden zu reduzieren.

1.3 ZUVERLÄSSIGE SYSTEME IM BETRIEB

Unsere Abhängigkeit von unterstützenden Systemen (z.B. Computer-Systemen) bzw. Steuergeräten ist hoch und spielt eine wichtige Rolle in unserem täglichen Leben. Es gibt Bereiche bzw. Branchen, wo analoge Systeme bzw. analoge Bestandteile in digitalen Systemen bzw. Steuergeräten eine untergeordnete Rolle spielen. Zu den Branchen mit überwiegend digitalen Systemen können Bank- bzw. Finanzdienstleistungen, Telekommunikation oder Steuerungen in der industriellen Produktion gezählt werden. Zu den Branchen, wo die analogen Komponenten (z.B. Sensoren) und Systeme von Bedeutung sind, zählen beispielsweise das Verkehrswesen, die medizinische Versorgung oder verschiedene militärische Systeme. Es existieren noch weitere Bereiche, in denen wir eine Dienstleistung von verarbeitenden Systemen erhalten, ohne dies direkt zu bemerken. Die tägliche Nutzung von Heizung, Küchengeräten, Fernsehen, Auto oder intelligenter Haustechnik (engl. Smart-Home), sind ohne informationsverarbeitende Systeme mit analogen Komponenten nicht realisierbar. Eine erweiterte Übersicht über die besonders kritischen Bereiche (Gefahr für Menschen oder monetäre Verluste) wird in (Knight 2012, S. 7–8) gegeben.

Bei Bank und Finanzdienstleistungen überwachen Computersysteme die Girokonten, überweisen Geld zwischen Banken bzw. Ländergrenzen, stellen Dienstleistungen wie Geldautomaten bzw. Kreditkartenzahlungen bereit, bieten einfachen Zugriff auf Markthandel bzw. Devisenwechsel und bieten noch viele weitere Finanztransaktionen an. In der Telekommunikation müssen alle Telekommunikationssysteme, die wir nutzen, zuverlässig arbeiten. Angefangen bei den kabelgebundenen Telefonen bis hin zu kabellosen mobilen Diensten mit Mobiltelefonen oder Tablet Computern. In der Produktionsfertigung setzen die Hersteller von Waren nicht erst seit der Industrie-4.0-Revolution auf Systeme zur Steuerung von Fließband, Lieferkettenmanagement, Betriebsmittel und Personal, Terminplanung oder auch Finanztransaktionen und Buchhaltung.

Im Verkehrswesen nutzen u.a. kommerzielle Flugzeuge oder die Flugsicherungssysteme umfassende Computersysteme zur Steuerung und Überwachung des Betriebes. Autos und die Fahrer sind abhängig von unterstützenden Systemen wie Motorsteuerung, ABS, ESP und zum Teil auch Unterhaltungssystemen, die analoge Elektronik-Komponenten enthalten. Es ist wichtig, deren Fehler zu genauer betrachten, weil ca. 80% der defekten Steuergeräte bei Automobilen auf analoge Bestandteile zurückzuführen sin (Sunter 2016). Züge können heutzutage nicht ohne

hochentwickelte elektronische Systeme bedient oder gesteuert werden. Die medizinische Versorgung und viele medizinische Leistungen hängen von der richtigen Funktion der Steuereinheiten ab. Dazu gehören bspw. Pumpen für Medikamente, Geräte für Strahlentherapie und chirurgische Roboter. Aber auch Informationssysteme wie Patientendokumentation oder Warenbestandskontrollen von Medikamenten spielen eine wichtige Rolle. In der militärischen Verteidigung ist die Verwendung umfangreicher Systeme in der Führung, Waffenleitung, Steuerung von Fluggeräten, Schiffen und Raketen oder auch Fertigungssteuerung von großer Relevanz. In all den genannten Beispielen muss man sich verstärkt den Fehlern der jeweils vorhandenen Analog-Komponenten und -Schaltungen widmen, weil sie einen signifikanten Anteil der Systemausfälle verursachen.

Im Folgenden werden einige Beispiele zu katastrophalen Unfällen in Verbindung mit Computersystemen bzw. Fehlern in analogen Systemen bzw. Komponenten wiedergegeben. Einer der häufig genannten Unfälle ist der **Ariane-5-Startfehler** vom 4. Juni 1996. Knapp 40 Sekunden nach dem Start erfolgte eine Selbstzerstörung wegen einer fehlerhaften Softwareimplementierung. Der materielle Schaden belief sich auf ca. 500 Mil. US-Dollar. Es stellte sich kurze Zeit später heraus, dass eine einfache Konvertierung von einer 64-Bit-Gleitkommazahl zu einer ganzen 16-Bit-Zahl scheiterte und zum Ausfall des redundanten Navigationssystem geführt hatte (James Gleick 1996). Bei der Konvertierung hatten die Entwickler nicht angenommen, dass die konvertierte Zahl größer als 16 Bit werden könnte. Dadurch wurden andere Daten im Speicher überschrieben, was wiederum weitere Systeme zum Fehlverhalten zwang. Weitere Untersuchungen haben ergeben, dass die analogen Ausgangssignale der Beschleunigungssensoren im Trägheitsbezugssystem (engl. Inertial Reference System, SRI) sehr wohl über einen Testaufbau hätten überprüft werden können, womit auch der Fehler vermieden worden wäre (Lions 1996). Insgesamt ist der Fehler als ein Entwurfsfehler bei Benutzung eines Analogsensors einzustufen. Bei dem genannten Unfall der Ariane 5 sind nur materielle Verluste, aber keine Folgen für die Unversehrtheit von Menschen entstanden.

Es gibt darüber hinaus auch eine Vielzahl von Fällen, wo durch Versagen von Systemkomponenten (z.B. Computer), durch Fehler von menschlichen Bedienern oder auch durch falsche Managemententscheidungen, Menschen oder ganze Bevölkerungsgruppen zu Schaden gekommen sind.

Es gibt eine Menge von bekannten Beispielen dafür. Hierzu liefert (Redmill und Dale 1997, S. 7–14) eine gute Übersicht über Ausfälle (menschliches und technisches Versagen) mit kritischen bzw. überlebenswichtigen Folgen für Menschen und Umwelt.

Ein häufig zitierter Ausfall bei medizinischen Systemen war ein computergesteuertes Strahlungstherapiegerät genannt **Therac-25**. Eingesetzt wurde es zwischen den Jahren 1986 und 1987. Dabei starben zwei Menschen und weitere Patienten wurden durch die Strahlungstherapie verletzt. Durch eine Fehlfunktion in der Software wurde die Strahlungstherapie um einen Faktor 100 zu hoch dosiert. Untersuchungen haben sowohl Entwurfsfehler in den hardwarebasierten Sicherungssystemen der analog-digitalen und digital-analog Subroutinen (engl. analog-to-digital limit checking), als auch explizite Fehler in der Steuerungssoftware aufgezeigt, die zu den genannten Fehlfunktionen führten (Leveson und Turner 1993; Leveson 1995).

Im Folgenden wird ein aktuelles Beispiel für analoge/gemischte Schaltungen exemplarisch hervorgehoben, nämlich der Absturz des **Fluges QZ8501** (2014). Eine fehlerhafte Lötstelle führte zu einer zumindest teilweise unterbrochenen Stromzufuhr am Ruderausschlagbegrenzer (engl. rudder travel limiter unit, RTLU). Eine entsprechende Warnmeldung wurde vier Mal im Cockpit angezeigt. Das RTLU ist ein elektromechanischer Aktuator, welcher den Ausschlag des Ruders auf einen vorgegebenen Höchstwert limitiert. Eine Lösung des technischen Problems war nicht erfolgreich. Zum Schluss haben die Piloten den Neustart des FAC Systems (engl. Flight Augmentation Computer) veranlasst, der das RTLU steuert. Durch die auftretenden Fehler hat sich auch der Autopilot ausgeschaltet und die Flugsteuerungslogik schaltete einen weiteren Modus (Alternate Law) ab. Diese Fehler sorgten dann insgesamt dafür, dass das Flugzeug von den Piloten nicht mehr in einen kontrollierten Flugzustand gebracht werden konnte. Die Ergebnisse der Untersuchung (KNKT 2015) belegten, dass die thermischen Zyklen ausschlaggebend für die fehlerhafte Lötstelle waren. D.h. das regelmäßige An/Ausschalten der Steuereinheit und die zusätzlichen Umweltbedingungen am Boden und in der Luft sorgten für eine Materialermüdung an der Lötstelle.

Die oben genannten Beispiele zeigen deutlich, dass Fehler und Ausfälle, ganz unabhängig davon, wie zuverlässig ein System aufgebaut wird, auftreten können. Wichtig ist bereits zu Beginn zu wissen, wie zuverlässig ein System zu entwickeln ist. Es muss klar kommuniziert werden welche Zuverlässigkeit erwartet wird und mit welcher Wahrscheinlichkeit die

unerwünschten Konsequenzen höchstens auftreten dürfen. Ebenfalls muss vor Betriebsbeginn sichergestellt werden, dass diese Wahrscheinlichkeit im späteren Betrieb nicht überschritten wird. Die gewünschte Wahrscheinlichkeit muss daher messbar sein, damit die bekannten Methoden der Zuverlässigkeitsbewertung dementsprechend umgesetzt werden können. Wie zuvor bereits genannt, benötigen wir zuverlässige Systeme, auf die wir uns verlassen (engl. depend) können.

1.4 ZIELSETZUNG DER ARBEIT

Die vorliegende Arbeit stellt neue Lösungsansätze für effizientere und dennoch hinreichend akkurate Simulation von Mehrfachfehlern analoger und gemischter Schaltungen vor. Die dafür benötigten Fehlermodelle für Einfachfehler und Mehrfachfehler werden mit einer objektorientierten Skriptsprache beschrieben, welches ebenfalls einen neuen Ansatz in der Fehlersimulation analoger Schaltungen darstellt. Durch einen Schaltungssimulator kann nicht nur die Funktionalität der Schaltungskonstruktion überprüft werden. In Verbindung mit der Fehlerinjektion kann die Qualität der Funktionserbringung oder deren Veränderung unter Fehlerauftritt untersucht werden. Das komponentenbezogene Aufstellen der Systemfunktion kann bereits bei kleinen Schaltungen nicht mehr per Hand erfolgen, sondern kann nur durch eine automatisierte Vorgehensweise bewerkstelligt werden.
Folgende Schlüsselprobleme und Beiträge wurden in dieser Dissertation analysiert bzw. neu entwickelt:

- Das Hauptziel dieser Arbeit ist die effiziente Fehlersimulation von Mehrfachfehlern bei analogen Signalen. Der Fehlerklassen-Algorithmus (FKA) soll den Simulationsaufwand im Vergleich zu bisherigen Ansätzen reduzieren, indem er die Anzahl der notwendigen Simulationsläufe unter Fehlerinjektion verringert und die Simulation dadurch erheblich beschleunigt (siehe Kapitel 4.1). Insbesondere soll das Verfahren für Schaltungen nützlich sein, die einen Anteil der möglichen Einfach- und Mehrfachfehler tolerieren können. Für Schaltungen, die ausschließlich Einfachfehler tolerieren, wird dagegen keine Verbesserung erwartet. Die neue Methode soll die Anzahl der Simulationsläufe für Dreifachfehler und höhere Fehleranzahl deutlich reduzieren. Die weggelassenen Simulationen sollen durch Berechnung ersetzt werden, die eine Näherungslösung liefern.
- Ein weiteres Ziel ist der Entwurf und die prototypische Implementierung eines auf Schaltungssimulation basierenden Fehlerinjektors für Schaltungen mit analogen und gemischten Signalen. Die geforderten besonderen Eigenschaften werden in den folgenden beiden Punkten festgehalten (Siehe auch Kapitel 4.2.3).
  - Die Fehlermodellierung erfolgt mit einer skriptbasierten Fehlerbeschreibung und wird objektbasiert auf die Komponenten der Schaltung angewendet.
  - Die Fehlerinjektion und Fehlermodellierung berücksichtigt neben den Einfachfehlern auch Mehrfachfehler. Auch können mehrere Fehlerarten pro Bauelement berücksichtigt werden.

1.5 AUFBAU DER ARBEIT

Die vorliegende Arbeit ist in die folgenden Abschnitte strukturiert. Kapitel 1 beinhaltet eine Einführung in das Prüfen von analogen Schaltungen, die Motivation zur Fehlersimulation und beschreibt die Zielsetzung der Arbeit. In Kapitel 2 werden zunächst die Grundbegriffe eingeführt, die Problemstellung der Fehlersimulation erläutert und eine detaillierte Einführung in die Bewertung der Zuverlässigkeit vorgestellt. In Kapitel 3 werden der aktuelle Stand der analogen Fehlersimulation, der Fehlermodellierung und bisherige Arbeiten über analogen Fehlersimulatoren im Detail beschrieben. Die neuen Ansätze dieser Arbeit sind Gegenstand von Kapitel 4. In Abschnitt 4.1 wird ein neues Verfahren zur Beschleunigung der analogen Fehlersimulation, nämlich der Fehlerklassen-Algorithmus (FKA) ausführlich dargelegt. In Abschnitt 4.2 wird ein neuartiger Fehlerinjektor für analoge Schaltungen präsentiert. Ebenso findet sich dort der neue Ansatz der skriptbasierten Fehlerbeschreibung für diskrete Bauelemente. In Kapitel 5 wird der skriptbasierte Ansatz der Fehlersimulation anhand einer Reihe von Beispielschaltungen konkret dargelegt und quantitativ bewertet. Den Abschluss der Arbeit bildet Kapitel 6 mit der Zusammenfassung der vorgestellten Ansätze und der durchgeführten Experimente. In Kapitel 7 wird ein Ausblick auf mögliche Erweiterungen des entwickelten effizienten Ansatzes der Fehlersimulation gegeben. In den Anhängen sind die Grundlagen zur Zuverlässigkeitstheorie, ein erweitertes Fehlermodell, Details zu den experimentell untersuchten Analogschaltungen und die Beschreibung der beiliegenden CD-ROM zu finden.

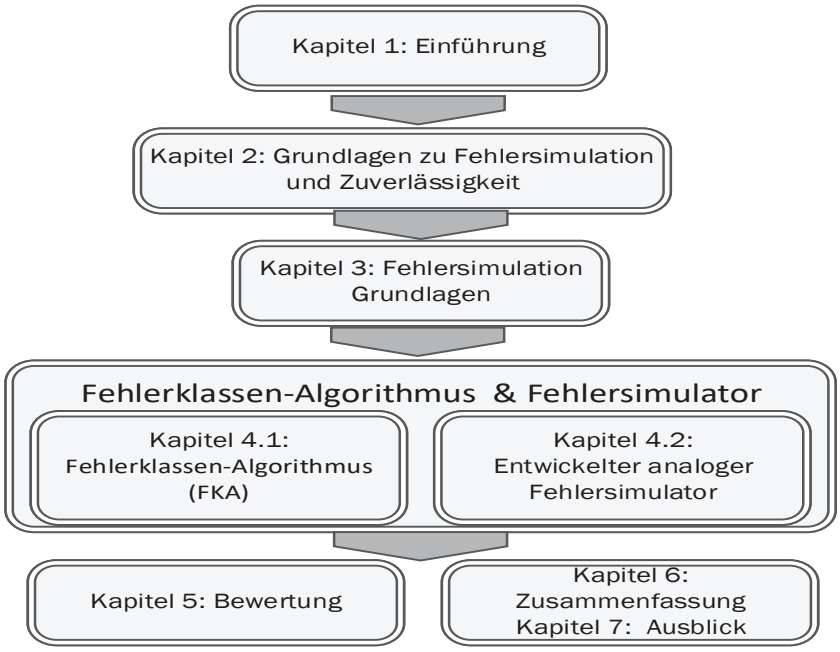


Abbildung 1. Gliederung der Arbeit

2 Grundlagen der Fehlersimulation und Zuverlässigkeit

Das Ziel dieses Kapitels besteht darin, eine genaue Charakterisierung der verwendeten Begriffe und Konzepte zu geben. Der Begriff der Zuverlässigkeit (engl. dependability) wird nach (Avizienis et al. 2004a) als ein globales Konzept eingeführt und die relevanten quantitativen Kenngrößen wie Verfügbarkeit, Funktionsfähigkeit, Sicherheit usw. diesem zugeordnet. Des Weiteren werden die Grundlagen der Fehlerinjektion und die relevanten Modelle zur Zuverlässigkeitsbewertung im Detail beschrieben.

2.1 Definitionen und Elemente von Zuverlässigkeit

System und Komponenten

Ein System wird durch (DIN EN 60300-1: 2015-01) als eine Entität definiert, die mit anderen Entitäten in Wechselbeziehung steht und mit diesen zusammen eine Anforderung erfüllt. Das bedeutet auch, dass ein System mit anderen Systemen interagiert. Zwischen einem gegebenen System und anderen Systemen wird eine reale oder abstrakte Systemgrenze gezogen, so dass die anderen Systeme als die Umwelt des gegebenen Systems aufgefasst werden können. Die Systemstruktur muss die gewünschten bzw. spezifizierten Funktionsanforderungen realisieren.

Ein Systemaufbau kann hierarchisch sein, beispielsweise: System - Subsystem - Komponente / Bauelement. Unter Strukturgesichtspunkten wird ein System aus einer Reihe von Komponenten bzw. Einheiten zusammengesetzt, die miteinander verbunden sind und aufeinander einwirken. Dabei kann eine Komponente wiederum ein eigenes System bilden. Dieser rekursive Ansatz endet, wenn eine Komponente als atomar betrachtet wird, d.h. wenn keine weitere innenliegende Struktur mehr besteht oder nicht mehr von Belang ist. Beispielsweise kann eine einfache (analoge) Schaltung als ein System betrachtet werden. Die untereinander verbundenen Bauelemente (z. B. Widerstände, Kondensatoren usw.) sind dann die atomaren Komponenten. Folglich besteht der Zustand des Systems aus der Kombination der Zustände der atomaren Komponenten. Um die Zuverlässigkeit eines Systems zu beurteilen macht es daher Sinn, die möglichen Fehlzustände und deren stochastisches Auftrittsverhalten der miteinander verbundenen Komponenten zu analysieren und die Systemfunktion anhand der Komponentenfunktionen aufzustellen.

Kenngrößen der Zuverlässigkeit

Die Zuverlässigkeit eines Systems wird in (DIN EN 60300-1: 2015-01) definiert als die Fähigkeit, unter gegebenen Bedingungen und für ein gegebenes Zeitintervall den geforderten Dienst (engl. Service) ohne Ausfall zu erbringen. Dabei gehören zu den Bedingungen auch die folgenden Gesichtspunkte, die als Voraussetzungen bzw. Anforderungen für die Funktionsfähigkeit des Systems betrachtet werden können, beispielsweise verschiedene Umweltbedingungen, das Auslastungsniveau und/oder die notwendigen Vorgehensweisen zur Instandhaltung. Das Zeitintervall wird in Abhängigkeit vom betrachteten System und seiner Verwendung ausgedrückt, beispielsweise Kalenderzeit, Betriebszyklen, Laufleistung usw. Zuverlässigkeit in ihrer Gesamtform kann anhand der folgenden Kenngrößen erfasst werden. Die wichtigsten Kenngrößen, d.h. Kriterien für die Entscheidung, ob ein erbrachter Dienst zuverlässig ist, sind die Verfügbarkeit (engl. availability), die Überlebenswahrscheinlichkeit (engl. reliability), die Wartbarkeit bzw. Instandhaltbarkeit (engl. maintainability), die Integrität (engl. integrity), die Betriebssicherheit (engl. safety) und die Vertraulichkeit (engl. confidentiality). Neben den quantitativ definierten Kenngrößen der Zuverlässigkeit ist auch eine qualitative und allgemeinere Definition der Zuverlässigkeit erwähnenswert. Diese definiert Zuverlässigkeit als eine Fähigkeit eines Systems, einen Dienst zu erbringen, dem ein Nutzer vertrauen kann. Die qualitative Definition betont das Bedürfnis nach der Rechtfertigung für das Vertrauen in ein System (Avizienis et al. 2004a; Knight 2012).

Weitere Definitionen der Zuverlässigkeit in aktuellen DIN-Normen weichen zum Teil von der oben gegebenen Definition ab. So definiert (DIN EN 62741:2015-10) die Zuverlässigkeit einer Einheit, als „die Fähigkeit zu funktionieren, wie und wann gefordert“. Dazu wird auch angemerkt, dass es sich bei dem Begriff Zuverlässigkeit um einen „Gattungsbegriff für die zeitbezogenen Qualitätsmerkmale“ handelt, wobei die Verfügbarkeit, Funktionsfähigkeit, Wiederherstellbarkeit, Instandhaltbarkeit und Instandhaltungsbereitschaft und in manchen Fällen auch die Haltbarkeit und Sicherheit dazu gehören. Werden beide Definitionen der Zuverlässigkeit miteinander verglichen, so kann man schlussfolgern, dass beide Definitionen zum Großteil dieselben Ziele verfolgen und auf die Gefährdung durch Fehler gleichermaßen aufmerksam machen. Abschließend kann man erwähnen, dass es unausweichlich zum Auftreten oder Vorhandensein von Fehlern kommen kann und daher jedes betrachtete System nicht vollkommen verfügbar, funktionsfähig oder sicher ist.

Im Folgenden werden die wichtigsten Kenngrößen der Zuverlässigkeit definiert (IEC 50 (191)1995; Rakowsky und Richardson 2001):

- **Verfügbarkeit:** Die Wahrscheinlichkeit, dass ein (Sub-) System zu einem gegebenen Zeitpunkt in einem funktionsfähigen Zustand anzutreffen ist.
- **Überlebenswahrscheinlichkeit:** Die Wahrscheinlichkeit, dass das untersuchte System bzw. die Einheit während eines bestimmten Zeitraumes (Zeitintervall: $[0, t]$) einen Grenzzustand (Übergang vom fehlerfreien (ff) in den fehlerhaften (fh) Zustand) nicht überschreiten wird. Der angelsächsische Begriff „reliability" hat im deutschen zwei unterschiedliche Bedeutungen. Die erste und quantitative Bedeutung ist die Überlebenswahrscheinlichkeit. Außerdem besitzt er auch die qualitative Bedeutung der Funktionsfähigkeit, d.h. der Fähigkeit eines Systems, eine geforderte Funktion unter gegebenen Anwendungsbedingungen für ein gegebenes Zeitintervall zu erfüllen.
- **Wartbarkeit:** Die Kombination aller technischen und administrativen Maßnahmen, mit denen eine Einheit im funktionsfähigen Zustand erhalten oder in ihn zurückversetzt werden kann, um den geforderten Dienst zu erfüllen.
- **Betriebssicherheit:** Begrenzung des Risikos von Personen- oder Sachschäden auf einen annehmbaren Wert. Abwesenheit von (nicht) katastrophalen Folgen für den Benutzer und die Umwelt.
- **Vertraulichkeit** Keine unbefugte Weitergabe von Informationen.
- **Integrität:** Schutz vor unautorisierten Systemmodifikationen und schreibendem Zugriff auf Informationen. Im Unterschied zur Vertraulichkeit liegt der Fokus auf der nicht bevollmächtigten Veränderung der gespeicherten Informationen.

Einige Hinweise zu Sicherheitsaspekten (insbesondere IT-Sicherheit), die bei den Kenngrößen von Zuverlässigkeit die IT-Sicherheit (engl. security) nicht erfasst werden. IT-Sicherheit wird aus einer Zusammensetzung von den Begriffen Verfügbarkeit, Integrität und Vertraulichkeit definiert. Es existiert keine allgemeingültige Definition von sicheren Systemen bzw. davon, wie ein System sicher gemacht werden kann. So müssen manche Systeme hauptsächlich vor unautorisiertem Zugriff geschützt werden, während bei anderen vorwiegend die Verfügbarkeit, d.h. Minimierung bzw. Vermeidung von Dienstausfällen, gewährleistet werden muss. Wenn bei einem System IT-Sicherheit erforderlich ist, so müssen die Risiken

und die Wichtigkeit von Verfügbarkeit, Integrität und Vertraulichkeit separat festgelegt und quantifiziert werden.

Betrachtet man die Kenngrößen der Zuverlässigkeit genauer, stellt man fest, dass Zuverlässigkeit verschiedene Ausprägungen haben kann. Beispielsweise ist ein System zuverlässig verfügbar, wenn es dauerhaft und ohne Unterbrechungen zur Verwendung bereitgestellt wird. Liefert ein System kontinuierlich seine geforderte Leistung, so ist es zuverlässig funktionsfähig. Spielt das zügige wieder in Betrieb nehmen eine wichtige Rolle, so muss es zuverlässig wartbar sein. Wenn die Vermeidung von katastrophalen Folgen im Vordergrund steht, so muss ein System zuverlässig sicher sein.

Fehlerursache, Fehler und Ausfall

Nach der oberen Definition kann ein System nur dann einen Dienst zuverlässig erbringen, wenn das System seine geforderte Funktion korrekt umsetzt. Ein Ausfall oder auch Versagen (engl. failure) ist ein Ereignis, dass zu einer Beendigung der Funktionsfähigkeit einer Komponente oder eines Systems führt. Das Ereignis charakterisiert die Unfähigkeit eines Systems, die festgelegte Leistung zu erbringen. Nach dem Ausfall befindet sich die Komponente bzw. das System in einem Fehlzustand. Das Erbringen eines Dienstes hängt von der bestehenden Systemspezifikation mit ihren Funktionen, Zuständen und Zustandsübergängen ab. Ein Dienstausfall bedeutet, dass ein oder mehrere Zustände der Komponenten von den geforderten Zuständen abweichen. Diese Abweichung wird häufig als Fehler oder auch Störung (engl. error) bezeichnet. Ein Fehler ist daher die Nichtübereinstimmung zwischen gemessenem bzw. beobachtetem Verhalten und dem spezifizierten bzw. korrekten Verhalten. Die Fehlerursache wird im englischen mit „fault“ bezeichnet. Die Fehlerursache muss nicht direkt einen Fehler aktivieren (engl. active fault), sondern kann auch latent bzw. ruhend (engl. dormant fault) sein.

Die Beziehung zwischen Fehlerursache → Fehler → Ausfall → Fehlerursache → usw. kann am besten an einem Beispiel erläutert werden. Nehmen wir eine Schaltung mit einem fehlerbedingt veränderten Widerstandswert an (Fehler, engl. eroneous resistor). Der Wert des betrachteten Widerstandes sollte zwischen 250 und 300 Ohm sein. Aufgrund von Wärmeeinwirkung (Fehlerursache) wird der ohmsche Wert des NTC-Widerstandes stetig erniedrigt. Die Fehlerart wäre in diesem Fall ein Parameterfehler und der Fehler bleibt solange ohne Konsequenz, wie die Schaltung noch innerhalb des definierten Bereiches der Leistungserfüllung bleibt. Der Bereich der Fähigkeit, eine Funktion zu erbringen, wird

als Toleranzbereich bezeichnet. Sobald sich der ohmsche Widerstand auf den Wert von unter 250 Ohm reduziert, wird die Fehlerursache aktiv, da nun der Widerstand außerhalb des spezifizierten Bereiches (250-300 Ohm) liegt. Der angenommene Fehler im Widerstand hat zunächst keine weiteren Folgen, so dass die Schaltung zunächst nicht versagt. Der Fehler kann sich aber womöglich fortpflanzen und an anderen Stellen der Schaltung weitere Fehler verursachen. Bei weiterer Wärmeeinwirkung sinkt in diesem Beispiel der Widerstandswert weiter bis zu einer Grenze von 120 Ohm, was einen Ausfall der Schaltung bewirkt (z.B. die Spannungsversorgung ist außerhalb des Toleranzbereiches). Der Ausfall des Systems kann auch die Fehlerursache für andere Systeme sein bzw. Auswirkungen auf die Umwelt haben. In der folgenden Abbildung 2 wird die Beziehung zwischen den Begriffen (Fehlerursache, Fehler und Ausfall) in Bezug auf das obige Beispiel verdeutlicht (Fehler in einem Bauelement und Ausfall einer Schaltung, deren Funktion die Abgabe einer Spannung zwischen 4 und 5 Volt ist). Für mehr Details der Begriffsabgrenzung siehe (IEC 50 (191)1995) und (Avizienis et al. 2001, S. 13–15).

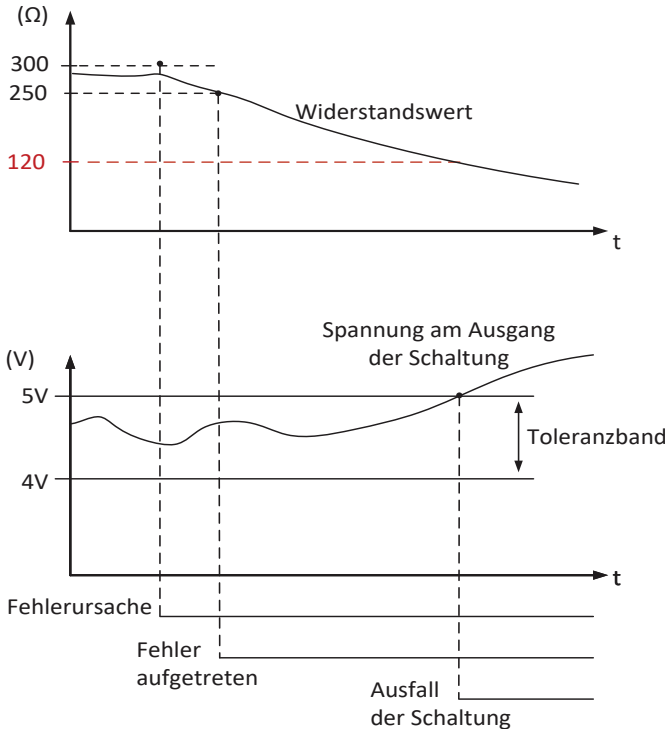


Abbildung 2. Unterscheidung zwischen Fehlerursache, Fehler und Ausfall.

Klassifizierung von Fehlern

Fehlerursache, Fehlerart und Ausfall haben nicht immer den gleichen Charakter und können auf unterschiedliche Art und Weise ausgeprägt sein. So lassen sich Fehler bzw. Fehlerursachen in mehrere Klassen, unter anderem nach der Dauer (permanenter Fehler vs. temporärer Fehler), der Fehlerart (z. B. zufällige Fehler) oder auch nach dem Fehlerursprung (interne Fehler vs. externe Fehler, Entwurfsfehler vs. Betriebsfehler usw.) unterteilen. Physikalische Fehler/Hardware-Fehler, die während der Betriebszeit auftreten, können nach ihrer zeitlichen Fortdauer klassifiziert werden. Permanente Fehler (engl. permanent fault) werden häufig durch einen nicht wiederherstellbaren Komponentenschaden (z.B. internen Kurzschluss) ausgelöst, verursacht durch einen Alterungsprozess, einen fehlerhaften Herstellungsprozess oder durch falsche Bedienung. Eine Wiederherstellung des Systems kann nur durch die Reparatur oder den Austausch der Komponente erfolgen. Temporäre bzw. intermittierende Fehler (engl. temporary or transient fault) werden häufig ausgelöst durch geänderte Umweltbedingungen wie elektromagnetische Strahlung oder Spannungsschwankungen. Fehlerursachen, die bei mehr als einer Komponente einen Fehler verursachen können, werden als zusammenhängende Vielfachfehler (engl. multiple related errors) bezeichnet (Avizienis et al. 2004b, S. 111). Diese Fehler verursachen nicht unbedingt einen permanenten Ausfall bei den Komponenten, können das System aber in einen temporären Fehlzustand bringen. Intermittierende Fehler treten häufiger als permanente Fehler auf (etwa um den Faktor 10) und sind viel schwieriger zu entdecken, da bei intermittierenden Fehlern die Aktivierungsbedingungen (z.B. Effekte der Temperatur-Veränderungen, verändertes Zeitverhalten des Systems) nicht in einfacher Weise reproduzierbar sind.

Die Abbildung 3 klassifiziert Fehler bzw. Fehlerursachen aus acht möglichen Kategorien. Es ergibt sich prinzipiell eine Vielzahl an Kombinationsmöglichkeiten, aber die Fehler lassen sich in drei Hauptkategorien einordnen. Entwicklungsfehler beinhalten alle Fehlerklassen, die während der Entwicklung bzw. des Entwurfs eines Systems vorkommen. Alle Fehlerklassen, die die Hardware betreffen, werden physikalische Fehler (z.B. externe Strahlung oder Rauschen an den Signaleingängen) genannt. Alle Fehler, die mit menschlichem Eingreifen zu tun haben und hauptsächlich während der Nutzungsphase auftreten, werden als Interaktionsfehler (z.B. Konfigurations- und Bedienerfehler) bezeichnet. Selbstverständlich

sind auch Kombinationen der drei Hauptkategorien (Entwurfs-, Interaktionsfehler und physikalische Fehler) möglich. Für weitere Details siehe (Avizienis et al. 2004b, S. 100).

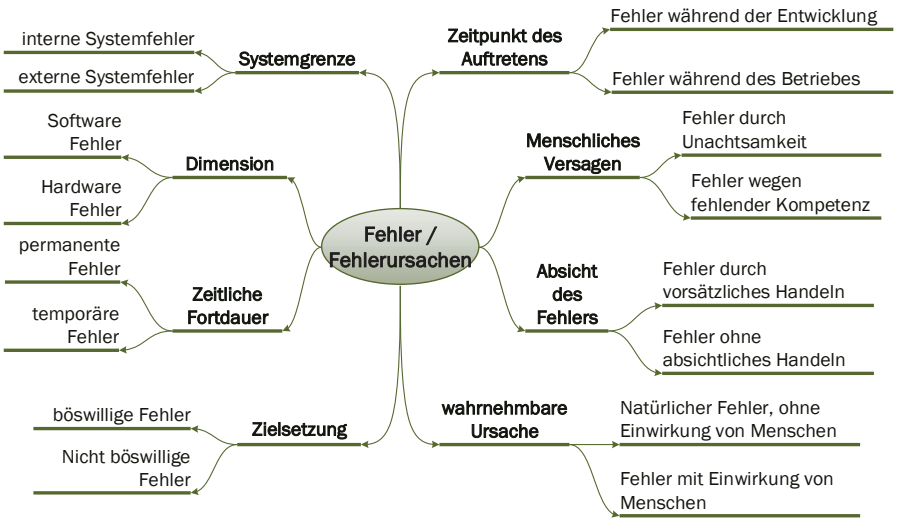


Abbildung 3. Übersicht der Kategorien von Fehlerursachen. Basierend auf (Avizienis et al. 2004b, S. 98).

Der Ausfall von Systemen oder Schaltungen wird durch Fehler in den Komponenten bzw. Bauelementen und den dadurch veränderten Zustand hervorgerufen, die zu Fehlzuständen führen. Je mehr Komponenten miteinander arbeiten, also gemeinsam im System verwendet werden, desto mehr Komponenten (z.B. Bauelemente von Schaltungen) können fehlerhaft sein. Ähnlich verhält es sich mit der Zunahme der Fehlermöglichkeiten bei steigender Komplexität einer Komponente. Die Anzahl der verwendeten Komponenten und die Komplexität heutiger Systeme erhöhen selbstverständlich die Wahrscheinlichkeit, dass eine oder mehrere Komponenten in einen Fehlzustand geraten.

Da die Ausfälle durch Fehler verursacht werden, liegt es in unserem Interesse, Fehler zu verhindern. Die Fehlerverhinderung (engl. fault prevention / avoidance) hat es zum Ziel, durch geplante Konstruktions- bzw. Entwurfsmaßnahmen das Auftreten von Fehlern zu vermeiden bzw. zu reduzieren. Neben der Fehlerverhinderung können noch drei weitere Kategorien von Verfahren zur Erhöhung der Zuverlässigkeit unterschieden werden. Die Fehlertoleranz (engl. fault tolerance) stellt die Fähigkeiten/Mittel (z.B. durch Redundanz) für ein System bereit, um auch bei einer begrenzten Anzahl von fehlerhaften Komponenten die spezifizierte

Funktion zu erbringen. Die Fehlerbeseitigung (engl. error removal) verfügt über Maßnahmen, um aus einem fehlerhaften Zustand heraus wieder einen fehlerfreien Zustand zu erreichen und damit die Fehlerauswirkungen zu beseitigen. Die letzte Kategorie ist die Fehlervorhersage (engl. fault forecasting), die durch Evaluation die Anwesenheit, das Auftreten und die Konsequenzen von Fehlern beschreibt.

Um ein hinreichendes Niveau der geforderten Zuverlässigkeit zu erreichen, wurden in den letzten Dekaden eine Vielzahl von Methoden bzw. Verfahren für jede der genannten Kategorien entwickelt. Die Fehlerprävention und -toleranz beinhaltet alle Verfahren der Zuverlässigkeitsverbesserung, die ein System mit den notwendigen Möglichkeiten ausrüsten, damit es seine spezifizierte Leistung erbringen kann. Fehlerbeseitigung und -vorhersage sind Verfahren, die das Vertrauen in die Fähigkeiten des Systems (die spezifizierte Leistung) stärken.

Fehlermodellierung

Fehlersimulation und insbesondere die Fehlerinjektion kann nur dann effektiv arbeiten, wenn die verwendeten Fehlermodelle so nah wie möglich mit den real vorkommenden z.B. physikalischen Fehlern übereinstimmen. Bedingt durch die Eigenschaften von analogen Schaltungen können Schaltungsparameter unendlich viele Werte annehmen. Daher existieren unendlich viele analoge Fehler und damit auch definierbare Fehlermodelle. Es muss eine optimale Teilmenge ausgewählt werden, damit eine Fehlersimulation unter realistischer Simulationsdauer und notwendiger Genauigkeit durchführbar ist. Diese auf Transistorebene definierte Fehlerliste dient als Input für die sequentielle oder nebenläufige Fehlersimulation. Die Generierung der Fehlerliste ist demnach ein sehr wichtiger Schritt der Fehlersimulation, da hieraus die Simulationsergebnisse und die notwendige Simulationsdauer abhängen. Die modellierten Defekte bzw. Fehlerarten werden mit den für die jeweilige Schaltung notwendigen Test-Stimuli (Versorgungsspannung, zeitabhängige Eingangssignale) simuliert.

Fehlermodelle auf Transistorebene

Vereinfacht formuliert, können die angenommen Fehlerarten (unabhängig davon, ob eine analoge, eine gemischte oder eine digitale Schaltung vorliegt) als eine Kombination von Kurzschlüssen, Unterbrechungen oder Parametervariationen der einzelnen Schaltungsbauelemente dargestellt werden. Die physikalischen Fehlerarten können in zwei grundsätzliche Klassen unterteilt werden. Entweder sind die Fehlerarten aufgrund des Schaltungslayouts entstanden oder durch die Bauelemente (auch

Komponenten genannt) der Schaltung hervorgerufen worden. Entwurfsbasierte Fehlerarten (z.B. Entwurfsfehler, engl. design error) beinhalten Fehler, die erst durch das Layout (bspw. der Anordnung von Komponenten) der Schaltkreise entstehen. D.h. die Bauelemente werden in einer Art und Weise auf der Platine oder auf einem integrierten Schaltkreis angeordnet, die häufig zu Fehlern führt. Beispielsweise entstehen parasitäre Kapazitäten, wenn zwei Leitungen nah genug aneinander liegen und die Frequenz in einem kritischen Bereich liegt. Dies kann ein hinreichendes Indiz für ein mögliches Übersprechen oder gar einen Kurzschluss zwischen diesen beiden Leitungen sein. Schaltungsbasierte Fehlerarten (engl. layout error) sind Fehlerarten, die hauptsächlich an und zwischen den Anschlüssen der passiven/aktiven Bauelemente (z.B. innerhalb von Transistoren, Widerständen, Kondensatoren, Dioden usw.) entstehen. So sind für jedes Bauelement mit drei Anschlüssen, sechs primäre Fehlerarten (drei Kurzschlüsse und drei Unterbrechungen) und für passive Bauelemente mit zwei Anschlüssen zumindest zwei Fehlerarten definierbar. Traditionell werden Unterbrechungen und Kurzschlüsse in Form von Widerständen mit hohen bzw. niedrigen Widerstandswerten definiert. Unterbrechungen haben dann einen Wert von beispielsweise >1 Giga-Ohm und Kurzschlüsse zwischen 0 - 500 Ohm. Da bei realen analogen Schaltungen in seltenen Fällen ideale Kurzschlüsse und Unterbrechungen existieren, müssen realisitsche Parameterwerte gewählt werden. In CMOS-Systemen haben reale Kurzschlüsse nur selten einen Widerstandswert von 0 Ω, am häufigsten jedoch < 500 Ω (Rodriguez-Montanes et al. 1992; Olbrich et al. 1996 1996; Harvey et al. 19941994). Neben den harten Fehlern existieren auch so genannte weiche Fehlerarten bzw. Parameterfehler. Diese verändern (erhöhen oder reduzieren) die physischen Parameterwerte permanent oder für kurze Zeit (z.B. Temperaturabhängig), verursachen ein falsches Verhalten der einzelnen Bauelemente und können zu einer Fehlfunktion der Schaltung führen.

Monotone Systeme und Monotonieeigenschaft

Die Monotonieeigenschaft drückt aus, dass der Ausfall einer Komponente den Systemzustand nicht verbessert, d.h. keinen Übergang von einem fehlerhaften in einen fehlerfreien Systemzustand bewirkt. Umgekehrt ist für ein monotones System nicht anzunehmen, dass es nicht ausfällt, wenn eine defekte Komponente durch eine funktionierende Komponente ersetzt wird. Ein System wird hinsichtlich seiner Komponentenausfälle als monoton bezeichnet, wenn ein Komponentenausfall niemals dazu führen kann, dass ein ausgefallenes System nach dem Auftreten eines Komponentenausfalls wieder funktionsfähig ist. Mit anderen Worten: Durch

Ausfälle kann sich die Funktionsfähigkeit des Systems nur verschlechtern, aber nicht verbessern. (Heidtmann 1997, S. 92–100; Crama und Hammer 2011). Die meisten Systeme aus der Praxis sind monoton. Es existieren jedoch auch nicht monotone Systeme. Unter Betrachtung von Mehrfachfehlern können sich bestimmte Fehler in unterschiedlichen Bauelementen gegenseitig kompensieren. Als ein Beispiel kann ein Spannungsteiler (einer Reihenschaltung von zwei ohmschen Widerständen) aufgeführt werden, dessen Ausgangsspannungen sich in Abhängigkeit von den Teilwiderständen ergibt. Ändern sich beide Widerstände aufgrund durch Parameterfehler in die gleiche Richtung um den gleichen Betrag, wird dieser Fehler durch die Struktur der Reihenschaltung des Spannungsteilers kompensiert und der Spannungspegel zwischen den beiden Widerständen bleibt unverändert.

2.2 PRÜFEN UND DIAGNOSE VON ANALOGEN SCHALTUNGEN

Wie in der Einführung (Kapitel 1) schon angemerkt, wächst die Größe und die Komplexität von integrierten Schaltungen (IC) stetig. Daher wird die Entwicklung von komplexen analogen und gemischten Schaltungen auch in der Zukunft wichtig und notwendig sein. Das Prüfen bzw. die Diagnose-Techniken von digitalen Stromkreisen haben sich sehr gut mit den Komplexitätssteigerungen entwickelt, wohingegen die Diagnosemöglichkeiten von analogen Schaltungen stets weniger weit entwickelt waren und es immer noch sind. Dafür gibt es eine Reihe von Gründen. Beispielsweise ist das Haftfehlermodell (engl. stuck-at) als ein systematischer Ansatz für das Prüfen im digitalen Umfeld weit verbreitet. Dagegen existiert in der analogen Welt kein vergleichbares einfaches Fehlermodell, das physikalische Fehler in einfache Fehlermodelle überführt. Zusätzlich können die Signale in der analogen Schaltung unendlich viele Zustände annehmen und daher auch unendlich viele Fehlerarten verursachen. Hierdurch besteht die Schwierigkeit, eindeutig zu definieren, welche Zustände einer Schaltung bzw. deren Komponenten als fehlerfrei zu betrachten sind und wann diese als fehlerhaft einzustufen sind. Die Fehlerausbreitung erfolgt nicht wie bei digitalen Schaltungen in eine Richtung (nämlich von Gatter-Ausgängen zu nachfolgenden Gatter-Eingängen), sondern in alle mögliche Richtungen (auch entgegengesetzt zur Ausbreitung eines Signals). Ein automatisiertes Verfahren zur vollständigen Prüfung (z.B. von Testsignalen) ist im Gegensatz zu digitalen Schaltungen nicht verfügbar. Moderne DFT-Verfahren (engl. design for testability) für einfacheres und besseres Prüfen von Schaltungen existieren nur im digitalen Umfeld. Ohne diese ATPG-Tools (engl. automatic test pattern generation tools) wäre es nicht möglich, die immensen digitalen Schaltungen der vergangenen Jahrzehnte in guter Qualität und zu geringen Preisen zu fertigen. Die Anforderungen des Prüfens in der analogen Welt können nicht mit denselben analytischen Techniken bewältigt werden. Dagegen haben die genannten vereinfachten Prüfkriterien von digitalen Schaltungen nahezu eine Automatisierung des Prüfens bzw. der Diagnose (Kabisatpathy et al. 2005, S. 4) erreicht.

2.2.1 FEHLERDIAGNOSE BEI ANALOGEN SCHALTUNGEN

Zu Beginn der industriellen Fertigung waren die analogen Bauelemente auf einer Leiterplatte (engl. printed circuit board) befestigt und damit leicht zugänglich. Somit konnten individuelle Bauelemente untersucht und Fehlerursachen identifiziert werden. Später und auch heute noch wurden die Schaltkreise soweit integriert, dass der direkte Zugriff auf die

internen Elemente verwehrt und stattdessen auf die Eingänge und Ausgänge begrenzt ist. Deshalb gestaltet sich die Diagnose und Prüfung schwieriger. Sowohl digitale als auch analoge IC sind heutzutage auf derselben Chipfläche untergebracht. Während die Fehlerdiagnose der digitalen Anteile weitgehend automatisiert ist, ist diese bei analogen Signalen immer noch wenig bis gar nicht automatisiert. Bei digitalen Schaltungen ist die Input-Output-Beziehung über die boolesche Algebra definiert, bei analogen Schaltungen dagegen über teils komplizierte Gleichungssysteme. Folglich kann keine eindeutige Unterscheidung zwischen fehlerfrei und fehlerhaft getroffen werden. So können Schwankungen möglicherweise zu Leistungseinbußen, aber nicht zu einem fehlerhaften Zustand führen. Um trotzdem eine hinreichende Fehlerdiagnose der analogen Komponenten zu ermöglichen, wurde in der Vergangenheit eine Anzahl von Algorithmen und Techniken vorgestellt. Weil kein einheitlicher Standard existiert, sind die Methoden oft spezifisch auf EDA-Tools ausgerichtet. Eine Übersicht für diesbezügliche Verfahren liefern (Roberts et al. 2012; Burns und Roberts 2001; El-Gamal 1990, S. 4–10).

In bisherigen Arbeiten werden zwei Arten analoger Fehlermodellen (siehe Kapitel 3.2) unterschieden. Zum einen sind es harte (bzw. katastrophale) Fehler, die üblicherweise als Kurzschlüsse oder Unterbrechungen modelliert werden. Zum anderen existieren weiche Fehler, welche üblicherweise als ein veränderter Parameterwert simuliert werden. Da es bei analogen Schaltungen theoretisch eine unendliche Anzahl von Parameterzwischenwerten geben kann, muss die Anzahl der Fehler auf eine handhabbare Menge reduziert werden.

2.2.2 ANALOGE SCHALTUNGSSIMULATION

Der Entwurf von Integrierten Schaltungen (IC) wird heutzutage mit Hilfe von EDA-Software durchgeführt. Das Entwerfen mehrerer Alternativen mit Hilfe der Schaltungssimulation ist heutzutage ein gewöhnlicher Entwicklungsprozess. Mit Hilfe von Schaltungssimulationen wird das Prüfen der entworfenen Schaltung vereinfacht. Simulation kann die meisten Schritte des Überprüfens mit realer Hardware nachbilden. Mit Hilfe von mathematischen Modellen wird das Verhalten der untersuchten Schaltung simuliert. Ein großer Vorteil ist die Fähigkeit, einen Zugriff auf bestimmte Teile der Schaltung zu ermöglichen, welche beim physikalischen Hardware-Prototyp nicht verfügbar wären. Dies sind besonders wichtig bei integrierten Schaltungen, wo der Zugriff auf die internen Schaltungsteile generell ausgeschlossen ist.

Bei der analogen Simulation hat SPICE (engl. Simulation Program with Integrated Circuit Emphasis) eine beherrschende Stellung. SPICE wurde Anfang der 70er Jahre an der Berkeley Universität (USA) entwickelt. SPICE ist somit der Inbegriff der analogen Schaltungssimulation und wird bei vielen modernen analogen Simulationswerkzeugen als Hauptsimulator verwendet. Das Besondere ist die textuelle Beschreibung der Schaltung, die Definition der Eingangsparameter und die Festlegung der Analysearten (z.B. AC, DC, Transiente oder Rauschsimulation). Mit der Zeit ist eine Vielzahl von kommerziellen SPICE -Simulatoren mit immer besseren bzw. erweiterten mathematischen Modellen entstanden. Wegen der BSD-Lizenz (Berkeley Software Distribution, Open Source Lizens) wurde der Schaltungssimulator als Basis für viele kommerzielle Simulatoren verwendet. Unternehmen wie Synopsys (HSPICE 2010), Cadence (Spectre 2013), National Instruments (Multisim 2012) und viele mehr haben kommerzielle Versionen von SPICE auf den Markt gebracht. Neben analogen Schaltungssimulatoren existieren auch Simulatoren für digitale bzw. gemischte Schaltungen (engl. mixed analog-digital), Antennen und Hochfrequenztechnik. Eine Übersicht über die aktuelle Verbreitung der Schaltungssimulation im Analogbereich liefert (Schwarz 2015). Einzelne kommerzielle Simulatoren bieten neben der Schaltungssimulation zusätzliche Funktionen, die eine Fehlersimulation im eingeschränkten Maße möglich macht, jedoch vorwiegend auf digitale Signale eingeschränkt. Die sogenannten Module (z.B. Synopsis TetraMAX ATPG oder Cadence Verifault-XL) arbeiten grundsätzlich auf höheren Abstraktionslevel (Baugruppen bzw. Block Ebene) und die Fehlerbeschreibung erfolgt in den meisten Fällen mit Verilog HDL (engl. hardware description language).

Auch wenn die Schaltungssimulation sehr nützlich ist, so hat jedes modellbasierte Simulationssystem auch Einschränkungen. Elektronische Komponenten können sehr komplex sein und ein mathematisches Verhaltensmodell kann nicht alle real existierenden Verhaltungsweisen exakt definieren. Für komplexe Schaltungen mit speziellem Verhalten kann die Simulation einen falschen Eindruck vermitteln oder schlicht unkorrekt sein. Ebenso können Rauschen oder Interferenzen auftreten, die beim simulierten Modell eventuell nicht berücksichtigt wurden. Ungenauigkeiten einzelner Komponentenparameter können zu größeren Ungenauigkeiten des betrachteten Systems führen.

2.3 FEHLERINJEKTION

Es gibt unterschiedliche Arten von Fehlersimulation und unterschiedliche Ebenen, auf denen eine Fehlerinjektion vorgenommen werden kann. Eine Fehlerinjektion wird als ein Validierungsverfahren für die Bewertung der Zuverlässigkeit von Systemen angesehen, sowohl von fehlertoleranten als auch von nicht fehlertoleranten Systemen. Die Überprüfung wird in einem kontrollierten Experiment durchgeführt, um das Verhalten des untersuchten Systems in Gegenwart von Fehlern zu bewerten. Durch den anschließend folgenden Schritt der Fehlersimulation können die Auswirkungen auf die Leistung und das Ausbreiten bzw. Fortpflanzen eines Fehlers oder auch von mehreren injizierten Fehlern untersucht werden. In der folgenden Grafik sind die unterschiedlichen Systemebenen dargestellt, die für die Fehlerinjektion in Frage kommen. Je nach betrachtetem System kann die Fehlerinjektion hardwarebasiert, softwarebasiert, emulationsbasiert oder simulationsbasiert erfolgen. In dieser Arbeit wird die simulationsbasierte Fehlerinjektion angewendet. Das untersuchte System (analoge Schaltung) und mögliche Hardwarefehler werden modelliert und das Verhalten mit einem Schaltungssimulator simuliert. Die Fehlersimulation wird durch die Modifizierung des Hardwaremodells mit vordefinierten Fehlermodellen durchgeführt. Die Fehlermodelle bei analogen Fehlersimulatoren beziehen sich in den meisten Fällen auf die Bauelemente (engl. component) und Baugruppenebene (engl. assembly). Durch die Modifizierung des Systems kann sich das System so verhalten, als würde der Hardwarefehler bestehen. Dadurch kann das Fehlverhalten studiert werden. Ebenso können Aussagen zu gewollter bzw. vorhandener Fehlertoleranz abgeleitet werden.

Wie in Abbildung 4 zu sehen ist, wird ein System in unterschiedliche Abstraktionsebenen unterteilt. Auf jeder der Ebenen wird eine andere Beschreibungssprache oder Modellierungssprache verwendet. Dieses verkompliziert nicht nur den Entwurf neuer Systeme, sondern auch die Überprüfung des Systems auf Fehlertoleranz. Für die dargestellten Ebenen existieren einzelne Implementierungen für Fehlersimulatoren, die je nach Ebene unterschiedliche Ansätze für Fehlerinjektion verfolgen. Grundsätzlich kann gesagt werden, dass je höher die Systemebene ist, desto abstrakter, allgemeiner und in vielen Fällen auch ungenauer sind die Fehlermodelle. Alle Ebenen in Abbildung 4 wurden als Zuverlässigkeitsblockdiagramme (ZBD, eng. reliability block diagram, RBD) modelliert und zeigen die notwendigen und redundanten Elemente bzw. Komponenten für die Erfüllung der Funktion auf. Jeder Block repräsentiert ein

Element aus der jeweiligen Ebene. Weitere Details zu RBD sind im Kapitel 2.4 zu finden.

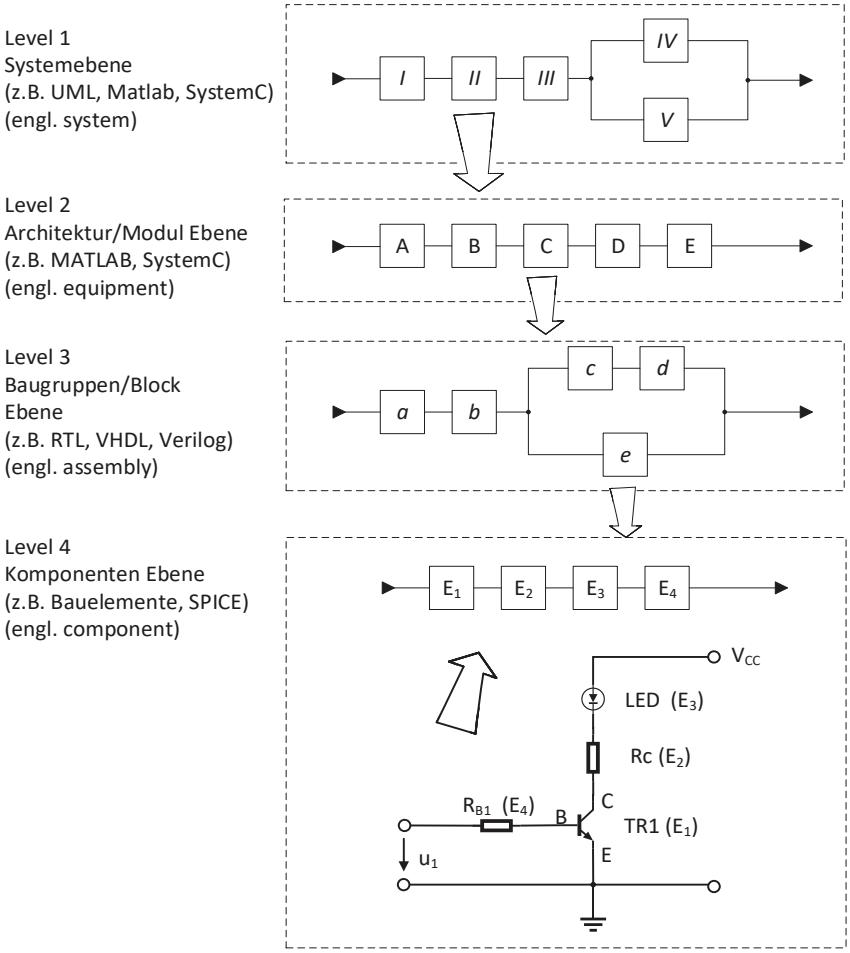


Abbildung 4. Die Ebenen der Fehlerinjektion und Fehlermodellierung. Darstellung mit RBD-Diagrammen. Basierend auf (Birolini 2014, S. 29) und (MIL-HDBK-338B 1998, S. 6–27)

Neben dem unterschiedlichen Vorgehen in der Fehlerinjektion können die einzelnen Methoden in Komplexität, Aufwand und im Fehlerverhalten unterschieden werden. Neben den Fehlermodellen, welche für die Fehlersimulation genutzt werden (siehe Kapitel 3.3) gibt es noch weitere physikalische Fehlerinjektionsarten, die für die vorliegende Arbeit nicht relevant sind. Dazu gehört in erster Linie die Fehlerinjektion durch die Änderung des (externen) Taktsignals, die Änderung der Temperatur auf einen Wert außerhalb des Toleranzbereiches, der Einfluss elektromagnetischer Wellen oder radioaktiver Schwerionen-Strahlung sowie der Einsatz gerichteter Laser. Eine detaillierte Beschreibung der einzelnen Verfahren ist bei (Piscitelli et al. 2015) zu finden.

2.4 BOOLESCHE MODELLE ZUR ZUVERLÄSSIGKEITSBEWERTUNG

Die wichtigste Annahme bei booleschen Zuverlässigkeitsmodellen ist, wie der Name schon sagt, dass Komponenten/Baugruppen/Systeme nur zwei Zustände, den funktionierenden Zustand (engl. operational) und den ausgefallenen Zustand (engl. failed), aufweisen. Die zweite und ebenfalls wichtige Annahme ist die Unabhängigkeit der Ereignisse, die besagt, dass der Ausfall einer Komponente nicht die Ausfallwahrscheinlichkeit einer anderen Komponente beeinflusst. Bezugnehmend auf die Grundlagen der Wahrscheinlichkeits- und Zuverlässigkeitstheorie aus Anhang A wird im Folgenden die quantitative Systemanalyse veranschaulicht.

Der entscheidende Schritt bei der Analyse der Systemzuverlässigkeit ist die Beschreibung der Beziehung zwischen Komponenten und dem Gesamtsystem, auch Systemfunktion bzw. Strukturfunktion (engl. structure function) genannt. Die Systemfunktion drückt die Funktionsfähigkeit des Systems in Abhängigkeit von der Funktionsfähigkeit seiner Komponenten aus. Dies bestimmt die „logische Struktur“ des Systems und damit auch das Zuverlässigkeitsmodell. Die Zuverlässigkeitstheorie untersucht u.a. die Beziehung zwischen dem Betriebszustand eines (komplexen) Systems und den Betriebszuständen der einzelnen Komponenten bzw. Bauelemente. Es wird im einfachsten Fall angenommen, dass sich das System und seine Komponenten nur in zwei Zuständen befinden können: betriebsbereit oder ausgefallen. Die Abhängigkeit zwischen den Zuständen der Komponenten und dem Zustand des Systems wird durch eine binäre Funktion, die sogenannte Struktur-Funktion (engl. structure function) ausgedrückt. Mit anderen Worten: Die Funktionsfähigkeit des Systems stellt sich als boolesche Funktion der Komponentenausfälle dar. Ab den 1960er Jahren entstanden mehrere grafische Darstellungsmodelle dieser booleschen Funktion. So werden u.a. die folgenden Modellierungsmethoden angewendet: Fehlerbaumanalyse (engl. fault tree analysis, FTA), binäres Entscheidungsdiagramm (engl. binary decision diagram, BDD), Ereignisbaum (engl. event tree), Zuverlässigkeits-Blockdiagrame (engl. reliability block diagram, RBD). Die am häufigsten verwendeten Modellierungs- und Analysemethoden sind die Fehlerbaumanalyse und die Zuverlässigkeitsblockdiagramme. Der (wichtigste) Unterschied zwischen den beiden Diagrammen ist die Herangehensweise, um ein Systemausfall zu modellieren. Ein Fehlerbaum ist ein Diagramm, welches das unerwünschte Ereignis eines Systemausfalls auf die dafür ursächlichen Fehlerfälle zurückführt. Das Zuverlässigkeitsblockdiagramm ist ein Diagramm, welches aufzeigt, welche fehlerfreien Komponenten mindestens

erforderlich sind, damit das System insgesamt fehlerfrei ist. Die Darstellungen der booleschen Systemfunktion durch FTA und RBD sind also komplementär zueinander. FTA und RBD setzen voraus, dass die Systemfunktion monoton ist. Wenn es sich nicht um speziell erweiterte Sonderformen von RBD bzw. FTA handelt, kann ein System wahlweise durch ein RBD oder ein FTA dargestellt werden. Schließlich kann eine Systemfunktion wie jede boolesche Funktion auch als Wahrheitstabelle dargestellt werden.

Ein Zuverlässigkeitsblockdiagramm wird genutzt, um die Zuverlässigkeit und Verfügbarkeit eines komplexen Systems zu bestimmen. Die Blöcke eines RBD können in serieller oder paralleler oder einer anderen Konfiguration angeordnet sein. Eine parallele Verknüpfung von zwei oder mehreren Blöcken wird genutzt, um auszudrücken, dass diese Blöcke redundant sind. Für die Funktionsfähigkeit der Parallelanordnung genügt es, wenn mindestens einer dieser Blöcke funktionsfähig ist. Eine serielle Verknüpfung von Blöcken verlangt dagegen, dass alle Blöcke funktionsfähig sind, damit die Serienanordnung insgesamt funktionsfähig ist.

Insgesamt gilt ein beliebiges durch RBD modelliertes System als funktionsfähig, wenn es mindestens einen Pfad vom Start- zum Zielknoten gibt, der ausschließlich über funktionsfähige Blöcke führt.

Sobald ein System als Netzwerk aus Blöcken richtig modelliert ist und die Fehlerraten für die Blöcke bekannt sind, können die Kennwerte (z.B. Überlebenswahrscheinlichkeit, Verfügbarkeit usw.) des Systems bestimmt werden.

In der folgenden Abbildung 5 sind grundsätzliche, also serielle bzw. parallele Blockdiagrammtypen neben äquivalenten Fehlerbäumen dargestellt. Zuverlässigkeitsblockdiagramme entsprechen nur in sehr einfachen Fällen der physikalischen Verknüpfung von System-Komponenten (Bauelementen eine Schaltung). In der folgenden Abbildung 6 sind zwei einfache elektr. Schaltungen in ein Zuverlässigkeitsblockdiagramm überführt worden. Schon auf den ersten Blick ist zu erkennen, dass eine direkte Überführung des physikalischen Schaltungsmodells in ein RBD unter Beibehaltung der Systemstruktur in der Regel nicht möglich ist. Die Funktion der Schaltung und das Verhalten bei Auftreten von Fehlern sind relevant, damit das korrekte RBD entsteht. Veränderungen in der Konfiguration des RBD führen auch zu Veränderungen der Werte der Zuverlässigkeitskenngrößen des untersuchen Systems.

Mit Hilfe der Fehlersimulation kann ein RBD wie folgt erstellt werden: Die Simulation verschiedener Fehlerfälle zeigt jeweils, ob der betreffende Fall toleriert wird oder nicht. Dadurch entsteht die Wertetabelle der Systemfunktion. Diese kann als RBD dargestellt werden.

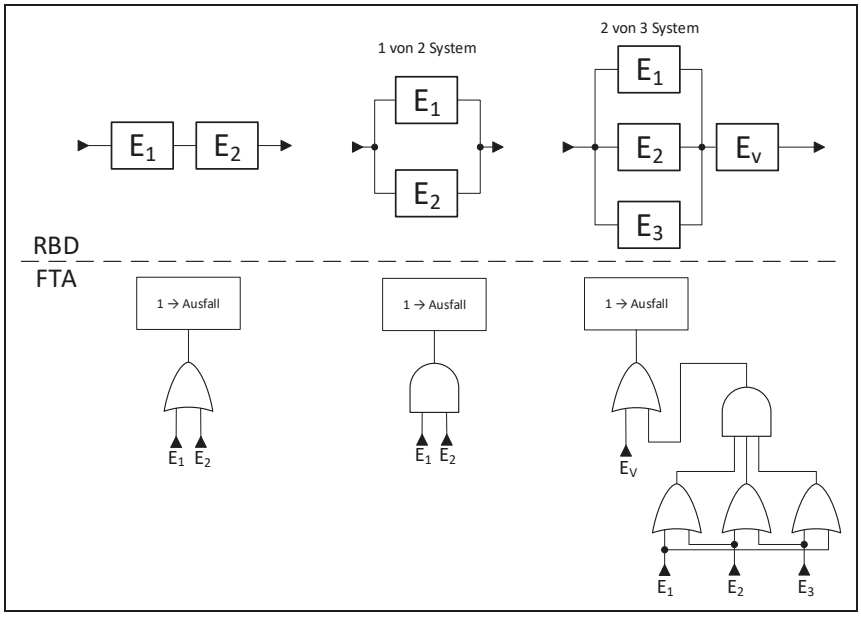
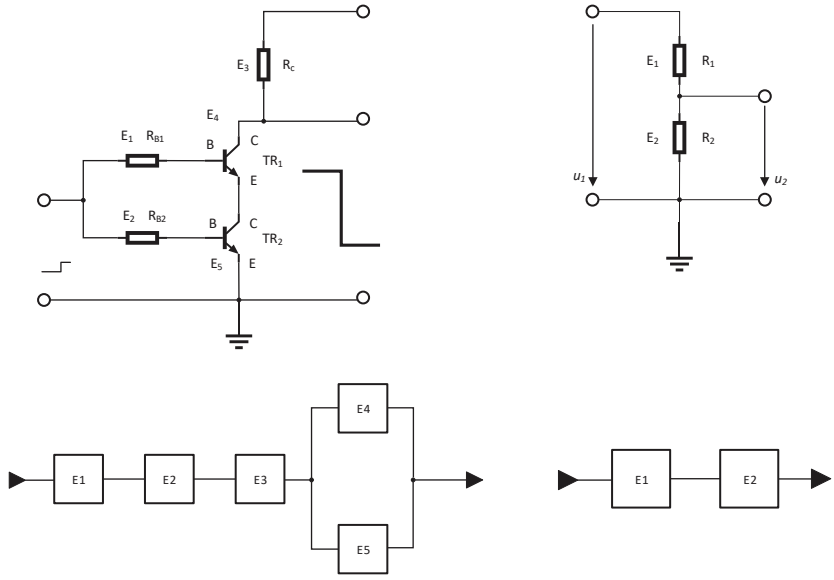


Abbildung 5. Vergleich der Modellierung mittels RBD und FTA



Die Transistoren TR_I und TR_2 sind im RBD-Modell redundant, wenn ihre Fehlerart, der Kurzschluss zwischen Emitter und Kollektor ist und die Widerstände R_{B1} und R_{B2} die Fehlerart Unterbrechung aufweisen

Abbildung 6. Zuverlässigkeitsblockdiagramme für einfache Stromkreise, basierend auf (Birolini 2014, S. 30)

In der folgenden Tabelle 1 und Tabelle 2 sind einfache und komplexe Blockdiagramme und dazugehörige Zuverlässigkeitsfunktionen R(t) abgebildet. Bei den parallelen Strukturen handelt es sich um nichtreparierbare identische Komponenten mit aktiver Redundanz. Manche Systeme lassen sich nicht als Serienparallelsystem ausdrücken (siehe Diagramm (IV) der Tabelle 2) – genau genommen: Wenn solche Systeme in ein Serienparallesystem überführt werden, müssen bestimmte Teile im Modell mehrfach vorkommen (wiederholte Blöcke).

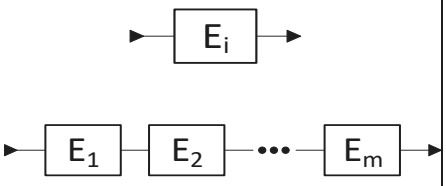
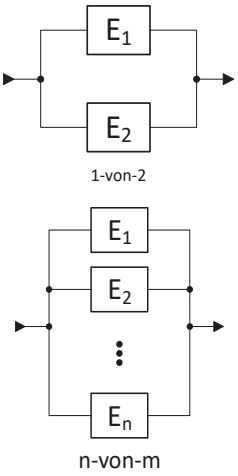
Zuverlässigkeitsblockdiagramm (RBD)	**Zuverlässigkeitsfunktion** $R_i = \mathrm{R}_i(t), \mathrm{R}_i(0) = 1$
Serielle Struktur (I) 	$\mathrm{R_S} = \mathrm{R_i}$ $R_S = \prod_{i=1}^{m} R_i$
Parallele Struktur (II) 	1-von-2 System: $R_S = \mathrm{R}_1 + \mathrm{R}_2 - \mathrm{R}_1\mathrm{R}_2$ n-von-m System: $\mathrm{E}_1 = \ldots = \mathrm{E}_m = \mathrm{E}$ $\rightarrow \mathrm{R}_1 = \ldots = \mathrm{R}_m = \mathrm{R}$ $R_S = \sum_{i=n}^{m} \binom{m}{i} R^i (1-R)^{m-i}$

Tabelle 1. Grundlegende Zuverlässigkeitsblock-Diagramme und Zuverlässigkeitsfunktionen, basierend auf (Birolini 2014, S. 31)

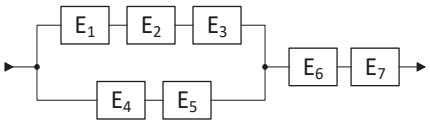
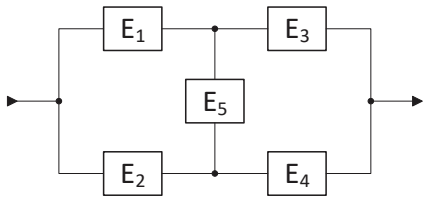
Zuverlässigkeitsblockdiagramm (RBD)	**Zuverlässigkeitsfunktion** $R_i = \mathrm{R}_i(t), \mathrm{R}_i(0) = 1$
Serielle-Parallele Struktur (III) 	$R_S = (\mathrm{R}_1\mathrm{R}_2\mathrm{R}_3 + \mathrm{R}_4\mathrm{R}_5 - \mathrm{R}_1\mathrm{R}_2\mathrm{R}_3\mathrm{R}_4\mathrm{R}_5)\,\mathrm{R}_6\mathrm{R}_7$
Brückenschaltung (IV) 	$R_S = \mathrm{R}_5(\mathrm{R}_1 + \mathrm{R}_2 - \mathrm{R}_1\mathrm{R}_2) * (\mathrm{R}_3 + \mathrm{R}_4 - \mathrm{R}_3\mathrm{R}_4) + (1 - \mathrm{R}_5) * (\mathrm{R}_1\mathrm{R}_3 + \mathrm{R}_2\mathrm{R}_4 - \mathrm{R}_1\mathrm{R}_2\mathrm{R}_3\mathrm{R}_4)$

Tabelle 2. Fortsetzung: Grundlegende Zuverlässigkeitsblock-Diagramme und Zuverlässigkeitsfunktionen, basierend auf (Birolini 2014, S. 31)

Über den Einsatz von Wahrheitstabellen und boolescher Algebra kann eine Berechnung der Zuverlässigkeitskennwerte auch ohne RBD erfolgen. Bei diesem Vorgehen müssen alle Fehlzustände der Komponenten und deren Kombinationen aufgeführt und jeder Fall mit seiner Auftrittswahrscheinlichkeit gewichtet werden. Das folgende Beispiel zeigt die Vorgehensweise. Für weitere Details der Methode siehe auch (MIL-HDBK-338B 1998, 6-33 bis 3-37).

In dem Beispiel werden für Systeme sowie für ihre Komponetne jeweils zwei Zustände definiert: 1 = funktional (engl. success) und 0 = ausgefallen (engl. failure). Die Methode ist aufwändiger als die unmittelbare Verwendung der Zuverlässigkeitsfunktion, weil alle Fehlerereignisse einen eigenen Fall darstellen und nicht zusammengefasst werden. Das Verfahren und der Bewertungsprozess werden wie folgt veranschaulicht. Zur Erstellung der Wahrheitstabelle wurde das Diagramm IV aus der oberen Tabelle 2 ausgewählt. Die Überlebenswahrscheinlichkeiten (R_i) der jeweiligen Komponenten (E_1, E_2, E_3, E_4, E_5) wurden wie folgt festgelegt:

Überlebenswahrscheinlichkeit: ($R_i = \mathrm{R}_i(t)$)				
E1	E2	E3	E4	E5
$R_{E1} = 0{,}9$	$R_{E2} = 0{,}9$	$R_{E3} = 0{,}8$	$R_{E4} = 0{,}8$	$R_{E5} = 0{,}7$

Mit der bekannten Zuverlässigkeitsfunktion wird die Systemzuverlässigkeit berechnet:

$$\begin{gathered} R_S = \mathrm{R_5}(\mathrm{R_1} + \mathrm{R_2} - \mathrm{R_1R_2}) * (\mathrm{R_3} + \mathrm{R_4} - \mathrm{R_3R_4}) \\ +(1 - \mathrm{R_5}) * (\mathrm{R_1R_3} + \mathrm{R_2R_4} - \mathrm{R_1R_2R_3R_4}) = 0{,}94176 \end{gathered} \tag{2.1}$$

Für die Berechnung mit Hilfe der Wahrheitstabelle müssen alle möglichen Kombinationen der Komponentenzustände (E_1, … , E_5) aufgelistet werden. Es ergeben sich insgesamt 2^n Kombinationen (siehe Nr 1- 32 in Tabelle 3), für die eine Prüfung auf Systemausfalls (siehe Spalte „System: Funktional (1)/ Ausfall (0)“) vorgenommen wird. Für jede funktionale Kombination (1) wird die Wahrscheinlichkeit durch Multiplikation der einzelnen Komponenten-Überlebenswahrscheinlichkeiten bzw. ihres Komplements bestimmt. Dies entspricht einem seriellen Modell. Somit ergibt der erste und zehnte Eintrag (Nr. 1 und Nr. 10) der Tabelle 3 die folgenden Wahrscheinlichkeiten:

$$\begin{gathered} R_{Nr\,1} = \mathrm{R_1R_2R_3R_4R_5} \\ = 0{,}9 * 0{,}9 * 0{,}8 * 0{,}8 * 0{,}7 = 0{,}36288 \\ \text{und} \\ R_{Nr\,10} = \overline{\mathrm{R_1}}\mathrm{R_2R_3}\overline{\mathrm{R_4}}\mathrm{R_5} = (1 - \mathrm{R_1})\mathrm{R_2R_3}(1 - \mathrm{R_4})\mathrm{R_5} \\ =(1 - 0{,}9) * 0{,}9 * 0{,}8 * (1 - 0{,}8) * 0{,}7 = 0{,}01008 \end{gathered} \tag{2.2}$$

Wenn disjunkte Fehlerfälle vorliegen (dies entspricht disjunkten Pfaden im RBD), ergibt die Summe aller Wahrscheinlichkeiten der Zustände, in denen das System funktionsfähig ist, (engl. state space method) die Systemüberlebenswahrscheinlichkeit $R_S = 0{,}94176$. Sowohl durch die Zuverlässigkeitsformel der Brückenschaltung als auch durch die Berechnung der Wahrheitstabelle erhalten wir dasselbe Ergebnis.

Das System kann zwei Zustände annehmen:

- einen (voll) funktionierenden Zustand, bei dem keine Komponenten einen Ausfall aufweisen oder aber die Systemfunktion ausgeführt wird, obwohl einige der Komponenten ausgefallen sind (d.h. es liegt Fehlertoleranz vor).
- Einen fehlerhaften Systemzustand, in dem die Systemfunktion nicht mehr erfüllt wird, aufgrund des Ausfalls einer oder mehrerer Systemkomponenten.

Nr	E_1	E_2	E_3	E_4	E_5	System: Funktional (1)/ Ausfall (0)	Zuverlässigkeit	Nr	E_1	E_2	E_3	E_4	E_5	System: Funktional (1)/ Ausfall (0)	Zuverlässigkeit
1	1	1	1	1	1	1	**0,3628**	17	1	1	1	1	0	1	**0,1555**
2	0	1	1	1	1	1	**0,0403**	18	0	1	1	1	0	1	**0,0172**
3	1	0	1	1	1	1	**0,0403**	19	1	0	1	1	0	1	**0,0172**
4	0	0	1	1	1	0	**0**	20	0	0	1	1	0	0	**0**
5	1	1	0	1	1	1	**0,0907**	21	1	1	0	1	0	1	**0,0388**
6	0	1	0	1	1	1	**0,0100**	22	0	1	0	1	0	1	**0,0043**
7	1	0	0	1	1	1	**0,0100**	23	1	0	0	1	0	0	**0**
8	0	0	0	1	1	0	**0**	24	0	0	0	1	0	0	**0**
9	1	1	1	0	1	1	**0,0907**	25	1	1	1	0	0	1	**0,0388**
10	0	1	1	0	1	1	**0,0100**	26	0	1	1	0	0	0	**0**
11	1	0	1	0	1	1	**0,0100**	27	1	0	1	0	0	1	**0,0043**
12	0	0	1	0	1	0	**0**	28	0	0	1	0	0	0	**0**
13	1	1	0	0	1	0	**0**	29	1	1	0	0	0	0	**0**
14	0	1	0	0	1	0	**0**	30	0	1	0	0	0	0	**0**
15	1	0	0	0	1	0	**0**	31	1	0	0	0	0	0	**0**
16	0	0	0	0	1	0	**0**	32	0	0	0	0	0	0	**0**
														Summe aller funktionalen Pfade (1):	**0,9417**

Tabelle 3. Kalkulation der Systemzuverlässigkeit anhand der Wahrheitstabelle

Die einzelnen funktionierenden Pfade in Tabelle 3 können in Form einer Gleichung der booleschen Algebra aufgeschrieben werden. Für das gewählte Beispiel sieht diese folgendermaßen aus:

$$\begin{aligned} R_S = {} & \mathrm{R}_1\mathrm{R}_2\mathrm{R}_3\mathrm{R}_4\mathrm{R}_5 + \overline{\mathrm{R}_1}\mathrm{R}_2\mathrm{R}_3\mathrm{R}_4\mathrm{R}_5 + \mathrm{R}_1\overline{\mathrm{R}_2}\mathrm{R}_3\mathrm{R}_4\mathrm{R}_5 \\ & + \mathrm{R}_1\mathrm{R}_2\overline{\mathrm{R}_3}\mathrm{R}_4\mathrm{R}_5 + \overline{\mathrm{R}_1}\mathrm{R}_2\overline{\mathrm{R}_3}\mathrm{R}_4\mathrm{R}_5 \\ & + \mathrm{R}_1\overline{\mathrm{R}_2\mathrm{R}_3}\mathrm{R}_4\mathrm{R}_5 + \mathrm{R}_1\mathrm{R}_2\mathrm{R}_3\overline{\mathrm{R}_4}\mathrm{R}_5 \\ & + \overline{\mathrm{R}_1}\mathrm{R}_2\mathrm{R}_3\overline{\mathrm{R}_4}\mathrm{R}_5 + \mathrm{R}_1\overline{\mathrm{R}_2}\mathrm{R}_3\overline{\mathrm{R}_4}\mathrm{R}_5 \\ & + \mathrm{R}_1\mathrm{R}_2\mathrm{R}_3\mathrm{R}_4\overline{\mathrm{R}_5} + \overline{\mathrm{R}_1}\mathrm{R}_2\mathrm{R}_3\mathrm{R}_4\overline{\mathrm{R}_5} \\ & + \mathrm{R}_1\overline{\mathrm{R}_2}\mathrm{R}_3\mathrm{R}_4\overline{\mathrm{R}_5} + \mathrm{R}_1\mathrm{R}_2\overline{\mathrm{R}_3}\mathrm{R}_4\overline{\mathrm{R}_5} \\ & + \overline{\mathrm{R}_1}\mathrm{R}_2\overline{\mathrm{R}_3}\mathrm{R}_4\overline{\mathrm{R}_5} + \mathrm{R}_1\mathrm{R}_2\mathrm{R}_3\overline{\mathrm{R}_4\mathrm{R}_5} \\ & + \mathrm{R}_1\overline{\mathrm{R}_2}\mathrm{R}_3\overline{\mathrm{R}_4\mathrm{R}_5} \end{aligned} \tag{2.3}$$

Durch Reduktionstechniken kann die Gleichung noch vereinfacht werden und wird dadurch kompakter. Die allgemeine Formel lautet:

$$R_S = \sum_{(i_1,..,i_n)\in\{0,1\}^n} F_S(i_1,..,i_n) * \prod_{j=1}^{n} \begin{Bmatrix} R_j \; falls \; i_j = 1, \\ 1 - R_j \; sonst \end{Bmatrix} \qquad (2.4)$$

Dabei ist F_S die Systemfunktion, die den Wert 1 liefert, wenn das System für $(i_1,..,i_n)$ funktionsfähig ist. Andernfalls liefert F_S den Wert 0. Im Vektor $(i_1,..,i_n)$ gibt das Element i_j an, ob Komponente j fehlerfrei ist $\left(bei\; i_j = 1\right)$ oder nicht $\left(bei\; i_j = 0\right)$.

2.5 ZUVERLÄSSIGKEITSBEWERTUNG BEI AUFTRETEN VON MEHREREN FEHLERARTEN

In Kapitel 2.4 wurden Modellierungstechniken vorgestellt, die für das System und deren Komponenten nur eine mögliche Fehlerart beim Ausfall von Komponenten vorsehen. Dies erleichtert die Erstellung der Systemfunktion und letztendlich auch die Bewertung der Zuverlässigkeit. Bei elektronischen Systemen und ebenfalls bei Bauelementen von analogen Schaltungen kann man von mehreren unterschiedlichen Fehlerarten ausgehen (siehe Kapitel 3.2). Die Fehlerarten können auch abhängig voneinander sein oder sich gegenseitig beeinflussen. Redundante Komponenten können die Zuverlässigkeit des Systems erhöhen, ohne dass sich die Zuverlässigkeit von einzelnen Komponenten ändert (siehe auch parallele Strukturen (II) in Tabelle 2).

Bei mehr als einer Fehlerart pro Bauelement müssten bei der Zuverlässigkeitsbewertung die tolerierten und die nicht tolerierten Fehlerarten in der Fehlersimulation berücksichtigt werden. Durch fehlerinjizierte Schaltungssimulationen wird jeweils das Verhalten der Schaltung bestimmt, so dass ein Zuverlässigkeitsmodell aufgestellt werden kann. Geplante und nicht geplante Redundanz in der Schaltung führen wie auch das Tolerieren von einzelnen Fehlerarten in Bauelementen zu einer erhöhten Zuverlässigkeit des Systems.

Ein sehr bekanntes Modell für die Bewertung der Systemzuverlässigkeit unter Annahme von zwei Fehlerarten pro Komponente wurde von (Pham 2003) ausführlich untersucht. In diesem Modell kann ein Bauelement mit zwei Anschlüssen (z.B. Widerstand, Diode, Relais, usw.) grundsätzlich auf zwei Arten ausfallen: entweder durch Unterbrechung oder durch einen Kurzschluss. Im Zusammenhang mit der Fehlersimulation von analogen Schaltungen unterbricht die Fehlerart Unterbrechung den Stromfluss völlig, während die Fehlerart Kurzschluss den maximalen Stromfluss bei Wegfall des Widerstands herbeiführt. Bei Mehrfachfehlern bzw. bei zwei Ausfallarten pro Komponente kann eine erhöhte Anzahl von Komponenten die Systemzuverlässigkeit entweder erhöhen oder verringern. Wenn die Fehlerarten (z.B. Kurzschluss (open) und Unterbrechung (short)) nicht gleichzeitig auftreten können (was einen Sonderfall darstellt), wäre die Systemzuverlässigkeit wie folgt zu berechnen: $R_S = 1 - P_{open} - P_{short}$, wobei P_{open} und P_{short} die Systemfehlerwahrscheinlichkeit für die jeweilige Fehlerart ist. Sollte bei einer oder mehreren Komponenten der untersuchten Schaltung eine von beiden Fehlerarten toleriert wer-

den, so ist dies als ein Zuverlässigkeitsgewinn des Systems zu interpretieren, siehe auch Tabelle 3. In der folgenden Tabelle 4 sind eine Auswahl von Zuverlässigkeitsfunktionen für zwei Fehlerarten (short und open) für unterschiedliche komponentenbasierte Systemstrukturen zu finden.

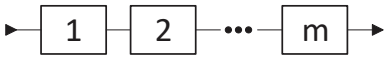
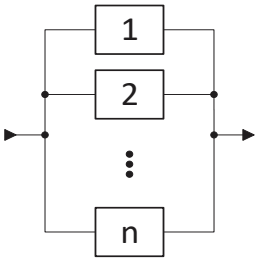
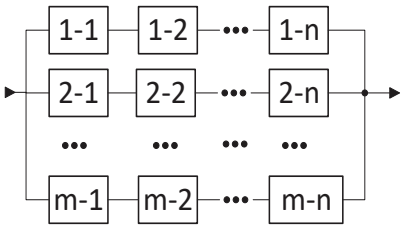
Grundlegende komponentenbasierte Strukturen	**Zuverlässigkeitsfunktion** $R_i = \mathrm{R}_i(t)$
Serielle Systemstruktur (I) 	$R_S = \prod_{i=1}^{m}(1 - R_{i_{open}}) - \prod_{i=1}^{m} R_{i_{short}}$
Parallele Systemstruktur (II) 	$R_S = \prod_{i=1}^{n}(1 - R_{i_{short}}) - \prod_{i=1}^{n} R_{i_{open}}$
Parallel-Serielle Systemstruktur (III) 	$R_{S(n,m)} = \prod_{i=1}^{m}(1 - \prod_{i=1}^{n} R_{i_{short}}) - \prod_{i=1}^{m}\left[1 - \prod_{i=1}^{n}(1 - R_{i_{open}})\right]$

Tabelle 4. Systemdiagramme und zugehörige Zuverlässigkeitsfunktionen bei unabhängigen identischen Komponenten für zwei Fehlerarten (open, short). Basierend auf (Pham 2003, S. 19–25)

Die Komponenten der Parallel-Serien-Systemstruktur (III) in der Tabelle 4 sind im Zustand „fehlerfrei“, „ausgefallen mit Fehlerart Kurzschluss“ (short) oder „ausgefallen mit Fehlerart Unterbrechung“ (open). Komponenten sind nicht reparierbar und die angenommenen Fehler treten unabhängig voneinander auf. Das System kann entweder fehlerfrei sein oder ausfallen, weil in jedem der parallelen Zweige eine der seriell angeordneten Komponenten eine Unterbrechung aufweist, oder ausfallen,

weil in mindestens einem der parallelen Zweige alle Komponenten einen Kurzschluss aufweisen. Für mehr Details zum zuverlässigkeitsoptimierten Systementwurf mit zwei Fehlerarten siehe (Pham 2003).

Im Folgenden wird ein Beispiel für die Zuverlässigkeitsmodellierung einer einfachen Schaltung von drei parallelen Zweigen mit je zwei in Reihe geschalteten Widerständen betrachtet (siehe Abbildung 7), wobei zwei Fehlerarten auftreten können (Kurzschluss und Unterbrechung in einem Widerstand).

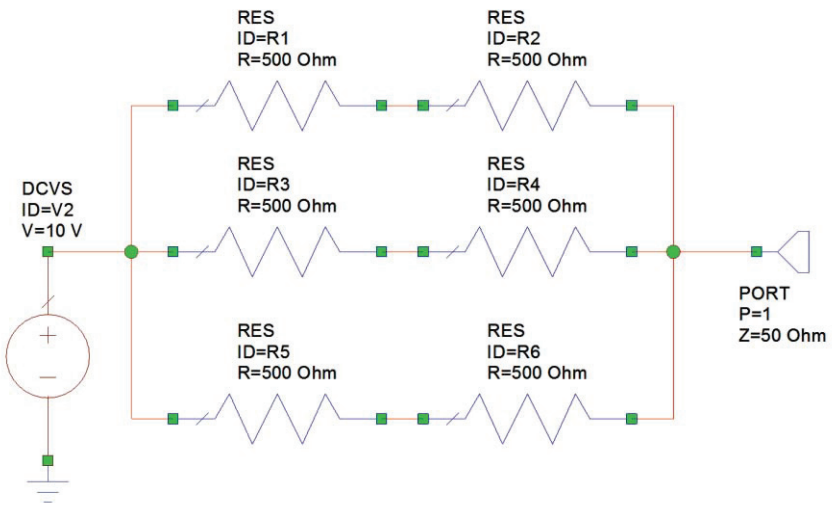


Abbildung 7. Beispiel einer Schaltung für die Zuverlässigkeitsanalyse.

Der resultierende Wiederstand am Port 1 muss in diesem Beispiel gemäß angenommener Schaltungsspezifikation größer als 150 Ohm und kleiner als 1100 Ohm sein. Wenn alle Widerstände funktionieren, beträgt der resultierende Widerstand:

$$\frac{1}{R_{ges}} = \frac{1}{R1+R2} + \frac{1}{R3+R4} + \frac{1}{R5+R6} = \frac{1}{1000} + \frac{1}{1000} + \frac{1}{1000}$$

$$R_{ges} = 333,\overline{3}\ Ohm$$

Wenn einer der Widerstände mit der Fehlerart **Unterbrechung** ausfällt,

$$\frac{1}{R_{ges}} = \frac{1}{\infty} + \frac{1}{1000} + \frac{1}{1000} = \frac{2}{1000} \rightarrow R_{ges} = 500\ Ohm$$

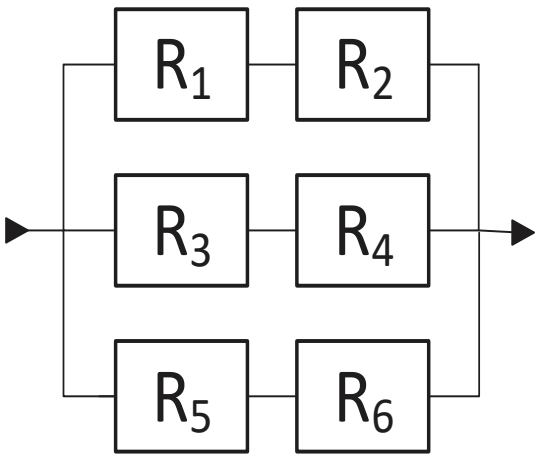
Der resultierende Widerstand beträgt 500 Ohm und liegt damit innerhalb des akzeptierten Bereiches. Daher wird die Schaltung/das System als funktionsfähig betrachtet. Fallen weitere Widerstände (z.B. R3 oder R4, zusätzlich zum ausgefallenen Widerstand R1 oder R2) mit der Fehlerart Unterbrechung aus, wird der Zustand des Systems durch die weitere Erhöhung des Widerstandswertes am Port 1 nicht besser (monotones Systemverhalten). Auch nach dem zweiten Ausfall bleibt der Widerstand im akzeptierten Bereich (zwischen 150 und 1100 Ohm):

$$\frac{1}{R_{ges}} = \frac{1}{\infty} + \frac{1}{\infty} + \frac{1}{1000} = \frac{1}{1000} \rightarrow R_{ges} = 1000\ Ohm$$

Bei Ausfall mindestens einer Komponente in jeder der drei seriellen Zweige (z.B. R1, R3 und R5) erfolgt ein Ausfall des Systems:

$$\frac{1}{R_{ges}} = \frac{1}{\infty} + \frac{1}{\infty} + \frac{1}{\infty} = \frac{1}{\infty} \rightarrow R_{ges} = \infty\ Ohm$$

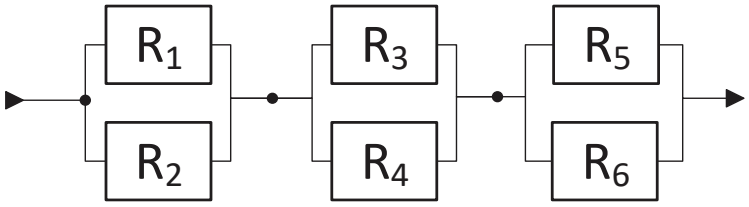
Würde man nur die Fehlerart Unterbrechung betrachten, läge aus der Sicht der Zuverlässigkeitsmodellierung (z.B. RBD) ein parallel-serielles System vor, so wie die Widerstände auch physisch angeordnet sind.



Würde nur die Fehlerart **Kurzschluss** betrachtet werden, so kann der Widerstand an dem betreffenden Bauelement (z.B. R1) nahe bei 0 Ohm angenommen werden. Der gemessene Gesamtwiderstand liegt dann bei:

$$\frac{1}{R_{ges}} = \frac{1}{500} + \frac{1}{1000} + \frac{1}{1000} \rightarrow R_{ges} = 250\ Ohm$$

Das System würde bei dem Ausfall eines Widerstandes mit Kurzschluss nicht als Ausfall, sondern als tolerant gegenüber der Fehlerart Kurzschluss gelten, da der Widerstandswert im definierten Bereich (150 bis 1100 Ohm) liegt. Der Ausfall weiterer Komponenten mit der Fehlerart Kurzschluss würde das System nicht unbedingt ausfallen lassen, sondern wäre für einzelne Kombinationen innerhalb des geforderten Widerstandswertes. Erst beim Ausfall von bestimmten Kombinationen („R1 und R2“, „R3 und R4“ oder „R5 und R6“) wäre der Widerstand nahe 0 Ohm und das System würde als ausgefallen gelten. Dadurch ergibt sich eine andere Struktur bei der Zuverlässigkeitsmodellierung. Ein RBD für die Fehlerart Kurzschluss wäre demnach ein seriell-paralleles System:



Wie die folgende Rechnung zu diesem einfachen Beispiel zeigt, würde sich bei Betrachtung der Fehlerart Unterbrechung die Zuverlässigkeit stärker erhöhen als bei der Fehlerart Kurzschluss. Angenommen die Fehler-Wahrscheinlichkeit eines Widerstands sei für Unterbrechung und für Kurzschluss jeweils 0,01. Dann fällt ein Seriensystem aus zwei Widerständen mit 0,01 + 0,01 – 0,01 · 0,01 ≈ 0,02 aus. Die Ausfallwahrscheinlichkeit des Gesamtsystems beträgt daher bei Unterbrechung 0,02 · 0,02 · 0,02 = 0,000008. Damit verbessert sich die Zuverlässigkeit um den Faktor 0,01 / 0,000008 = 1250. Ein Parallelsystem aus zwei Widerständen fällt aus mit 0,01 · 0,01 = 0,0001. Die Ausfallwahrscheinlichkeit des Gesamtsystems beträgt daher bei Kurzschluss $3 \cdot 0{,}0001 - 3 \cdot 0{,}0001^2 + 0{,}0001^3 \approx 0{,}0003$. Damit verbessert sich die Zuverlässigkeit bei Kurzschluss um den Faktor 0,01 / 0,0003 ≈ 33.

Um die Systemzuverlässigkeit mit mehreren Fehlerarten bestimmen zu können, wird ein Verfahren entsprechend Tabelle 3 auf mehrere Fehlerarten pro Komponente erweitert. Da sich die beiden Fehlerarten Unterbrechung und Kurzschluss innerhalb einer Komponente gegenseitig ausschließen, errechnet sich die Systemzuverlässigkeit für die einzelnen Zustände über die Formel 2.5. Dieser Ansatz geht weiter als die Betrachtung von nur einer möglichen Fehlerart pro Komponente, vielmehr berücksichtigt er beliebig viele Fehlerarten pro Komponente, beispielsweise Kurzschluss, Unterbrechung oder auch Parameterfehler, und lässt darüber hinaus zu, das in verschiedenen Komponenten Fehler unterschiedlicher Art gleichzeitig auftreten.

$$R_S = \sum_{\substack{(i_1,..,i_n) \\ \in\{0,..,m\}^n}} F_S(i_1,..,i_n) * \prod_{j=1}^{n} \begin{Bmatrix} 1 - \sum_{k=1}^{m} P_{jk} \ falls\ i_j = 0, \\ P_{jk}\ falls\ i_j = k \end{Bmatrix} \qquad (2.5)$$

Dabei ist F_S die Systemfunktion in Abhängigkeit von den Komponentenzuständen, die den Wert 1 liefert, wenn das System für die Komponentenzustände $(i_1,..,i_n)$ funktionsfähig ist. $i_j = 0$ entspricht „fehlerfrei“ und $i_j = k$ entspricht „Ausfallart k". n ist die Anzahl der Komponenten und m Anzahl der Fehlerarten. P_{jk} ist die Wahrscheinlichkeit, dass Ausfallart k in Komponente j auftritt.

2.6 KOMPLEXITÄT DER FEHLERSIMULATION BEI MEHRFACHFEHLERN UND MEHREREN FEHLERARTEN

Der Aufwand der Fehlersimulation von analogen Schaltungen ist von mehreren Faktoren abhängig:

- Anzahl der Komponenten (n),
- Anzahl der Fehlerarten (m) pro Komponente
- Die Fehlertiefe (FT, Anzahl der injizierten Mehrfachfehler)

Die Anzahl der Komponenten einer Schaltung ist für die Dauer einer Fehlersimulation maßgebend. Der Aufwand für die notwendigen Durchläufe aller Fehlersimulationen ist wegen der Kombination aus Anzahl der Fehlerarten und zusätzlich der Fehlertiefe ausschlaggebend. Der Grund dafür ist, dass die Anzahl der Simulationen deutlich zunimmt, wenn die Komponenten (n) mehr als eine Fehlerart (m) aufweisen. Die maximale Fehlertiefe ist begrenzt durch die Komponentenanzahl, die mit der Fehlersimulation untersucht wird. In der folgenden Tabelle wird eine Beispielrechnung für den notwendigen Aufwand (Anzahl der Fehlersimulationen) durchgeführt, um eine bestimmte Fehlertiefe zu erreichen:

Fehlertiefe	Anzahl der Simulationen	Ergebnis für n=10 und m=3
0	$\binom{n}{0} * m^0$	1 (Simulation der fehlerfreien Schaltung)
1	$\binom{n}{1} * m^1$	30
2	$\binom{n}{2} * m^2$	405
3	$\binom{n}{3} * m^3$	3240
4	$\binom{n}{4} * m^4$	17010

Tabelle 5. Beispielrechnung für den Aufwand der Fehlersimulation

Der Aufwand der Fehlersimulation lässt sich beschreiben durch:

$$f(FT) = \left\{ \binom{n}{FT} * m^{FT}, \qquad FT \geq 0 \right.$$

Der Aufwand mit der Schrankenfunktion (O-Notation) lässt sich über den binomischen Lehrsatz bestimmen:

$$\sum_{FT=0}^{n} \binom{n}{FT} * m^{FT} = O((m+1)^n)$$

Der Aufwand (CPU-Zeit) ist im schlechtesten Fall (worst-case Laufzeit) in der exponentiellen Komplexitätsklasse $O(m^n)$. Wird eine bestimmte Fehlertiefe wie in Tabelle 5 festgelegt, ist die Komplexitätsklasse mit $O(n^m)$ polynomial.

Insgesamt ist die Anzahl der Fehlersimulationen bei Berücksichtigung von Mehrfachfehlern und mehreren Fehlerarten sehr hoch. In der praktischen Anwendung ist der Aufwand viel niedriger, da eine Vielzahl von Fehlern und Fehlerkombinationen bei höheren Fehlertiefen aufgrund von geeigneten Techniken (beispielsweise Monotonie-Regel) reduziert werden kann, siehe auch Kapitel 3.6.

3 FEHLERSIMULATION

3.1 FEHLERSIMULATION VON ANALOGEN UND GEMISCHTEN SCHALTUNGEN

Fehlersimulation selbst ist ein System aus Hardware und Software welches dazu verwendet wird, das Verhalten des Untersuchungsgegenstandes (z.B. einer analogen Schaltung) zu analysieren. Dazu wird das Verhalten bzw. das Fehlverhalten durch Fehlerinjektion hervorgerufen, wobei die Injektion des Fehlers auf mehrere Arten erfolgen kann. Das Ziel ist die Überprüfung des untersuchten Systems auf die durch Fehler hervorgerufenen Veränderungen des Systemverhaltens, einschließlich Folgefehlern.

Durch die weitere Entwicklung der eingesetzten Technologien (z.B. Nanostrukturen) nimmt die Wahrscheinlichkeit von potentiellen Fehlern in integrierten Schaltungen stetig zu. Daher ist das Interesse an der Erkennung von Fehlerursachen und am Einsatz von Fehlertoleranz-Mechanismen gestiegen (Ziade et al. 2004). In der Halbleiterindustrie ist die Ausbeute (engl. yield) ein Indiz für die Funktionsfähigkeit und die Zuverlässigkeit von integrierten Schaltungen (IC). Während der IC-Herstellung ist die Reduktion der Ausbeute (engl. yield loss) auf Schaltungsdefekte, Variation in den Prozessparametern und Schwächen im Schaltungsentwurf zurückzuführen. Die Zuverlässigkeit von Schaltungen kann direkt mit der Ausbeute in Verbindung gebracht werden.

Fehlerinjektion ist das wichtigste Instrument der Fehlersimulation und eine sehr nützliche Technik, um die Zuverlässigkeit des Systems unter Einwirkung von Fehlern zu evaluieren. Eine Fehlerinjektion kann auf unterschiedliche Arten und Abstraktionsebenen erfolgen. Die hardwarebasierte Fehlerinjektion (engl. hardware-based fault injection), die auf physikalischen Fehlermodellen basiert, wird bei Hardware-Prototypen angewendet. Dabei werden zwei Arten von Fehlerinjektion unterschieden, kontaktlose bzw. berührungsfreie Injektion (z.B. Ionen-Strahlung) und Injektion über Kontakte (z.B. über Anschlusspins). Die softwarebasierte Fehlerinjektion (engl. software-based fault injection) bzw. softwareimplementierte Fehlerinjektion hat zum Ziel, auf Softwareebene (z.B. Softwareanwendung) einen Fehler zur Laufzeit oder zur Entwicklungszeit zu injizieren und das Fehlerverhalten zu analysieren. Dabei werden durch Software die Auswirkungen von angenommenen Hardwarefehlern nachgebildet. Dieses Verfahren kann keine Fehler in Bereiche injizieren, die außerhalb des Softwarezugriffes liegen. Sowohl die hardwarebasierte als

auch die softwarebasierte Fehlerinjektion benötigt einen Prototyp des untersuchten Systems (Mei-Chen Hsueh et al. 1997). Ein Prototyp ist jedoch in den frühen Phasen der Systementwicklung nicht verfügbar. Die für diese Arbeit relevante Methode ist daher die simulationsbasierte Fehlerinjektion (engl. simulation-based fault injection). Diese simuliert ein Modell des untersuchten Systems und die möglichen Fehler auf einem Software-System, welches üblicherweise als Fehlersimulator (engl. fault simulator) bezeichnet wird. Die Fehler werden durch Modifikation des Modells von Hardware-Elementen in das simulierte System injiziert. Fehlerinjektionen können softwareseitig entweder zur Kompilierzeit oder während der Laufzeit erfolgen (Kooli und Di Natale 2014; Ziade et al. 2004)

Vorteile:
Einer der wichtigsten Vorteile der Simulation eines Systemmodells besteht darin, dass es im Stande ist, den Benutzern ein Feedback zu liefern, bevor das reale System gebaut wird. Es gibt einem Systementwickler auch die Möglichkeit, die Leistungsfähigkeit oder auch die Zuverlässigkeit des Systemdesigns zu bestimmen. Folglich können ebenfalls alternative Systemdesigns untersucht bzw. bewertet werden, ohne das System physisch aufbauen zu müssen. So können auch Veränderungen des Systems frühzeitig in die Konstruktion aufgenommen werden, wodurch die Gesamtkosten für das Konstruieren eines Systems reduziert werden. So ist die Entwicklung von Schaltungen (analoge bzw. gemischte Signale) mit Hilfe der Schaltungssimulation heutzutage sehr weit verbreitet. Sowohl kleinere als auch komplexe Schaltungen können entwickelt und mit Hilfe der Schaltungssimulation getestet und in ihrer funktionalen Leistungsfähigkeit verifiziert werden. Würden die Ideen (wie das Platzieren der Bauelemente und die Verdrahtung der Komponenten) eines Schaltungsentwicklers zuerst als physische Designs hergestellt werden, um anschließend die Leistungsfähigkeit und Zuverlässigkeit zu bewerten, wäre dies ein zeitintensiver und kostspieliger Prozess. Durch die simulationsbasierte Nachahmung eines Schaltungsdesigns kann der Entwickler mit Informationen versorgt werden, welche die richtige Funktion und Effektivität von alternativen Designs bewertet. Nach sorgfältigem Abwiegen der Alternativen kann dann das beste Design (z.B. im Sinne von Effizienz oder Zuverlässigkeit) hergestellt werden. Ein weiterer Vorteil des simulationsbasierten Ansatzes ist die Möglichkeit, die Untersuchungen auf unterschiedlichen Abstraktionsebenen durchzuführen. Durch die Fehlerinjektion auf höherer Ebene kann das Gesamtverhalten eines Systems auf hö-

herem Abstraktionsniveau untersucht werden. Die Kosten für die Untersuchungen (im Sinne des Zeitaufwandes) sind niedriger als bei hardwarebasierter Fehlerinjektionen. Der simulationsbasierte Ansatz hat auch einen starken Kontroll- und Prüfcharakter. D.h. das simulierte System kann sehr genau unter Anwesenheit von Fehlern untersucht und das Verhalten ist an verschiedenen Messpunkten der Schaltung beobachtet werden.

Nachteile:
Nichtsdestotrotz haben alle Methoden der Fehlerinjektion neben den vorher genannten Vorteilen auch ihre Nachteile. Der gravierende Nachteil, welcher bei simulationsbasierten Verfahren hervorzuheben ist, besteht im immensen Rechenaufwand. Als Folge davon sind Ergebnisse erst nach einer umfangreichen, zeitraubenden Simulation verfügbar. So kann ein komplexes System nur mit viel Aufwand und enormer Rechenkapazität simuliert werden. Andererseits erhöhte sich die Rechenleistung in den letzten Jahrzehnten stets. Zur Reduktion der Simulationsdauer können unterschiedliche Verfahren zum Einsatz kommen (siehe dazu Kapitel 4.1). Wenn solche Reduktions-Techniken einbezogen werden, kann es allerdings zu Abweichungen in den Simulationsergebnissen kommen. Allgemein betrachtet ist das simulationsbasierte Verfahren abhängig von der Genauigkeit der verwendeten Fehler- und Systemmodelle. Daher kann selbst der beste Simulator das Vertrauen in ein System nur erhöhen, aber nicht garantieren – es sei denn, dass alle möglichen Zustände und deren Folgen aussagekräftig durch die Simulation überprüft wurden. In der Realität scheidet eine vollständige Simulation aber meist aus Aufwandsgründen aus. Für die Fehlersimulation analoger Schaltungen kommt hinzu, dass noch kein von der Industrie allgemein akzeptiertes Fehlermodell vorhanden ist.

3.2 VON PHYSIKALISCHEN FEHLERN ZU ADÄQUATEN FEHLERMODELLEN

Um genaue Fehlermodelle bilden zu können, müssen die Ursachen und die Entstehung von Fehlern verstanden werden. Besonders bei analogen Schaltungen haben das Schaltungslayout und die Komponentenparameter starke Auswirkungen auf mögliche Fehlfunktionen bzw. Abweichungen von der spezifizierten Leistung. Defekte bzw. unerwünschte Merkmale können sich nicht nur während der Fabrikation (z.B. Herstellungsfehler) einschleichen, sondern auch während der Nutzungsphase entstehen. In der Tabelle 6 und Tabelle 7 sind die physikalischen Defekte und ihre Fehlermodelle dargestellt.

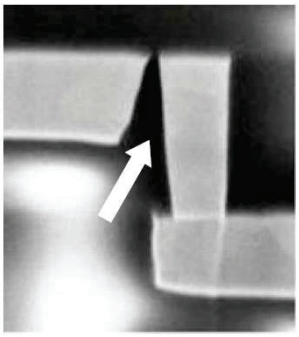
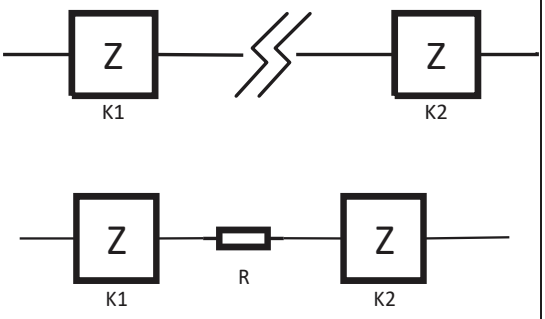
Physikalischer Defekt	**Elektronisches Fehlermodell**
Unterbrechung (engl. open defect)  (Wunderlich 2010, S. 2)	 Eine Unterbrechung kann über Attribute definiert werden: • Standort: zwischen zwei Bauelementen oder innerhalb eines Bauelements (z.B. zwischen Transistor-Anschlüssen) • Komplette (starke) Unterbrechung, d.h. ohne elektrische Verbindung • Teilweise (schwache) Unterbrechung, d.h. Veränderung des Leitungswiderstandes

Tabelle 6. Physikalische Defekte und elektronische Fehlermodelle

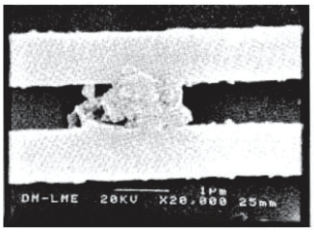
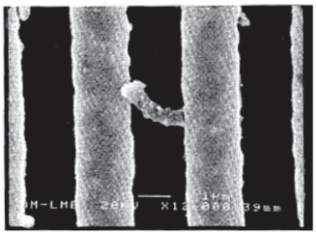
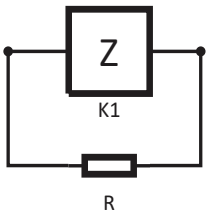
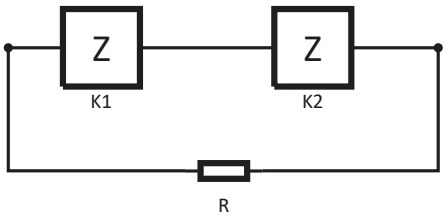
Physikalischer Defekt	Elektronisches Fehlermodell
Kurzschluss (engl. bridging defect / short)  Kurzschluss mit niedrigem Widerstandswert (Wunderlich 2010, S. 41)  Kurzschluss mit hohem Widerstandswert (Wunderlich 2010, S. 41)	  Wie in der oberen Grafik zu sehen kann ein Kurzschluss zwischen den Anschluss-Pins von Bauelementen oder zwischen einzelnen Verbindungsleitungen angenommen werden.

Tabelle 7. Fortsetzung der physikalischen Defekte und elektronischen Fehlermodelle

3.3 REALISTISCHE FEHLERMODELLE AUF KOMPONENTENEBENE

Zur Definition der einzelnen Fehlerarten bzw. zur Festlegung von komponentenbasierten Fehlermodellen wird in dieser Arbeit eine skriptbasierte Beschreibungssprache genutzt. Durch eine API-Schnittstelle (National Instruments 2015) der verwendeten EDA-Software ist ein objektbasierter Zugriff auf die Hauptbestandteile der Simulationssoftware und der Schaltungsbestandteile gegeben. Durch entwickelte VB-Skript-Routinen wird somit die gesamte Fehlersimulation gesteuert. Dies beinhaltet unter anderem die Fehlermodellierung, die Fehlerinjektion, die Schaltungssimulation, die Verarbeitung von Messdaten, die Erzeugung zusätzlicher Schaltungen und Bauelemente und die Veränderung im Sinne der Fehlersimulation. Tabelle 8 und Tabelle 9 liefern eine Übersicht über die möglichen Fehlerarten bzw. Fehlermodelle. Neben den Fehlermodellen wie Kurzschlüsse von Leitungen (ohne einen Widerstandswert und ohne eine Kapazität) sind auch eine Vielzahl anderer Fehlermodellierungen möglich.

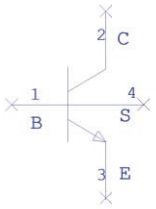
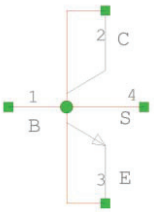
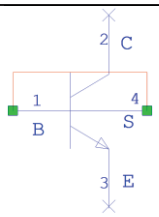
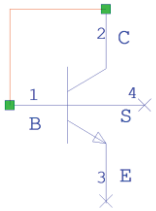
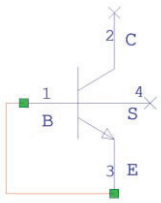
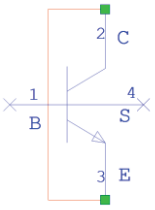
Transistor	
	 Kurzschluss (zw. allen Anschlüssen)  BS Kurzschluss (S= Substrat, B = Basis)    BC Kurzschluss BE Kurzschluss CE Kurzschluss …. und alle Kombination aus 3 Ports sind ebenfalls möglich. Bei Parameterfehlern sind alle primären und sekundären Parameterwerte des Modells veränderbar.

Tabelle 8. Fehlermodelle für Transistoren. Für eine Übersicht an möglichen Parameterwerten eines „Gummel-Poon“ BJT SPICE-Modells siehe (Microwave Office Element Catalog 2016)

Komplette bzw. starke Unterbrechungen von Komponenten kann durch Deaktivieren der Bauelemente modelliert werden. Parameterfehler, schwache Unterbrechungen oder schwache Kurzschlüsse können über die Anpassung der Parameterwerte definiert werden. Grundsätzlich kann der implementierte Fehlersimulator auf alle in der EDA-Umgebung (NI Microwave/Analog Office) hinterlegte Bauelemente bzw. Schaltungsmodelle angewendet werden. Alle Fehlerarten werden per Skriptsprache (VB-Skript) beschrieben und vor jeder Fehlersimulation direkt auf die Schaltung angewendet.

Komponente	Komponente mit Fehlerinjektion
Widerstand 	 Kurzschluss Unterbrechung RES ID=RES1 R=Rx * (1+K) kOhm  Parameterfehler (modellabhängig)
Diode 	 Kurzschluss Unterbrechung **DIODE1** Nu=1.2 T=x * (1+ K) DegC Io=x * (1+ K) mA  Parameterfehler (modellabhängig)
Kondensator 	 Kurzschluss Unterbrechung CAP ID=CAP1 C=Cx * (1 + K) pF  Parameterfehler (modellabhängig)

Tabelle 9. Fortsetzung der Fehlermodelle für den Widerstand, die Diode und den Kondensator

Eines der Grundprobleme der analogen Fehlersimulation besteht darin, dass es kein effizientes und vollständiges Fehlermodell wie bei digitalen Schaltungen (das Stuck-at Modell) gibt, denn die Fehlermodelle (z.B. „nur Unterbrechung“) umfassen nicht alle Fehler von Analogschaltungen. Analoge Fehlermodelle können in zwei Kategorien unterteilt werden. Harte Fehler bzw. katastrophale Fehler (engl. catastrophic/hard faults) und weiche Fehler bzw. Parameterfehler (engl. parametric/soft faults). Harte Fehler (Z.B. Kurzschluss und Unterbrechung) sind diejenigen, die eine Schaltung in den meisten Fällen außerhalb des definierten Arbeitsbereiches bringen. Es gibt jedoch auch harte Fehlerarten, die nicht zu einem Ausfall der Schaltung führen, da diese beispielsweise durch die Fehlertoleranzmechanismen der Schaltung toleriert werden. Durch einen Kurzschluss oder eine Unterbrechung einzelner Komponenten (-anschlüsse) ändert sich die Struktur der verdrahteten Komponenten in den meisten Fällen radikal. Weiche bzw. Parameterfehler entstehen aus der Variation von Parameterwerten bei einzelnen oder mehreren Komponenten. So kann beispielsweise ein Widerstandswert schon während der Fabrikation von der Spezifikation abweichen oder sich während der Betriebsphase verändern (z. B. durch Temperaturänderungen). Ein Parameterfehler und ein katastrophaler Fehler (harte Fehler) müssen nicht unbedingt zu einem Ausfall führen. So kann die Schaltung trotz einer Abweichung einzelner Parameter im spezifizierten Arbeitsbereich bleiben.

Explizite und implizite Redundanz

Die Struktur und der Aufbau der Komponente eines Systems gewährleisten nicht nur die Systemfunktion, sondern können auch eine (un)bestimmte Redundanz erzeugen. Die Redundanz kann geplant sein (explizite Redundanz), um Fehlertoleranz für bestimmte Fehlerarten zu gewährleisten. Eine explizite Redundanz sorgt in erster Linie dafür, dass ein oder mehrere Fehler nicht zu einem Ausfall führen. Ebenso kann ein System (z.B. eine analoge Schaltung) mit unerwarteter bzw. ungeplanter (impliziter) Redundanz aufgebaut sein, was den gleichen positiven Effekt wie eine explizite Redundanz hat und einer Zuverlässigkeitssteigerung gleichkommt, ohne dass die Redundanz dafür vorgesehen war. Gründe für das Vorhandensein impliziter Redundanz können darin liegen, dass ein Schaltungsteil von einer Vorgänger-Schaltung übernommen wurde oder dass manchmal der mehrfache Einsatz billiger Bauteile kostengünstiger als die Verwendung von weniger teuren Elemente ist.

Beispiel für explizite Redundanz in Analogschaltungen.
Als Beispiel für eine explizite Redundanz in einer Analogschaltung wird die einfache Schaltung in Abbildung 8 herangezogen. Die Schaltung soll gewährleisten, dass die LED aufleuchtet, wenn die Steuerungsspannung u_1 einen bestimmten Wert überschreitet (z.B. 4 Volt). Der Transistor (TR_1) fungiert als ein elektrischer Schalter. Da alle Bauelemente für die Funktionalität benötigt werden, ergibt die Schaltung ein Zuverlässigkeitsblockdiagramm, das aus einem seriellen System besteht, siehe Abbildung 8.

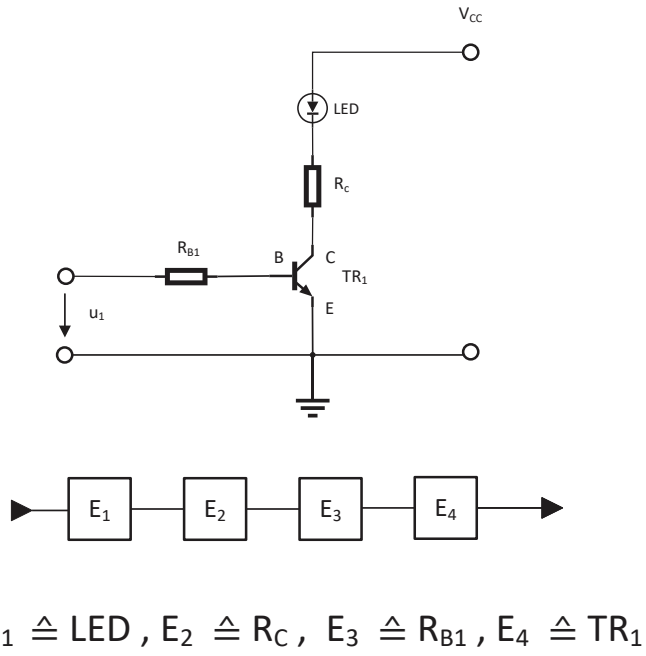
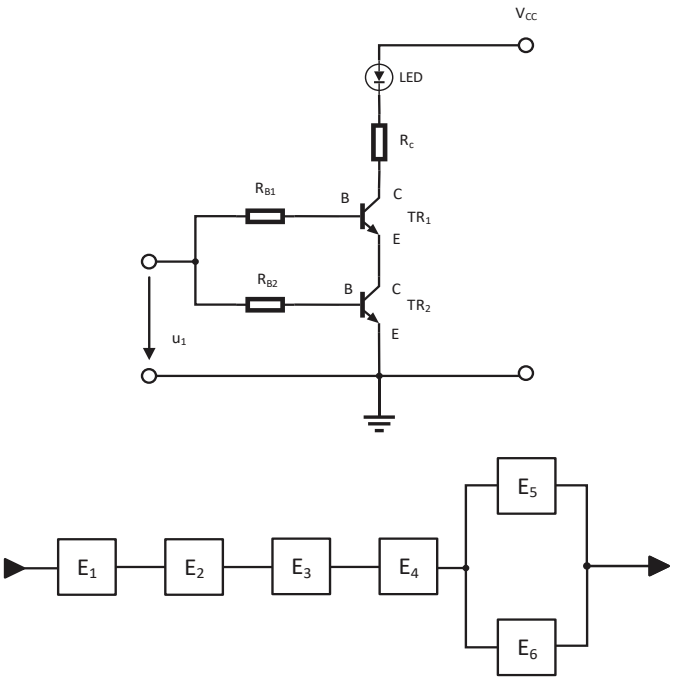


Abbildung 8. Beispiel für ein Zuverlässigkeitsblockdiagramm einer Schaltung. Basierend auf (Birolini 2010, S. 50).

Dieser Beispielschaltung aus Abbildung 8 wird unterstellt, dass der Transistor eine zu hohe Fehlerrate besitzt und kein besserer Transistor als Ersatz verfügbar ist. Um die Zuverlässigkeit der Schaltung zu erhöhen wird explizite Redundanz genutzt. Wenn die Fehlerart Kurzschluss zwischen Emitter-Kollektor für die Transistoren und Unterbrechung für die Widerstände angenommen wird, ergibt sich eine verbesserte Schaltung und das dazugehörige Zuverlässigkeitsdiagramm in der folgenden Abbildung 9. Würde die betrachtete Fehlerart Unterbrechung zwischen Emitter-Kollektor sein, dann wären die Blöcke E_5 und E_6 im Zuverlässigkeitsblockdiagramm in Serie anzuordnen und die explizit geplante Redundanz wäre in diesem Fall mit einer Reduktion der Gesamtzuverlässigkeit von Nachteil, da mehr Komponenten in Serie sind.



$E_1 \triangleq$ LED, $E_2 \triangleq R_C$, $E_3 \triangleq R_{B1}$, $E_4 \triangleq R_{B2}$,
$E_5 \triangleq TR_1$ $E_6 \triangleq TR_2$

Abbildung 9. Beispiel für explizite Redundanz einer Schaltung. Basierend auf (Birolini 2010, S. 51).

Wird die Systemfunktion (nur R_{B1}, R_{B2}, TR_1 und TR_2) des RBD-Diagrammes der Abbildung 9 betrachtet, kann eine Systemfunktion (siehe linkes KV-Diagramm der Abbildung 10) aufgestellt werden. Bei Betrachtung von Doppelfehlern ergibt sich jedoch eine nicht-monotone Systemfunktion (rechtes KV-Diagramm der Abbildung 10). Der Grund für die Nicht-Monotonie ist die Kombination von bestimmten Fehlerarten. Der Kurzschluss eines Transistors (z.B. TR_1) kompensiert die zeitgleiche Unterbrechung des zugehörigen Widerstandes (z.B. R_{B1}) an seiner Basis.

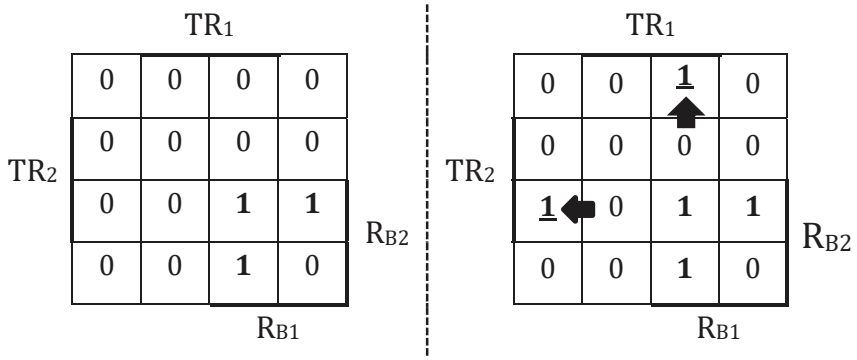


Abbildung 10. Systemfunktion als KV-Diagramm

3.4 BISHERIGE ARBEITEN ZU ANALOGEN FEHLERSIMULATOREN

Heutzutage bestimmen die Prüfkosten der analogen Anteile oft die Gesamtkosten von vielen modernen integrierten Schaltungen (IC's). Wie in Kapitel 1.5 bereits erläutert, haben sich die digitalen DFT-Verfahren sehr zügig durchgesetzt, woraus die Kosten für die Prüfung durch den automatisierten Ansatz stark reduziert werden konnten. Im Vergleich zum aktuellen Stand der Technik digitaler DFT-Verfahren, sind die analogen DFT-Verfahren weit weniger entwickelt und verbreitet. Die fehlende Verfügbarkeit eines allgemeinen analogen Fehlersimulators und/oder automatisierten DFT-Verfahren könnte ein wesentlicher Grund dafür sein, dass die Prüfkosten bei analogen IC's vergleichsweise hoch sind.

Die Kosten für die Prüfung eines neuen Produktes können mehr als 50% der gesamten Produktionskosten betragen. System-on-Chip (SOC) nutzt bis zu 30% der Chipfläche für analoge Komponenten (z.B. analoge Signalverstärker). Bei modernen SOC-Designs wird nur 10% - 15% der Gesamttestzeit für das Prüfen der digitalen Komponenten aufgewendet (Poeh et al. 2010). Die übrige Zeit wird für das Prüfen gemischter (engl. mixed-signal) Signale (z.B. Analog/Digital Konverter), Sende- / Empfangsgeräten oder PMU (engl. power management unit) aufgewendet, siehe Abbildung 11.

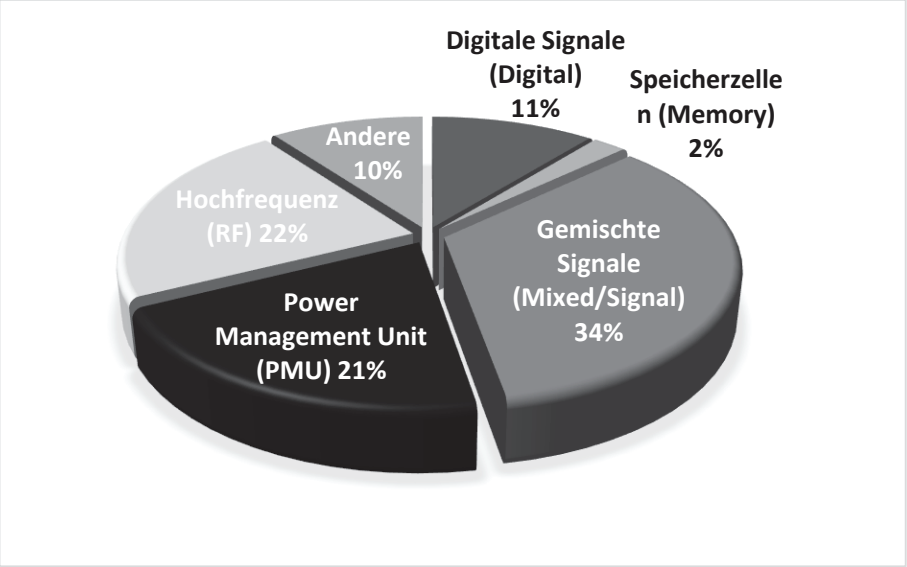


Abbildung 11. Prüfaufwand (in %) in Abhängigkeit von der Signalart. Basierend auf (Poeh et al. 2010).

Die meisten akademischen Arbeiten und Anwendungen analoger Fehlersimulationen wurden ab den 1990er Jahren veröffentlicht. Jedoch wurde keiner der analogen Fehlersimulatoren zu einem weit verbreiteten kommerziellen Produkt entwickelt. Ein Grund hierfür könnte die lange Simulationszeit sein, um die hohe Anzahl möglicher Fehler in modernen bzw. komplexen Schaltungen zu simulieren. Ein weiterer Grund könnte der nicht übereinstimmende Konsens über ein einheitliches analoges Fehlermodell sein. Abbildung 12 zeigt eine Übersicht der publizierten wissenschaftlichen Veröffentlichungen zu analogen Fehlersimulatoren. In den letzten 20 Jahren wurden einige analoge Fehlersimulatoren vorgestellt, welche ihre individuelle Leistungsfähigkeit und Funktionalität aufzeigen. Die ersten Fehlersimulatoren modellieren und simulieren ausschließlich permanente harte Fehler (z.B. Unterbrechung, engl. hard fault). Später wurden auch weiche Fehlermodelle (z.B. Parameterfehler, engl. soft fault) herangezogen. Andere Arbeiten konzentrieren sich auf die Effizienzsteigerung der Simulation, die Automatisierung der Testgenerierung und die Reduktion der Gesamtanalysezeit.

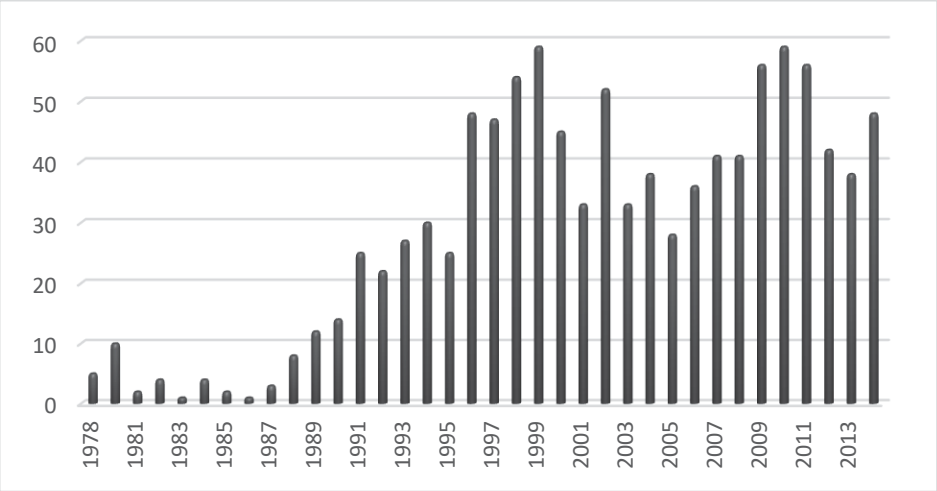


Abbildung 12. Wissenschaftliche Publikationen zu analogen Fehlersimulatoren (IEEE Xplore 2015).

3.5 ÜBERSICHT ÜBER ANALOGE FEHLERSIMULATOREN

Im Folgenden wird ein Überblick über bisherige Arbeiten zur Fehlersimulation unter Verwendung von Schaltungssimulationen (z.B. SPICE) gegeben.

AnaFAULT (Sebeke et al. 1995) wurde an der Universität Hannover entwickelt. Dieser Fehlersimulator nutzt die Anwendung LIFT für das automatische Extrahieren von realistischen Fehlern aus einem Schaltungslayout. Es unterstützt lokale und globale Kurzschluss- bzw. Unterbrechungsmodelle. Für die Fehlermodellierung werden unterschiedliche Widerstandsgrößen (z. B. 0-1Ω für den Kurzschluss und >100 MΩ für eine Unterbrechung) genutzt. Die Fehler werden automatisiert injiziert und mit SPICE-Netzlisten simuliert. Hierfür wird ANACAD Eldo als Simulator für die Verifikation genutzt, wobei nur Einfachfehler untersucht werden. GenFAULT (Hoffmann 2002) ist eine Erweiterung des Fehlersimulators, welche die automatische Generierung von Fehlerlisten aus dem Layout vornimmt, Informationen erzeugt und die vorhandene Methode erweitert. Die Entwicklung von AnaFAULT reicht zurück in das Jahr 1988. Zu dieser Zeit wurde auch der Fehlersimulator FAUSTUS (Fault Simulation Tool Using SPICE) (Lipinski 1990) entwickelt. FAUSTUS bestand aus einer Anzahl von Fortan- und PASCAL-Skripten (Machado 1996).

ANTICS (Spinks 1997) hat einen vergleichbaren Simulationsansatz wie AnaFAULT mit zum Teil anderen Funktionalitäten. ANTICT nutzt HSPICE als Schaltungssimulator und steuert die Fehlersimulation innerhalb der SPICE-Netzlisten durch SPICE-ähnliche Befehle. ANTICS ist in C implementiert und die Software besteht aus vier Programmteilen. ANAFINS (Fault Injection) ist für die Fehlerinjektion zuständig, MC_RAND (Ramdom Number Generation) ist ein Monte-Carlo-Generator für die zufällige Fehlereingabe-Generierung, ANAFAME (Network Simulation Management) sorgt für verteilte Fehlersimulation in einem Rechner-Netzwerkverbund und ANACOV (Post Processing Analysis) bietet Analysemöglichkeiten für Simulationsergebnisse (Bell 2007).

CONCERT (Junwei Hou 1998) ist ein nebenläufiger Fehlersimulator, der sich auf die effiziente Simulation von transienten Fehlern in analogen Schaltungen konzentriert. Dieser Fehlersimulator für nichtlineare analoge Schaltungen beschleunigt die Simulationszeit durch mehrere heuristische Techniken (z.B. dynamische Fehlersortierung für die Verwendung zwischen den einzelnen Simulationsläufen). Dieser spezielle Ansatz ver-

spricht die transiente Fehlersimulation ohne Verlust an Genauigkeit signifikant zu beschleunigen. CONCERT2 (Junwei Hou und Chatterjee 2003) ist eine erweiterte Implementierung des Fehlersimulators in C/C++, welches das Interface von SPICE3f5 adaptiert und damit auch SPICE-Netzlisten akzeptiert. Jeder Fehler der erzeugten Liste wird während der nebenläufigen Fehlersimulation in eine Kopie der fehlerfreien Schaltung injiziert. DRAFTS (Nagi et al. 1993) und FLYER (Variyam und Chatterjee 1997) sind ebenfalls transiente Fehlersimulatoren für lineare analoge Schaltungen, welche als Vorentwicklungen von CONCERT betrachtet werden können.

aFSIM (Straube et al. 2000) wurde am Fraunhofer Institut für integrierte Schaltungen in Dresden entwickelt. Der Prototyp eines analogen Fehlersimulators wurde in Java entwickelt und unterstützte den Schaltungssimulator TITAN (Infineon) und SABER (Analogy). Später wurde aFSIM zu einem Simulator erweitert, der auf unterschiedlichen Abstraktionsebenen Fehlersimulationen durchführen kann (Straube et al. 2002). Daher können Schaltungen sowohl in Form von Netzlisten mit analogen Bauelementen (wie z.B. Widerstände, Kondensatoren, Transistoren usw.) oder in Form einer analogen Beschreibungssprache (Verilog-AMS oder VHDL-AMS) vorliegen. Der zuletzt genannte Ansatz reduziert die Simulationszeit bzw. die Diagnosezeit im Vergleich zur Simulation auf Transistorebene, wobei die geringere Simulationszeit gegen die geringere Genauigkeit der Simulationsergebnisse eingetauscht wird.

Tessent DefectSim (Mentor Graphics 2016) ist zum Zeitpunkt der vorliegenden Arbeit ein weiterer Versuch, ein kommerzielles Produkt zu etablieren, welches die bisher besten akademischen Ansätze von analogen Fehlersimulatoren zu kombinieren versucht. Neben der Anwendung der bekannten analogen Fehlermodelle wird auch ein Fokus auf die Reduktion der Simulationszeit gelegt. DefectSim nutzt den Mentor Graphics Eldo-Simulator, wenn die Schaltung nur SPICE- und Verilog-A-Modelle enthält und Mentor Graphics Questa ADMS bei Verilog-AMS oder RTL-Modellen.

3.6 LÖSUNGSANSÄTZE ZUR REDUKTION DER SIMULATIONSZEIT

Eines der Grundprobleme in der analogen Fehlersimulation ist die relativ lange Simulationszeit. Die Beschleunigung der Fehlersimulation und damit in erster Linie die Reduktion der Gesamtsimulationszeit ist ein wichtiges Ziel in der analogen Fehlersimulation. Zur Reduktion des Simulationsaufwandes mit Fehlerinjektion kommen grundsätzlich zwei Herangehensweisen in Betracht: die Reduktion der Anzahl der Simulationsläufe (mit Fehlerinjektion) und die Beschleunigung der einzelnen Simulationsläufe. Viele Methoden zur Verringerung der Simulationszeit wurden im Laufe der Jahre in wissenschaftlichen Veröffentlichungen vorgeschlagen. Bezüglich der folgenden Techniken wird nun aufgezeigt, wie die Simulationszeit zum Teil sehr stark reduziert werden kann.

3.6.1 ZUFÄLLIG AUSGESUCHTE STICHPROBE

Eine einfache Zufallsstichprobe (engl. simple random sampling) ist eine weitgehend bekannte Auswahlmethode für digitale Fehlerklassen. Es handelt sich um eine klassische Vorgehensweise, in der ein kleiner Stichprobenumfang ausgewählt wird, um bestimmte charakteristische Eigenschaften (z.B. Fehlererkennungsgrad, engl. fault coverage) der gesamten Fehlermenge zu bestimmen. Die Genauigkeit des bestimmten Wertes hängt von der Größe der ausgewählten Stichprobenmenge ab. Wenn beispielsweise die Anzahl der potenziellen bzw. möglichen Fehler bei 40.000 liegt und 4.000 Fehler ausgewählt werden, würde dies einer 10-fachen Reduktion der Simulationszeit gleichkommen. Durch eine so starke Reduktion der Fehlermenge werden selbstverständlich auch Ungenauigkeiten in den Untersuchungsergebnissen verursacht. Die Abweichung der zu bestimmenden Größe (in diesem Beispiel der Fehlererkennungsgrad) ist abhängig von der tatsächlich ausgewählten Prüfmenge (Agrawal 1990).

3.6.2 SIMULATION AUF HÖHEREM ABSTRAKTIONSLEVEL

Eine weit bekannte Methode zur Beschleunigung der Fehlersimulation ist die Simulation auf einem höheren Abstraktionslevel. Teile der simulierten Schaltung werden beispielsweise in einer Beschreibungssprache formuliert, die sich für analoge Schaltungen eignet (Verilog-AMS oder VHDL-AMS), oder durch vereinfachte SPICE-Modelle dargestellt. Der Simulator kann die einzelnen Simulationsläufe beschleunigen, indem er Teile der Schaltung (z.B. Komponenten ohne Fehlerinjektion) durch Verhaltensmodelle auf höherer Ebene und/oder durch effizientere Modelle ersetzt, so dass insgesamt simuliert werden kann. In den meisten Fällen reduziert sich dadurch die Simulationsdauer. Die Simulation von Fehlern in SPICE-Netzlisten und die gleichzeitige Ausführung von Verilog/VHDL muss vom

Schaltungssimulator unterstützt werden. Die Art der injizierten Fehler ändert sich ebenfalls in Abhängigkeit von der Abstraktionsebene.

3.6.3 PARALLELE SIMULATION EINER SCHALTUNG

Die Simulation auf mehreren physischen oder mehreren logischen CPU-Kernen kann die Schaltungssimulation ebenfalls beschleunigen. Hierbei können mehrere CPU-Kerne jeweils Teile einer fehlerinjizierten Schaltung gemeinsam simulieren. Diese gemeinsame Simulation einer Schaltung kann nur sinnvoll durchgeführt werden, wenn die CPU-Kerne untereinander synchronisiert sind. Die parallele Simulation einer Schaltung ist viel schwieriger zu realisieren als die parallele Simulation der verschiedenen Fehlerfälle. Im Fall der Fehlersimulation können die fehlerinjizierten Schaltungen separaten simuliert werden. Da jeder CPU-Kern einen anderen Fehlerfall simuliert und damit so gut wie keine Interaktion zwischen den Simulationsläufen notwendig ist, kann die Fehlersimulation sehr effizient durchgeführt werden.

Komponente	Umgebungstemperatur	Sperrschichttemperatur	Leistungsstress	Spannungsstress	Stromstress	Durchschlagspannung	Technologie	Komplexität	Gehäuseform	Anwendungsart	kompakte Konstruktion-	Produktreihe	Produktionsdauer	Umwelteinflüsse	Produktqualität
Digitale und lineare ICs		D		x	x	x	x	x	x				x	x	x
Hybride Schaltungen	D	D	D	D	D	x	x	x	x	x	x	x	x	x	x
Bipolar Transistoren		D	D	x		x	x		x	x	x	x	x	x	x
FETs		D	D	x		x	x		x	x	x		x	x	x
Dioden		D	x	x	x	x	x		x	x	x	x	x	x	x
Optoelektronische Komp.		D		x	x		x	x	x				x	x	x
Widerstände	D		D				x					x	x	x	x
Kondensator	D			D			x		x	D		x	x	x	x
Spulen, Transformatoren	D		x	x			x						x	x	x
Relais, Schalter	D			x	x		x	x	x	x	D		x	x	x
(Steck)Verbinder	D				x		x		x	x	D	x	x	x	x
D: Dominanter Parameter x: wichtiger Parameter															

Tabelle 10. Wichtige Parameter, die die Komponenten einer Schaltung beeinflussen. Basierend auf (Birolini 2014, S. 32).

3.6.4 NACH AUFTRITTSWAHRSCHEINLICHKEIT VON FEHLERARTEN

Einige Studien haben die Auftrittswahrscheinlichkeit von einzelnen Fehlerarten in verschiedenen Bauelementen untersucht. In Tabelle 10 sind die relevantesten Parameterarten festgelegt, welche die Funktion bzw. den Ausfall der Schaltungskomponenten beeinflussen. Keine der aufgelisteten Komponenten weist lediglich einen Stressparameter auf, der einen Ausfall der Komponenten verursachen kann. Es ist stets eine Kombination von mindestens einem bzw. mehreren dominanten (D) und weiteren wichtigen (x) Parametern. Tabelle 11 liefert eine Übersicht zur statistischen Verteilung zwischen den Fehlerarten und den Bauelementen. Daraus wird ersichtlich, dass 60-100% der Fehlerarten bei analogen Komponenten auf Kurzschlüsse und Unterbrechungen zurückzuführen sind. Somit kann die Fehlersimulation ggf. auf die wahrscheinlichsten Fehlerarten (z. B. 70%) der einzelnen Komponenten beschränkt werden. Die weniger wahrscheinlichen Fehlerarten werden bei den Fehlerinjektionen ausgeschlossen. Die Angaben in der Tabelle 11 sind als ungefähre Indikatoren zu interpretieren und können je nach Anwendungsbereich variieren.

Bauelement/Komponente		Kurzschluss	Unterbrechung	Abweichen der Parameterwerte	Funktionsstörung
Digitale bipolare ICs		50	30	–	20
Digitale MOS ICs		20	60	–	20
Lineare ICs		–	25	–	75
Bipolarer Transistor		85	15	–	–
FETs		80	15	5	–
Diode (Si)	Allgemein	80	20	–	–
	Zener	70	20	10	–
Thyristor		20	20	50	10
Optoelektronik (Optokoppler)		10	50	40	–
Widerstand	fest	–	40	60	–
	variabel	–	70	20	10
Kondensator	Folien	15	80	5	–
	Keramik	70	10	20	–
	TA(fest)	80	15	5	–
	AI (flüssig)	30	30	40	–
Spule		20	80	–	–
Relais (elektromechanisch)		20	–	–	80
Quarzkristall		–	80	20	–

Tabelle 11. Statistische Richtwerte für Fehleraten bei elektronischen Komponenten in %. Basierend auf (Birolini 2014, S. 100).

3.6.5 FRÜHZEITIGER ABBRUCH EINER FEHLERSIMULATION

Bei der Fehlersimulation einer Schaltung kann für alle zu messenden Parameterwerte ein Toleranzbereich festgelegt werden. Dieser Bereich kann durch die Simulation der fehlerfreien Schaltung ermittelt werden, wobei für jeden simulierten Zeitpunkt sofort geprüft wird, ob eine der berechneten Größen ihren Toleranzbereich überschreitet. Liegt ein analoger Prüfparameter über oder unter den festgelegten Limits des Toleranzbereiches, kann auf die Prüfung der späteren Simulationszeitpunkte verzichtet und daher die Simulation für diesen Fehlerfall abgebrochen werden. Die konkrete Reduktion der Simulationszeit hängt von der Fehlermenge und der untersuchten Schaltung bzw. der vorliegenden Parameterwerte ab. Dieses Verfahren fand schon in der transienten Fehlersimulation u.a. in (Junwei Hou 1998; Junwei Hou und Chatterjee 2003) Anwendung.

3.6.6 MONOTONIE-REGEL BEI MEHRFACHFEHLERN

In dieser Arbeit wird bei der Fehlersimulation mit Mehrfachfehlern die Monotonie-Annahme vorausgesetzt. Die Monotonie-Regel besagt: Wenn ein Einfachfehler von einer Schaltung nicht toleriert wurde, dann wird auch die Kombination dieses Fehlers mit einem weiteren Fehler (Doppelfehler) nicht toleriert. Unter „toleriert“ wird verstanden, dass die Werte der bewerteten Größen der geprüften Schaltung noch innerhalb des tolerierbaren Wertebereiches (z.B. innerhalb des Toleranzbereiches) liegen. Entsprechend gilt für Mehrfachfehler, welche der Monotonie-Annahme entsprechen: Wenn ein n-facher Fehler nicht toleriert wird, dann wird auch ein (n+1)-facher Fehler, der den n-fachen Fehler enthält, nicht toleriert. Auch wenn die Monotonie-Regel nicht für alle Schaltungen bzw. alle Fehler erfüllt ist, so sind in der Praxis auftretende Systeme in den meisten Fällen monoton. Vereinfacht ausgedrückt: Eine Verschlechterung der Komponentenzustände bedeutet meist keine Zuverlässigkeitsverbesserung des betrachteten Systems (Heidtmann 1997, 92).

Da die ausgeschlossenen Mehrfachfehler-Kombinationen als nicht-toleriert (negative Fehlersimulation) festgelegt werden, kann die Vorgehensweise entsprechend der Monotonie-Annahme als pessimistisches Verfahren bezeichnet werden. Bei der praktischen Anwendung der Fehlersimulation von analogen Schaltungen existieren nur einzelne Ausnahmefälle von der Monotonie-Regel. Ein Beispiel für die Ausnahme von der Monotonie-Regel bei Mehrfachfehlern ist in Kapitel 3.3 Abbildung 10 zu finden. Im Vergleich zu analogen Fehlermodellen gilt bei digitalen Fehlermodellen (binären Fehlern) näherungsweise eine 50:50-Chance, dass sich zwei

Fehler gegenseitig aufheben (Liu 1991, S. 165–166). Die Monotonie-Annahme bei analogen Schaltungen hat den großen Vorteil, dass viele nichtrelevante Mehrfachfehler bereits vor der Fehlersimulation vernachlässigt werden können. Die Reduktion der Fehlermenge ist auffallend groß. Das Ausschließen von Doppelfehlerkombinationen reduziert ebenfalls die Fehlermenge der Dreifachfehlerkombinationen usw.

Abbildung 13 zeigt deutlich, wie Mehrfachfehler und mehrere Fehlerarten die Simulationsdauer (hier: Anzahl der Simulationsläufe) beeinflussen. Das Diagramm zeigt eine mittelgroße Schaltung mit insgesamt 20 Komponenten und jeweils zwei Fehlerarten. Die blaue Linie zeigt die Anzahl aller Kombinationsmöglichkeiten in Abhängigkeit von der Fehlertiefe (Anzahl der Mehrfachfehler). Die rote gestrichelte Linie zeigt die entsprechende Kurve für den Fall, dass ab Fehlertiefe 1 nur noch die 10 Komponenten betrachtet werden, deren Einzelfehler toleriert worden sind. Es ist erkennbar, dass die Anzahl der Simulationsläufe und damit auch die Simulationszeit signifikant reduziert werden kann, wenn diese Monotonie-Annahme zugrunde gelegt wird. Bei der Fehlertiefe 5 sind nur noch 8.064 Fehlerfälle anstatt 496.128 zu simulieren.

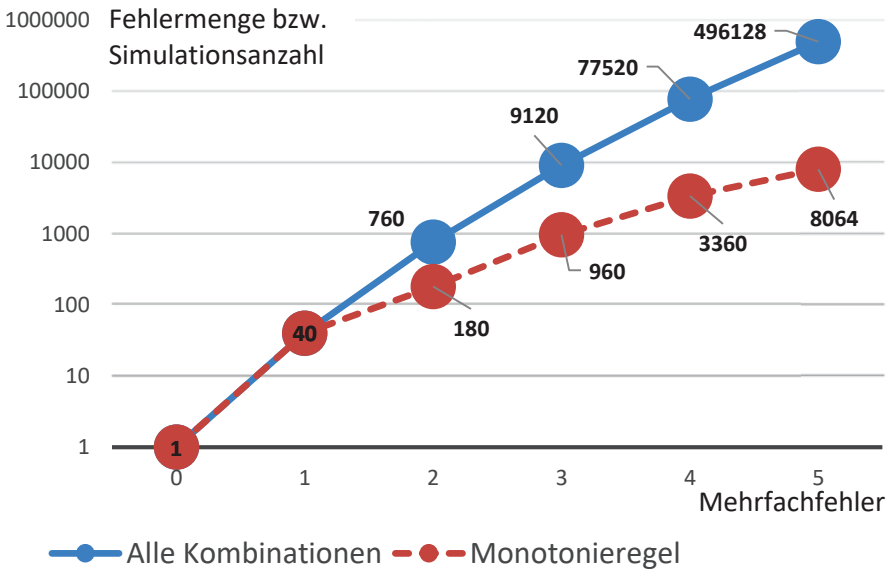


Abbildung 13. Komplexität der Fehlersimulation für eine mittelgroße Schaltung (20 Komponenten mit jeweils 2 Fehlerarten).

3.6.7 KOMBINATION MEHRERER BESCHLEUNIGUNGSTECHNIKEN

In der folgenden Tabelle 12 wird eine Übersicht zu dem in Kapitel 3.6 beschriebenen Techniken gezeigt. Die Spalten „Min“ und „Max“ geben einen Bereich des erwarteten Simulations-Beschleunigungsfaktors an, wobei diese Faktoren auf Erfahrungswerten beruhen. Es können große Abweichung in der erzielten maximalen Beschleunigung der Fehlersimulation vorkommen, die von der untersuchten Schaltung abhängen. In praktischen Anwendungen würde nicht jede Schaltung die maximal mögliche Beschleunigung erzielen. Neben den genannten Techniken können noch weitere anwendungsspezifische Techniken angewendet werden, auf die hier nicht eingegangen werden soll. Eine Übersicht weiterer Techniken sind u.a. bei (Sunter et al. 2015; Dhifi 2003; Sunter 2016) zu finden.

Technik	Min	Max
Zufällig ausgesuchte Stichprobe	10	>100
Simulation auf höheren Abstraktionslevel	2	30
Parallele Simulation einer Schaltung	1	1,5
Nach Auftrittswahrscheinlichkeit von Fehlerarten	1	1,5
Früher Abbruch einer Fehlersimulation	1	>100
Monotonie-Regel bei Mehrfachfehlern	2	>100
Fehlerklassen-Algorithmus (FKA) bei Mehrfachfehlern (Kapitel 4.1)	3	>100

Tabelle 12. Bewertung mehrerer Techniken zur Beschleunigung der Fehlersimulation. Basierend auf (Sunter et al. 2015; Junwei Hou und Chatterjee 2003) und Ergebnissen der Bewertung aus Kapitel 5.

4 FEHLERKLASSEN-ALGORITHMUS (FKA)

In diesem Kapitel wird der Fehlerklassen-Algorithmus (FKA) (Kapitel 4.1) und der implementierte Fehlersimulator (Kapitel 4.2) im Detail beschrieben. Der FKA ist eine effiziente Methode der Mehrfachfehlersimulation für analoge Schaltungen und stellt die Hauptneuerung und den wesentlichen wissenschaftlichen Beitrag der vorliegenden Arbeit dar. Der implementierte Fehlersimulator ist in dieser Arbeit mit der skriptbasierten Fehlermodellierung ein wissenschaftlicher „Nebenbeitrag". Der Fehlersimulator wird – als Mittel zum Zweck – benötigt, um die Effizienz des FKA anhand einer Reihe ausgewählter Schaltungen zu untersuchen.

4.1 EFFIZIENTER ALGORITHMUS BEI MEHRFACHFEHLERSIMULATION

Eine neue Methode zur Mehrfachfehlersimulation ist der Fehlerklassen-Algorithmus (FKA). Dabei handelt es sich um eine systematische Vorgehensweise zur Reduktion des Prüfaufwandes bei der Fehlersimulation von Mehrfachfehlern. Dazu werden Mengen gleicher Fehlerarten in so genannten Fehlerklassen (FK) zusammengefasst. Für einen Teil der Fehlerklassen kann das Simulationsergebnis in Form des Toleranzgrades mit guter Präzision interpoliert werden. Daher müssen die interpolierten Fehlerklassen nicht durch eine zeitaufwändige Fehlersimulation untersucht werden. Diese Technik erlaubt basierend auf den Ergebnissen der Mehrfachfehlersimulation (z.B. von Zweifach-Fehlern) eine effiziente Mehrfachfehleranalyse (z.B. von Dreifach-Fehlern).

4.1.1 DEFINITIONEN DES FEHLERKLASSEN-ALGORITHMUS

Eine Fehlerklasse (FK) wird als eine Teilmenge der Menge der Mehrfachfehler (mindestens Doppelfehler, Fehlertiefe = 2) definiert. Einfachfehler können keine Fehlerklasse bilden. Da die Fehler einer Fehlerklasse im Zuge der Fehlersimulation in eine analoge Schaltung injiziert werden, kann eine Fehlerklasse auch als eine Menge von Fehlerinjektionen (anstatt Fehlern) aufgefasst werden.

Jede Fehlerklasse hat stets eine feste Anzahl von Fehlern (z.B. 3 gleichzeitig auftretende, d.h. gleichzeitig zu injizierende Fehler) und beinhaltet Fehlerinjektionen (FI) mit gleichen Fehlerarten (z.B. 2 mal „open“, d.h. Unterbrechnug, und 1 mal „short“, d.h. Kurschluss). Eine FK legt nicht fest, in welche konkreten Bauelemente (z.B. Komponenten einer analogen Schaltung) die Fehlerarten injiziert werden. Eine Fehlerklasse wird individuell und unabhängig von Komponentenart und Fehlerort definiert. Allgemeine Fehlerarten wie Kurzschlüsse oder Unterbrechungen können in unterschiedliche Arten von Bauelementen injiziert werden. Diese Fehlerinjektionen werden dennoch in einer Fehlerklasse zusammengefasst. Parameterfehler hingegen können sehr vielfältig sein und lassen sich nur in bestimmten Fällen so abstrahieren, dass sie auf unterschiedliche Arten von Bauelementen anwendbar sind. Allgemein gilt: Wenn sich eine Fehlerart in gleicher Weise auf mehrere Arten von Bauelementen anwenden lässt, wird sie als eine Fehlerart behandelt und nicht nach Bauelement-Art unterschieden.

Alle Fehlerklassen (FK_n) werden durch die zu injizierenden Fehlerarten definiert, wobei manche Fehlerarten in einer bestimmten Anzahl auftreten können. Beispielsweise steht $FK_2(x, y)$ für alle Doppel-FI mit der Fehlerart x und y. Die Fehlerklasse $FK_3(x, y, y)$ umfasst alle Dreifachfehler, wovon ein Fehler der Art x und zwei Fehler der Art y sind. Auf die Reihenfolge der Nennung der Fehlerarten kommt es nicht an. Die Fehlerklassen $FK_2(x, y)$ und $FK_2(y, x)$ sind identisch, weil eine eindeutige Sortierung der Fehlerarten (z. B. Injektion der identischen Fehlerart in unterschiedliche Elemente) mit $x \leq y$ für $FK_2(x, y)$ vorausgesetzt wird. Allgemein wird für jede Fehlerklasse $FK_n(x_1, x_2, \ldots, x_n)$ gefordert, dass $x_1 \leq x_2 \leq \ldots \leq x_n$ ist. Somit wird eine Fehlerklasse für Doppel-FI (I_2) als die Menge an potentiellen Doppel-FI: $FK_2(x, y) = \{\{(f, e), (f', e')\} \in I_2 : f = x, f' = y\}$ definiert. Darin bedeutet ein Tupel (f, e) die Injektion von Fehlerart f in ein Bauelement e, wobei e und e’ unterschiedliche Bauelemente einer analogen Schaltung

sind. Eine Fehlerklasse für die Injektion von n Fehlern wird folglich definiert als: $FK_n(x_1,\ldots, x_n) = \{\{(f_1, e_1),\ldots, (f_n, e_n)\} \in I_n : f_i = x_i\}$. Weitere Definitionen sind in der formalen Beschreibung in Kapitel 4.1.3 zu finden.

Die Teilmenge der FK, die bei der Fehlersimulation einer FK toleriert werden, wird als Toleranzklasse (TK) bezeichnet. $TK_n(x_1, \ldots, x_n)$ ist die Teilmenge der tolerierten FI der Fehlersimulation aus der Fehlerklasse $FK_n(x_1, \ldots, x_n)$. Der Toleranzgrad t ergibt sich aus der Anzahl der FI aus TK dividiert durch die Anzahl aller FI der FK (zu TK gehörenden Fehlerklasse). Der Quotient aus $|TK_n(x_1, \ldots, x_n)|$ und $|FK_n(x_1, \ldots, x_n)|$ ist der Toleranzgrad $t(FK_n(x_1,\ldots, x_n) = t(FK)$, siehe folgende Formel 4.1.

$$t(FK_n(x_1, \ldots, x_n)) = \frac{|TK_n(x_1, \ldots, x_n)|}{|FK_n(x_1, \ldots, x_n)|} \qquad (4.1)$$

Unter Fehlerdistanz d(FK, FK') (ähnlich der Hamming-Distanz) wird die Anzahl der unterschiedlichen Fehlerarten von zwei FK verstanden. Wenn man die Fehlerarten einer Fehlerklasse als Multimengen auffasst, lässt sich die Fehlerdistanz d(FK, FK') mit $FK = FK_n(A)$ und $FK' = FK_n(B)$ formal definieren: $d(FK, FK') = |A| - |A \cap B|$. Dabei bezeichnet „$\cap$“ die Durchschnitt-Mulimenge und „|...|“ die Kardinalität einer Multimenge. Man beachte, dass die Fehlerdistanz zwischen zwei Fehlerklassen nur definiert ist, wenn sie die gleiche Fehlertiefe n aufweisen. Anmerkung: Die Definition $d(FK, FK') = |A| - |A \cap B|$ wäre gleichwertig, weil A und B wegen der übereinstimmenden Fehlertiefe die gleiche Elementanzahl aufweisen.

Beispielsweise ist die Fehlerdistanz $d(FK_5(x, y, y, z, z), FK_5(x, x, y, y, y)) = 2$. Dies lässt sich intuitiv damit begründen, dass in beiden Vektoren drei Fehlerarten (x, y, y) übereinstimmen und zwei Fehlerarten (z, z) sich unterscheiden. Formal kann die Fehlerdistanz für dieses Beispiel wie folgt bestimmt werden: $FK_5(x, y, y, z, z) = FK_5(A)$ mit $A = \{x, y, y, z, z\}$. Für die zweite Fehlerklasse gilt $FK_5(x, x, y, y, y) = FK_5(B)$ mit $B = \{x, x, y, y, y\}$. Die Durchschnitt-Mulitmenge ist $A \cap B = \{x, y, y\}$. Daraus ergibt sich: $d(FK_5(A), FK_5(B)) = |A| - |A \cap B| = 5 - 3 = 2$.

Der Fall d(FK1, FK2) = d(FK2, FK3) = 1 bedeutet, dass sich jedes Paar der Fehlerklassen nur in einer Fehlerart unterscheidet.

Ein Vektor $(FK1_n, \ldots, FKm_n)$ aus m Fehlerklassen $FK1_n, \ldots, FKm_n$ der Fehlertiefe n wird als Fehlerklassenkette (kurz: FK-Kette) bezeichnet. Eine

FK-Kette aus drei Fehlerklassen wird als Dreierkette bezeichnet. Man beachte, dass alle Fehlerklassen einer Kette die gleiche Anzahl von Fehlern (Fehlertiefe) aufweisen müssen.

Die in Kapitel 5.1 **Fehler! Verweisquelle konnte nicht gefunden werden.**untersuchten analogen Schaltungen zeigen, dass es viele Dreierketten ($FK1_n$, $FK2_n$, $FK3_n$) für Mehrfachfehler mit einer Fehlerdistanz $d(FK1_n, FK2_n) = 1$ und einer Fehlerdistanz $d(FK2_n, FK3_n) = 1$ gibt, die einen monoton steigenden/sinkenden Toleranzgrad (siehe Formel 4.1), aufweisen, d.h. $|TK1_n| / |FK1_n| \leq |TK2_n| / |FK2_n| \leq |TK3_n| / |FK3_n|$ bzw. $|TK1_n| / |FK1_n| \geq |TK2_n| / |FK2_n| \geq |TK3_n| / |FK3_n|$. Wenn diese Eigenschaft erfüllt ist, wird die Dreierkette als M-Kette bezeichnet (wobei „M“ auf die Monotonie hinweist).

Wenn die Fehleranzahl n aus dem Kontext eindeutig ist oder keine Rolle spielt, wird sie nicht genannt. Beispielsweise wird eine FK-Kette ($FK1_n$, $FK2_n$, $FK3_n$) dann mit (FK1, FK2, FK3) bezeichnet.

Wichtig: Es können nicht alle Fehlerklassen mit einer Fehlerdistanz von 1 zu einer M-Kette zusammengefasst werden. Es hängt vielmehr von der Reihenfolge der Fehlerklassen ab. Eine FK-Kette ist als „Kandidat“ für eine M-Kette auszuwählen, wenn die mittlere Fehlerklasse (FK2) aus allen gemeinsamen Fehlerarten und allen unterscheidenden Fehlerarten von FK1 und FK3 kombiniert wurde. Daraus ergibt sich ein bestimmtes Muster der kombinierten Fehlerarten der FK-Kette. Eine oder mehrere Fehlerarten sind bei allen FK der FK-Kette gleich, eine Fehlerart nimmt bei der FK-Kette ab und eine andere nimmt zu. Weitere Details zu diesem Muster sind in Kapitel 5.3 zu finden. Beispielsweise ist die FK-Kette ((x, x, z), (x, y, z), (y, y, z)) ein Kandidat für eine M-Kette.

Kandidaten für eine M-Kette, die entsprechend dem genannten Muster gebildet sind, werden als m-Ketten bezeichnet. Wie sich später zeigen wird, sind m-Ketten fast immer M-Ketten. Wenn dies in seltenen Fällen nicht der Fall ist, dann wird die Monotonie meist nur geringfügig verletzt.

Die Bildung von m-Ketten ist in Abbildung 14 und in dem folgenden Beispiel für eine Dreierkette (FK1, FK2, und FK3) mit jeweils drei Fehlerarten (FA) dargestellt:

FK1 (FA: 3× open), FK2 (FA: 2× open, 1× short) und FK3 (FA; 1× open, 2× short) sind Mengen von FI mit den genannten kombinierten Fehlerarten. Die Fehlerdistanzen betragen d(FK1, FK2) = d(FK2, FK3) = 1 bzw. d(FK1,

FK3) = 2, woraus folgt, dass die Bedingungen für eine m-Kette erfüllt sind. Wenn die Fehlersimulation bestätigt, dass es sich um eine M-Kette handelt, liegt ein steigender ($t(FK1) \leq t(FK2) \leq t(FK3)$) oder ein fallender ($t(FK1) \geq t(FK2) \geq t(FK3)$) Toleranzgrad vor.

Mit den folgenden 3 Schritten können m-Ketten, d.h. FKA-Ketten mit monoton steigendem bzw. fallendem Toleranzgrad gebildet werden:

1. Analyse der Menge mit Fehlerinjektionen nach m-Ketten, d.h. nach Fehlerklassen-Ketten, die die Bedingung der Fehlerdistanz und der Reihenfolge erfüllen. Es werden also die FK-Ketten (z.B. Dreierketten) einer bestimmten Fehlertiefe gesucht, die untereinander eine Fehlerdistanz von 1 aufweisen und deren Reihenfolge dem genannten Muster entspricht. Allgemein sind auch längere Ketten als eine Dreierkette möglich.
2. Entscheidung darüber, welche der gefundenen FK-Ketten auf einen auf- bzw. absteigenden Toleranzgrad hinausführen, d.h. als m-Kette herausgegriffen werden. Für die Entscheidung auf der Ebene der Fehlertiefe n werden die Fehlerklassen der Fehlertiefe n – 1 analysiert. Als Beispiel kann die folgende Abbildung 14 betrachtet werden, die zur Bildung einer Dreierkette führt. Die detaillierte Vorgehensweise zu Schritt 2 ist in Kapitel 4.1.2 zu finden.
3. Bestimmung des Toleranzgrades der ersten und der letzten Fehlerklasse einer Dreierkette (z. B. FK1 und FK3) durch eine Fehlersimulation. Die Bestimmung des Toleranzgrades der verbleibenden Fehlerklassen (z.B. FK2) in der Mitte der Kette erfolgt durch Interpolation aus den Toleranzgraden von FK1 und FK3. Hierdurch wird die rechenintensive Simulation einer Fehlerklasse gespart, woraus sich der Aufwand der Simulation einer FK-Kette reduziert – grob gerechnet um 1/3, weil nur noch zwei von drei Fehlerklassen zu simulieren sind. Die Reduktion des Gesamtaufwandes dagegen kann höher sein, weil sich mehrere FK-Ketten finden lassen, wo die erste und letzte Fehlerklasse (FK1 bzw. FK3) identisch ist und nur einmal durch Simulation analysiert werden muss. Eine detaillierte Betrachtung der Effizienz und eine Bewertung der Genauigkeit der Interpolation sind in Kapitel 5.2 zu finden.

In der folgenden Abbildung 14 sind die Schritte für das Aufstellen der m-Ketten und die Bestimmung der Reihenfolge der Fehlerklassen in den m-

Ketten grafisch dargestellt. Hinweis: MFK_2 ist eine Menge von Fehlerklassen mit jeweils 2 Fehlerarten, MFK_3 ist die Menge von Fehlerklassen mit drei Fehlerarten.

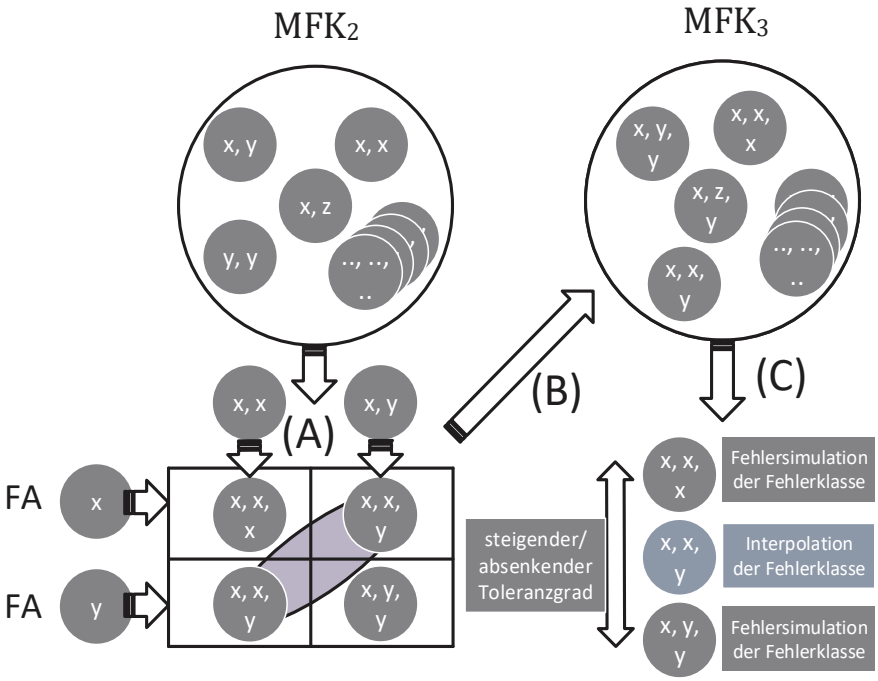


Abbildung 14. Vorgehen für das Aufstellen von der MFK_3 Menge der Fehlerklassen FK_3 aus den Ergebnissen von MFK_2.

Die einzelnen Schritte zur Kombination von Fehlerklassen FK_3 aus MFK_3 der Fehlertiefe 3 zwecks Bildung einer m-Kette sind:

A. Die Kombination von zwei Fehlerklassen $FK_2(x, y)$ und $FK_2'(x, x)$ aus MFK_2, wobei die Fehlerdistanz zwischen diesen 1 betragen muss, mit zwei Arten von Einfachfehlern (x und y) ergibt drei Fehlerklassen aus MFK_3, nämlich $FK_3(x, x, x)$, $FK_3(x, x, y)$ und $FK_3(x, y, y)$. Daraus erkennt man auch, wie die Reihenfolge in der m-Kette entsteht.

B. Die Kombination von Fehlern aus der Menge von MFK_2 und einem Fehler einer weiteren Art ergibt sich die Fehlermenge MFK_3 der Fehlerklassen FK_3.

C. Die in Schritt (A) aufgestellten Fehlerklassen und die gemäß FKA gebildeten m-Ketten erlauben die Bestimmung des Toleranzgrads der Fehlerklassen in der Mitte der m-Ketten durch Interpolation.

Durch die selben Schritte (A bis C) würden sich Fehlerklassenmengen MFK_4, MFK_5 usw. bilden lassen und genauso die m-Ketten aus FK_4 FK_5 usw.

4.1.2 IMPLEMENTIERUNG VON FKA

Der Ansatz des Fehlerklassen-Algorithmus (FKA) wird in den folgenden Schritten (Schritt 1 bis Schritt 3) im Detail beschrieben und der Algorithmus in Abbildung 15 dargestellt. Es wird davon ausgegangen, dass die Toleranzklassen $TK_1(\ldots)$ und $TK_2(\ldots)$ für die Fehlertiefen 1 bzw. 2 bereits vom Fehlersimulator bestimmt worden und daher bekannt sind. In den folgenden drei Schritten wird beschrieben, wie die Fehlerklassen $FK_3(\ldots)$ für Dreifach-Fehlersimulation bzw. Interpolation gebildet werden.

Schritt 1 - Generierung der Fehlerklassen

Eine Fehlerklasse $FK_3(x, y, z)$ mit drei Fehlern wird durch die Kombination aller FI von TK_2 mit allen FI von TK_1 in der folgenden Weise erzeugt: Jede Kombination von $tk_2 \in TK_2(x, y)$ und $tk_1 \in TK_1$ (z) bildet eine Fehlerinjektion $fk_3 \in FK_3(x, y, z)$ mit den Fehlerarten x, y und z, die jeweils nacheinander in alle Bauelemente zu injizieren sind. Das bedeutet, dass keine zwei Fehlerarten (z.B. x und y) zugleich in das gleiche Bauelement injiziert werden. Hinweis: Unter dem Begriff „Kombination“ im vorletzten Satz ist eine Multimengen-Vereinigung zu verstehen, d.h. eine Mengenvereinigung, bei der ggf. manche Elemente mehrfach vorkommen können (engl. multiset).

Schritt 2 - Suchen und Bilden der Fehlerklassen-Ketten

Die Suche nach m-Ketten beginnt mit der Analyse von TK_2 (Fehlertiefe 2). Zuerst werden alle Paare von Toleranzklassen $TK_2(x, y)$ und $TK_2(x', y')$ verglichen. Sodann werden die Kombinationen (Paare) ausgewählt, die eine Fehlerdistanz von 1 und darüber hinaus einen signifikanten Unterschied im Toleranzgrad (Mindest-Differenz Δ) aufweisen. Hierfür gilt $d(TK_2(x, y), TK_2(x', y')) = 1$ und $|t_2(x, y) - t_2(x', y')| \geq \Delta$, wobei Δ im Bereich von ungefähr 5% der absoluten Werte liegen kann. Es soll jedoch keine allgemeine Empfehlung gegeben werden. Aus der Fehlerdistanz von 1 zwischen $TK_2(x, y)$ und $TK_2(x', y')$ können wir schließen, dass für die Fehlerart entweder $x = x'$ oder $y = y'$ gilt. Im Folgenden geht man ohne Beschränkung der Allgemeinheit davon aus, dass $x = x'$ und $y \neq y'$ gilt. Aus den Toleranzklassen $TK_2(x, y)$ und $TK_2(x, y')$ wird durch Kombination mit den zusätzlichen Fehlerarten y und y’ eine m-Kette der Fehlertiefe 3 gebildet: $(FK_3(x, y, y), FK_3(x, y, y'), FK_3(x, y', y'))$ gebildet. Das bedeutet, es wird eine Kombination mit den Fehlerarten gebildet, die sich in den ausgewählten Toleranzklassen ($TK_2(x, y)$ und $TK_2(x, y')$) unterscheiden. Durch die Fehlerdistanz von jeweils $d = 1$ unterscheiden sich die erste

und die letzte Fehlerklasse einer Dreierkette unabhängig von der Fehlertiefe stets bei zwei Fehlerarten, siehe auch Kapitel 5.3. Ausgehend von der Fehlertiefe 3 entstünde eine m-Kette für die nächsthöhere Fehlertiefe 4 beispielsweise ausgehend von $FK_3(x, y, z)$ und $FK_3(x, y, z')$ in Kombination der Fehlerarten $z \neq z'$ die Dreierkette ($FK_4(x, y, z, z)$, $FK_4(x, y, z, z')$, $FK_4(x, y, z', z')$).

Schritt 3 – Bestimmung des Toleranzgrades

Aus der Beobachtung des steigenden/fallenden Toleranzgrades bei Fehlerklassen-Ketten werden nur die FI der ersten und der letzten Fehlerklasse einer Dreierkette simuliert. Mit der Fehlersimulation von $FK_3(x, y, y)$ und $FK_3(x, y', y')$ werden die Toleranzklassen $TK_3(x, y, y)$ und $TK_3(x, y', y')$ und damit die Toleranzgrade $t_3(x, y, y)$ und $t_3(x, y', y')$ bestimmt. Der Toleranzgrad $t_3(x, y, y')$ der „inneren" Fehlerklassen in der Dreierkette wird durch Interpolation ermittelt – und zwar durch Bildung des arithmetischen Mittels:

$$t_3(x, y, y') = (t_3(x, y, y) + t_3(x, y', y')) / 2.$$

Der Algorithmus für die Suche der FK-Ketten und Bestimmung der Toleranzgrade für die Fehlertiefe (=2) ist in Abbildung 15 dargestellt.

Procedure: Search Fault Class Chains

```
for all pairs (TK2, TK2') of tolerance classes with two injections do
  if d(TK2(x, y), TK2(x', y')) = 1 and |t2(x, y) – t2(x', y')| ≥ Δ then
  {  TK = fault_simulation(FK3(x, y, y));
     TK'' = fault_simulation(FK3(x, y', y'));
     t3(x, y, y) = TK /FK3 (x, y, y);
     t3(x, y', y') = TK'' /FK3 (x, y', y');
     t3(x, y, y') = (t3(x, y, y) + t3(x, y', y')) / 2;
  }
```

Abbildung 15. Algorithmus für die Suche der Fehlerklassen-Ketten und Bestimmung des Toleranzgrades

Für die Fehlersimulation und die Verwendung des FKA für mehr als drei Mehrfachfehler können ebenfalls die Schritte 1 bis 3 angewendet werden. Der Übergang von 2-fach-Fehler auf 3-fach-Fehler ist der wichtigere, da 3-fach Fehler häufiger vorkommen als 4-fach Fehler oder gar Mehrfachfehler mit noch höherer Fehleranzahl.

Die Bewertung der Genauigkeit des Fehlerklassen-Algorithmus wird in Kapitel 5.2 (Tabelle 15) anhand einer Auswahl von Beispielschaltungen durchgeführt. Eine mögliche Erklärung des monotonen Verlaufes der FK-Ketten bezüglich des Toleranzgrades wird in Kapitel 5.3 analysiert.

4.1.3 FORMALE BESCHREIBUNG DER MEHRFACHFEHLERSIMULATION

Um eine bessere Präzision für die Beschreibung der FKA-Methode und der Fehlerklassen zu erreichen, wird das Verfahren in diesem Kapitel formal beschrieben. Die Beziehung zwischen den Fehlern, den Elementen einer Schaltung, den Fehlerinjektionen und den Simulationsläufen wird durch die folgenden Tupel und Attribute definiert:

1) $C = \{c_0, \dots, c_m\}$ bezeichnet die Menge der zu untersuchenden Schaltungen mit Fehlerinjektion, $c_0 \in C$ ist die fehlerfreie Schaltung.
2) E = {transistor1, transisitor2, ..., resistor1, ... } ist die Menge der Elemente der fehlerfreien Schaltung c_0.
3) F = {short_circuit, open_circuit, parameter_mod., ...} ist die Menge der unterschiedlichen Fehlerarten der Elemente einer Schaltung.
4) $I = \{(f, e) \in F \times E$: Fehler f ist in Element e möglich} ist die Menge der potentiellen Fehlerinjektionen.
5) $I^* = \{i^* \subset I : (x \in i^*, y \in i^*, x \neq y) \Rightarrow x|E \neq y|E\}$ ist die Menge der potentiellen Mehrfachfehlerinjektionen. I* ist eine Teilmenge aus der Potenzmenge von I. Mit x|E und y|E wird angegeben, in welches Element injiziert wird. Die Ungleichung $x|E \neq y|E$ schließt die gleichzeitige Injektion von verschiedenen Fehlern in das gleiche Element aus.
6) $Q : F \times E \rightarrow [0, 1]$ ist die Auftrittswahrscheinlichkeit eines Fehlers $f \in F$ in einem Element $e \in E$. Wenn die Fehlerart $f \in F$ auf das Element $e \in E$ nicht anwendbar ist, dann ist Q(f, e) = 0. Für ein fehlerhaftes Element $e \in E$ ist die Summe der Fehleraufttrittswahrscheinlichkeit immer $\Sigma_{f \in F}$: Q(f, e) = 1.

Beispiel: Angenommen es sind zwei Fehlerarten F = {open, short} möglich und es sind nur zwei Elemente $E = \{R_1, R_2\}$ im System vorhanden. Dann gibt es vier Einfachfehlerinjektionen $I = \{(open, R_1), (open, R_2), (short, R_1), (short, R_2)\}$ und vier Doppel-Fehlerinjektionen. Insgesamt ergibt sich dann eine potentielle Menge von acht Einfach- und Mehrfachfehlerinjektionen:

$$I^* = \{\{(open, R_1)\}, \{(open, R_2)\}, \{(short, R_1)\}, \{(short, R_2)\},$$
$$\{(open, R_1), (open, R_2)\}, \{(short, R_1), (short, R_2)\},$$
$$\{(open, R_1), (short, R_2)\}, \{(short, R_1), (open, R_2)\}\}.$$

Wenn Kurzschlüsse (engl. short circuit) wahrscheinlicher für R_1 sind und Unterbrechungen (engl. open circuit) für R_2 wahrscheinlicher sind, dann würden wir beispielsweise Folgendes erhalten:
$Q(open, R_1) = 0.2$, $Q(short, R_1) = 0.8$ $(0.2 + 0.8 = 1)$ und $Q(open, R_2) = 0.4$, $Q(short, R_2) = 0.6$ $(0.4 + 0.6 = 1)$.

Der Wert der Funktion P : E → [0,1] gibt die Wahrscheinlichkeit an, dass ein Element e∈E nicht fehlerbehaftet ist und somit als fehlerfrei anzusehen ist.

Die Funktion R: I*→ {0,1} gibt das Resultat für jeden Simulationsdurchlauf mit allen Mehrfachinjektionen i ∈ I* an. Wenn der injizierte Fehler von der Schaltung (in Abhängigkeit von den definierten Toleranzkriterien) toleriert wird, ist das Ergebnis 1, andernfalls 0. Im Folgenden wird der Simulationsablauf für Einfachfehler, Doppelfehler und Dreifachfehler unter der Anwendung der Monotonie-Regel für Mehrfachfehlersimulation beschrieben.

Fehlersimulation mit Einfachfehlern:

$I_1 = I$ ist eine Menge aller Einfachfehlerinjektionen, die durch Fehlersimulation evaluiert werden. $T_1 = \{i \in I_1: R(\{i\}) = 1\}$ ist eine Menge von tolerierten Einfachfehlerinjektionen und entspricht der Toleranzklasse aus Kapitel 4.1.2.

Doppelfehlerinjektionen:

$I_2 = \{\{(f, e), (f', e')\}: (f, e) \in T_1, (f', e') \in T_1, e \neq e'\}$ ist eine Menge von Doppelfehlerinjektionen, die durch Fehlersimulation geprüft werden. I_2 wird gemäß der Monotonie-Annahme (siehe auch Kapitel 3.6.6) auf der Basis von T_1 und nicht auf der Basis von I_1 definiert. D.h. alle nicht tolerierten Fehlerinjektionen ($I_1 \setminus T_1$) wurden ausgeschlossen und daher nicht berücksichtigt. Dadurch können viele Doppelfehlerkombinationen vernachlässigt werden. $T_2 = \{i^* \in I_2 : R(i^*) = 1\}$ ist eine Menge von Doppelfehlerinjektionen, die toleriert wurden.

Dreifachfehlerinjektionen:

$I_3 = \{\{(f, e), (f',e'), (f'',e'')\} : \{(f, e), (f', e')\} \in T_2, (f'', e'') \in T_1, e \neq e', e \neq e'', e' \neq e''\}$ ist eine Menge von Dreifachfehlerinjektionen, die durch die Fehlersimulation geprüft werden. Auch hierbei wird die Monotonie-Regel angewendet, wodurch die nicht tolerierten Fehlerinjektionen verworfen werden. $T_3 = \{i^* \in I_3 : R(i^*) = 1\}$ ist eine Menge von Dreifachfehlerinjektionen, die toleriert wurden. Ein Teil von T_3 kann durch FKA in hinreichender Genauigkeit durch Interpolation bestimmt werden.

Injektionen mit höherer Fehleranzahl:

Der Simulationsprozess von weiteren kombinierten Fehlerarten bis n-fach Fehlerinjektion I_n (maximale Komponentenanzahl) wird auf dieselbe Art und Weise durchgeführt:

$$I_n = \{\{(f_1, e_1), \dots, (f_n, e_n)\}: \{(f_1, e_1), \dots, (f_{n-1}, e_{n-1})\} \in T_{n-1}, (f_n, e_n) \in T_1, e_1 \neq e_2, e_2 \neq e_3, \dots, e_{n-1} \neq e_n, e_1 \neq e_n\}$$

4.2 ANALOGER FEHLERSIMULATOR

4.2.1 ÜBERLEGUNGEN ZUM ENTWICKELTEN FEHLERSIMULATOR

Wie bei einem Schaltungssimulator ist ein Fehlersimulator ein Werkzeug (Hardware und/oder Software-Tool), welches das Verhalten der Schaltung unter Fehlerinjektion für einen gegebenen Stimulus untersucht. Die simulierte Schaltung wird nicht nur durch ein fehlerfreies Modell, sondern auch durch mehrere bis viele Schaltungsmodelle, die jeweils einen oder auch mehrere injizierte Fehler enthalten, dargestellt. Alle Fehler, die in die Schaltung injiziert werden, wurden durch eine Fehler-Liste festgelegt. Die Liste wird entweder vor und teilweise auch während der Fehlersimulationen (z.B. bei Mehrfachfehlern unter Berücksichtigung der Monotonie-Regel) erstellt. Es gehört daher zu den vorbereitenden Schritten der Fehlersimulation, die Fehlerliste zu bestimmen. Abbildung 17 in Kapitel 4.2.4 zeigt den grundlegenden Ablauf einer Fehlersimulation mit Mehrfachfehlern. Dabei wird dem zu untersuchenden System unterstellt, dass mehrere Fehler zum selben Zeitpunkt auftreten. Um möglichst flexibel mehrere Fehlerarten pro Bauelement bei der Mehrfachfehlersimulation zu untersuchen, wurde ein spezifischer Fehlersimulator für analoge Schaltungen entworfen und implementiert. Die erweiterten Lösungsansätze zur Reduktion der Simulationszeit wurden in dem Fehlersimulator realisiert. Dieser soll insbesondere die Effizienz der Monotonie-Regeln (Kapitel 3.6.6) und des FKA (Kapitel 4.1) demonstrieren.

Im Folgenden werden die Herausforderungen der Fehlersimulation, insbesondere die Fehlermodellierung und der Umgang mit Komponententoleranzen, welche die Effektivität der Fehlersimulation sowie die Simulation von fehlerhaften Bauelementen bestimmen, nochmals hervorgehoben. Anders als bei der Analyse digitaler Systeme, ist eine vollständige Modellierung und Simulation fehlerhafter Bauelemente nicht denkbar. Schon bei Schaltungen mit wenigen Komponenten können die möglichen Fehlerarten sehr zahlreich bzw. unbegrenzt sein. Zum Beispiel kann ein fehlerhafter Widerstand eine unendliche Anzahl von Widerstandsgrößen aufweisen. In vielen Bereichen werden Komponentenmodelle für die allgemeine Schaltungssimulation erstellt. Realitätsgetreue Fehlermodelle eines Bauelementes sind manchmal nicht verfügbar und müssen für die Fehlerinjektion eigens erstellt werden. Eine große Schwierigkeit eines analogen Fehlersimulators ist die Berücksichtigung der Auswirkungen von bestimmten Bauteil-Toleranzen auf die Funktionalität einer Schaltung. Keine analoge Schaltung wird ohne Parametertoleranzen hergestellt, welche bei rein digitalen Schaltungen keine Rolle spielen würde.

Bei einer analogen Schaltung weichen im Gegensatz zu digitalen Schaltungen die Parameterwerte fast immer von den Sollwerten ab. Daher muss jeder analoge Fehlersimulator das Problem der Parametertoleranzen der Bauelemente berücksichtigen. Hinzu kommen aufgrund von Temperaturveränderungen oder verändertem Stromfluss die abhängigen Fehler und Folgefehler in weiteren Komponenten. Man beachte, dass sich daraus ein Bedarf für die Untersuchung auf Mehrfachfehler bzw. für die Mehrfachfehlersimulation ableitet.

4.2.2 ENTWICKLUNGSHINTERGRUND DES FEHLERSIMULATORS

Die Überprüfung des Systemverhaltens bei Vorhandensein von Störungen ist in allen Phasen des Entwicklungsprozesses notwendig, vorzugsweise vor erfolgter Inbetriebnahme. Folglich ist die Untersuchung des Systems mit Hilfe der Fehlersimulation ein wertvoller Ansatz für die Prüfung eines hochverfügbaren bzw. fehlertoleranten Systems in Bezug auf bestimmte Fehlerarten. Mit der Fehlersimulation soll die Prüfung der Systemfunktionsfähigkeit bei Vorhandensein von Fehlern realisiert werden, woraus auch Aussagen über die Zuverlässigkeit abgeleitet werden.

Bei der Entwicklung des in dieser Arbeit implementierten Fehlersimulators stand schon zu Beginn der Entwicklung fest, dass kein neuer Schaltungssimulator entwickelt werden soll, der womöglich effizienter arbeitet als die bisherigen Schaltungssimulatoren. Der Aufwand solch einer Entwicklung steht nicht im Verhältnis zum gewünschten Ziel der effizienten Fehlerinjektion. Daher wurde der Fehlerinjektor in Verbindung mit einem modernen und kommerziellen EDA-Schaltungssimulator (NI AWR Design Environment 2016) entwickelt. Der Fehlersimulator wurde in VB-Skript implementiert und besteht aus ca. 3300 Zeilen Programmcode. Er ist einfach zu nutzen und kann für beliebig komplexe analoge Schaltungen verwendet werden. Der Fehlersimulator wurde als eine Erweiterung bzw. eine separate Software-Schicht entwickelt (siehe Abbildung 16).

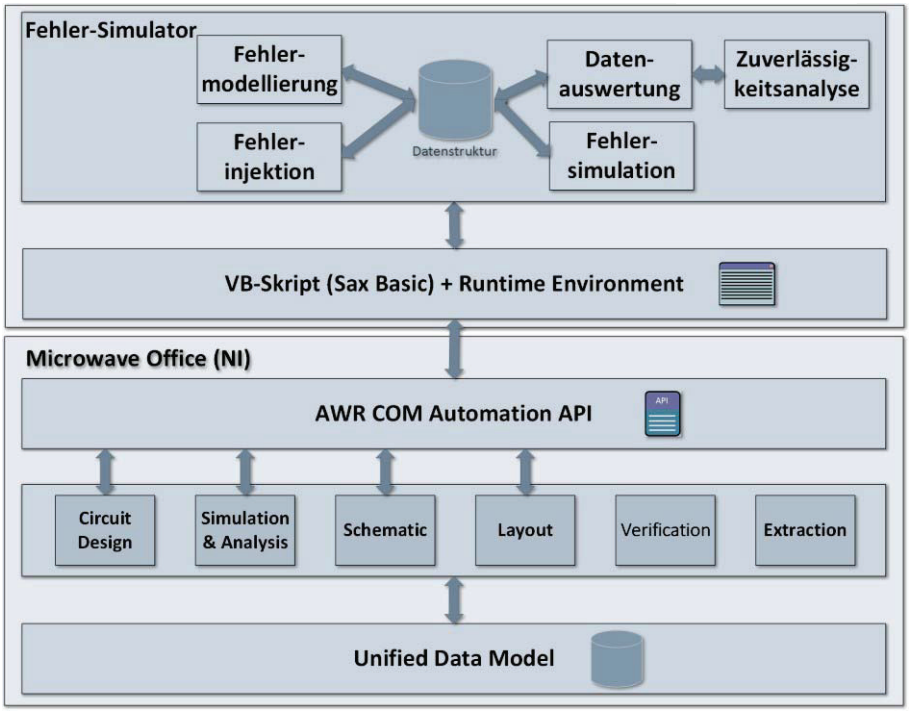


Abbildung 16. Schichtenarchitektur des Fehlersimulators

4.2.3 ABGRENZUNG VON ANDEREN FEHLERSIMULATOREN

Die Besonderheiten des in dieser Arbeit implementierten Fehlersimulators sowie die Unterschiede zu den in Kapitel 3.4 genannten Fehlersimulatoren sind in den folgenden Punkten zu sehen:

- Die Fehlermodellierung erfolgt nicht über SPICE bzw. Verilog-Netzlisten, sondern über skriptbasierte und objektorientierte Fehlermodellierung mittels VB-Skript. So wurden bisher stets Text-Parser bei analogen Fehlersimulatoren genutzt, um die SPICE-Netzlisten Zeile für Zeile einzulesen, die bauelementbezogenen Informationen zu verarbeitet und die Fehlerinjektionen (häufig in Form eines annähernd unendlich großen bzw. vernachlässigbar kleinen Widerstandes) als eine neuen SPICE-Netzliste darzustellen. SPICE-Netzlisten beinhalten grundsätzlich Informationen zu Bauelementen, Verbindungen von Kontakten (Pins), Parametern (z.B. Spannungswerte, Einstellungen der Bauelemente) oder Eingabe/Ausgabeparameter. Dieser Schritt der Umwandlung einer Eingabe in die Form einer SPICE-Netzliste ist bei dem hier entwickelten analogen Fehlersimulator nicht erforderlich.
- Die objektorientierte Programmierung (OOP) mit Klassen, Bauelement-Objekten mit spezifischen Eigenschaften und Methoden/Subroutinen (z.B. Fehlermodelle) wurde in dieser Arbeit für die Fehlermodellierung und die Fehlerinjektion eingesetzt. Die Vorteile sind die einfachere und zeitsparende Fehlermodellierung im Vergleich zu früheren Ansätzen (z.B. durch Text-Parser). Die Fehlermodelle gemäß OOP sind einfacher zu entwerfen, klarer strukturiert und allgemein einfacher in der Handhabung.
- Der Ansatz erlaubt die Fehlerinjektion in allen Bauelementen der analogen Schaltung. Durch die skriptbasierte Programmierung können sowohl komponentenbasierte als auch komponentenübergreifende Fehlermodelle erstellt und einfach wiederverwendet werden. Die verwendete Skriptsprache (VB-Skript) erlaubt beliebige Parameterfehler und die Programmierung beliebiger Fehlfunktionen der Bauelemente. Beliebige Parameterfehler und Fehlfunktionen definieren sich durch harte (z. B. Kurzschluss, Unterbrechung) oder weiche Fehlerarten (z. B. Parameterfehler). Das Fehlverhalten der einzelnen Schaltungskomponenten wird durch ein Skript, das die Fehlerinjektion beschreibt, ausgedrückt. Damit können sowohl exakte wie auch komplexe Fehlermodelle definiert werden und flexibel in Bauelemente bzw. Baugruppen injiziert werden.

- Die meisten auf Transistorebene arbeitenden Fehlersimulatoren (Kapitel 3.5) nutzen SPICE als Simulationskern. Der in dieser Arbeit entwickelte Fehlersimulator verwendet ein kommerzielles EDA-Tool mit den APLAC Simulator (NI AWR Design Environment 2016) für die Simulation der fehlerinjizierten analogen Schaltungen und ist als eine Erweiterung bzw. Softwareschicht des EDA-Tools implementiert. Der APLAC- Simulator stellt einen modernen, über die letzten 15 Jahre optimierten Schaltungssimulator dar, der beispielsweise die volle Leistungsfähigkeit von Multikern-PCs ausnutzen kann. Die detaillierten Eigenschaften des Simulators sind unter (NI AWR 2015) zu finden.
- Der in dieser Arbeit implementierte Fehlersimulator läuft vollautomatisch (erfordert also keine Benutzereingriffe), sobald die notwendigen Parameter (Fehlerarten, Fehlertiefe und Komponenten) ausgewählt wurden. Die Fehlersimulationen unterstützten sowohl Einfachfehler- als auch Mehrfachfehlerinjektion. Um die Gesamtprüfzeit bei Mehrfachfehlern zu reduzieren, wird die Monotonie-Regel und der Fehlerklassen-Algorithmus angewandt (siehe Kapitel 3.6.6 und 4.1).

4.2.4 FEHLERMODELLIERUNG UND FEHLERINJEKTION

Die Fehlermodellierung kann auf unterschiedliche Arten erfolgen. Jedoch ist stets zu beschreiben, wie sich der Fehler bzw. eine Fehlerart auf eine oder mehrere Komponenten (Bauelemente einer Schaltung) auswirkt. Dazu ist die fehlerbedingte Modifikation der normalen Verhaltensweise des betreffenden Bauelements zu beschreiben. Bei analogen Schaltungen handelt es sich bei den Fehlerarten gemäß ihren physikalischen Fehlerursprung um Kurzschlüsse, Unterbrechungen und verschiedene Veränderungen der Parameterwerte (siehe auch Kapitel 3.2, Tabelle 10 und Tabelle 11). So kann eine Fehlerart ein Kurzschluss sein, der zwischen zwei oder auch mehr Kontaktstellen eines Bauelements auftritt (beispielsweise ein Kurzschluss zwischen Basis und Emitter, Basis und Kollektor, Emitter und Kollektor oder allen drei Anschlüssen eines bipolaren Transistors zugleich). In Abbildung 24 (Kapitel 4.2.6) wird exemplarisch eine Fehlerart durch eine Subroutine in VB-Skript beschrieben. Die Routine greift aktiv auf das Modell der Schaltung zu und sorgt für die gewünschte Veränderung zwecks Fehlerinjektion. Die Fehlermodelle werden vor der Fehlerinjektion definiert, zentral gespeichert und können auch in Kombination als Mehrfachfehlerinjektion angewendet werden.

Die zweite wichtige Funktion der Fehlermodellierung ist die Aufstellung der Fehlerliste für die anschließende Fehlerinjektion. Die Generierung der Fehlerliste kann auf unterschiedliche Arten erfolgen, angefangen von der manuellen Auswahl einzelner Bauelemente bis hin zum automatisierten Verfahren der Zufallsstichprobenauswahl von Fehlern.

In vielen Publikationen sind alle angenommen Defekte bzw. Fehlerarten als gleich wahrscheinlich angenommen. Zur einfacheren Zuverlässigkeitsanalyse wird auch hier die gleichverteilte Auftrittswahrscheinlichkeit von Fehlern unterstellt. Diese Annahme erleichtert die statistische Datenanalyse. Jedoch zeigen Untersuchungen der aufgetretenen Fehlerarten von Bauelementen analoger Schaltungen (siehe Tabelle 11), dass die Gleichverteilung der Fehlerarten einer Komponente nicht als allgemeingültig angesehen werden sollte. Im Fehlerklassen-Algorithmus (FKA, siehe Kapitel 4.1.1) wird der Toleranzgrad unabhängig von den Auftrittswahrscheinlichkeiten der Fehlerarten bestimmt. Statistisch selten auftretende und vernachlässigbare Fehlerarten, die von der Fehlersimulation ausgeschlossen werden, führen zu einer kleineren Fehlerliste und damit auch zu einer kürzeren Analysedauer. Der entwickelte Fehlersimulator verwendet für die effiziente Fehlersimulation die Monoto-

nie-Regel (Kapitel 3.6.6) und FKA (Kapitel 4.1). Weitere Verfahren zur Beschleunigung der Fehlersimulation sind in Kapitel 3.6 zu finden. Mit dem Aufstellen der Fehlerliste ist die Fehlermodellierung quasi abgeschlossen. Während der Simulation unter Fehlerinjektion kann bei Mehrfachfehleranalysen die Fehlerliste auch stetig erweitert werden.

Der entwickelte Fehlersimulator nutzt die API des EDA-Tools und kann damit direkt auf die Schaltungen (engl. schematics) zugreifen. Dieser nutzt nur das Originalmodell der Schaltung als Input, injiziert die Fehlerarten und setzt die Fehlerinjektion nach der Fehlersimulation wieder zurück. Für die Fehlerinjektion wird die vom EDA-Tool bereitgestellte API genutzt (siehe auch Abbildung 16). Der API-Zugriff wird über die Microsoft COM-Automationsfähigkeiten (COM Server) auf dem Betriebssystemen Microsoft Windows realisiert. D.h. andere Anwendungen in MS Windows, die als ein Automationsdienst (COM Client) konfiguriert sind, können so den Schaltungssimulator über spezifische Befehle kontrollieren. Diese Fähigkeiten nutzt der entwickelte Fehlersimulator aus, um Steuerungsbefehle an den Schaltungssimulator zu senden. Das bedeutet, die Fehlerinjektion und die Beschreibung der Fehlerarten werden über einen objektorientierten Ansatz durchgeführt.

Die Steuerung der Fehlersimulation ist ein wichtiger Bestandteil der Anwendung. Sie steuert die Ausführung der Fehlerinjektion, das Auslesen der Simulationsdaten für die Datenanalyse und das Entfernen der Fehlerinjektion aus einer Schaltung. Dieser Kreislauf des gesteuerten Ablaufes ist in der Abbildung 17 dargestellt.

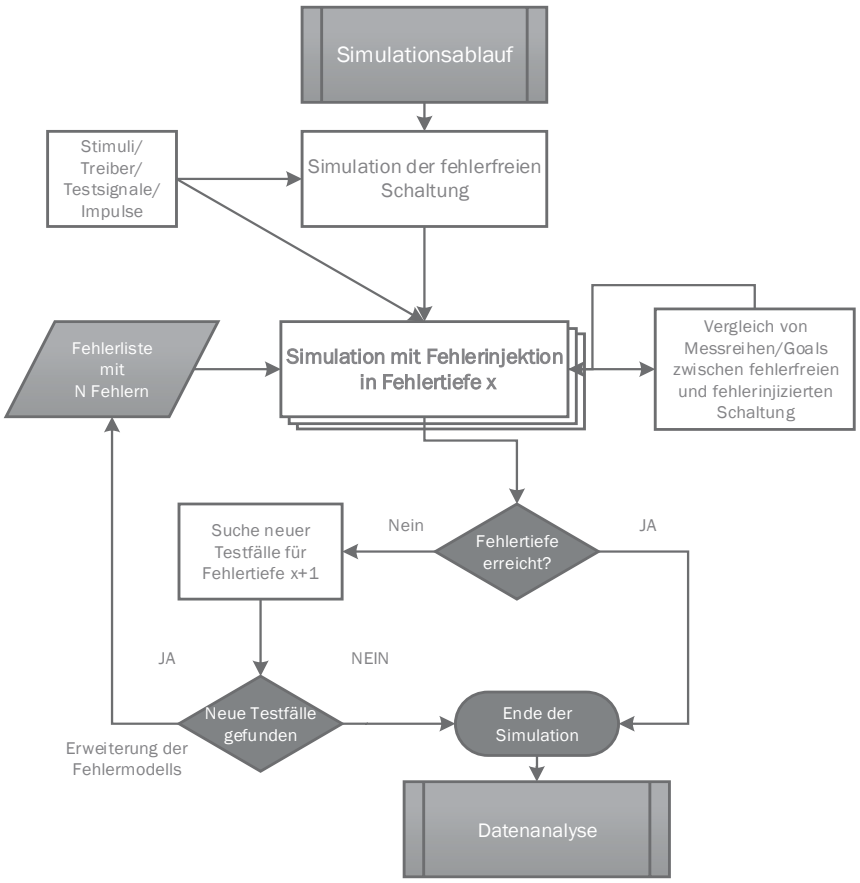


Abbildung 17. Grundsätzliche Struktur der Fehlersimulation für Mehrfachfehlerinjektion

Zur Prüfung der Auswirkungen der injizierten Fehler werden die Analysemethoden der analogen Schaltungssimulationsprogramme verwendet. Typische Analysemethoden sind die DC (Arbeitspunkt), die AC (Frequenzgang), die AC- und DC-Einschwingvorgang-Analyse, die Komponentenparametervariation, die Fourier-Analyse, die Grenzfallanalyse (auch Worst-Case-Analyse genannt). Der Schaltungssimulator berechnet die elektrischen Signale und bestimmt dadurch das Verhalten der untersuchten Schaltung. Der überwiegende Teil der vom Simulator gelieferten Ergebnisse sind zeitabhängige und frequenzabhängige Amplituden (z.B. Ströme und Ausgangspannungen) in Abhängigkeit von einem Eingangsstimulus. Durch die Injektion der einzelnen modellierten Fehlerarten werden die simulierten Signale mit den Signalen der fehlerfreien Schaltung verglichen. Der Fehlersimulator analysiert die Simulationsergebnisse automatisch nach vorher manuell festgelegten Zielen (engl. goals) oder in Form von Toleranzbereichen, die in messpunktbasierte Graphen dargestellt werden. Die Wahl der Fehlerarten und der Komponenten (mit

Fehlerinjektion) erfolgt über die Benutzeroberfläche des Fehlersimulators. Aus dieser Auswahl wird eine Fehlerliste mit Einfehlerannahme generiert, siehe auch die folgende Abbildung 18 der grafischen Benutzeroberfläche. Im Falle der Mehrfachfehlersimulation wird die Fehlerliste sukzessiv während der Fehlersimulation durch Kombination von Fehlern aktualisiert und nicht mehr relevante Fehlerinjektionen (nach Monotonregel, siehe auch Kapitel 3.6.6) werden aussortiert.

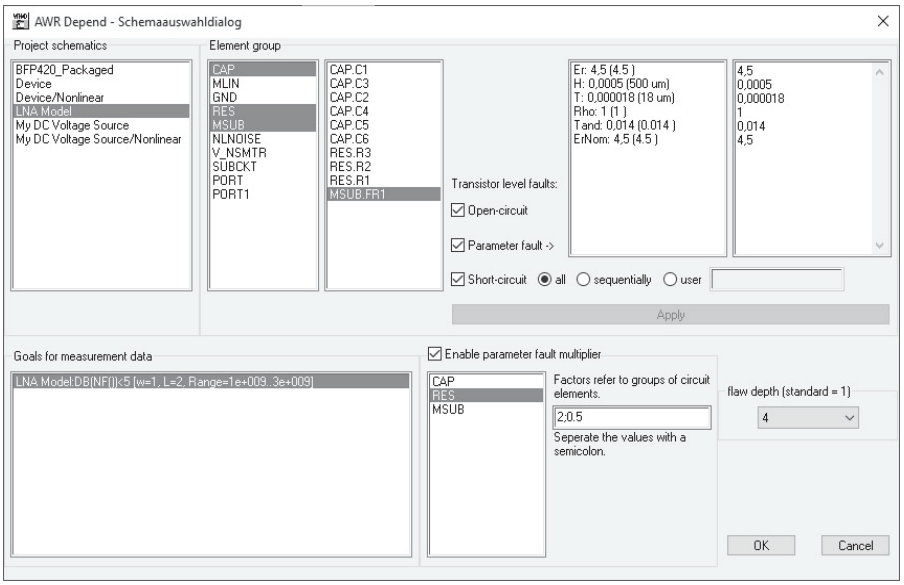


Abbildung 18. Benutzeroberfläche der Fehlersimulation für Mehrfachfehler

4.2.5 DATENAUSWERTUNG UND TOLERANZBEREICH

Die Datenauswertung (insbesondere die Prüfung, ob ein Fehler tolerier wird) wird, wie der Name schon andeutet, nach jeder Simulation einer Fehlerinjektion durchgeführt. In der Auswertungsphase des Fehlersimulationsprozesses werden für jeden injizierten Fehler die zeitabhängigen Verläufe der gemessenen Größen von fehlerhafter und fehlerfreier Schaltung verglichen, um die Entscheidung treffen zu können, ob ein injizierter Fehler zu einer Überschreitung des Toleranzbereichs geführt hat oder nicht. Die gemessenen Größen können in Diagrammen dargestellt werden, für die unterschiedliche Diagrammtypen wählbar sind (Rechteckgraph bzw. x-y-Diagramm, Polar Smith Chart, Histogramme etc.) und über die EDA-Simulationssoftware bereitgestellt werden. In Abbildung 19 ist ein x-y-Diagramm mit einer Messreihe dargestellt. Je nach Schaltung können unterschiedliche Messgrößen auf der Ordinate (z.B. Spannung, Strom, Leistung usw.) sinnvoll sein. Die Messgrößen (Messpunkte) beziehen sich in den meisten Fällen auf Ausgangssignale der Schaltung bei gegebenen Eingangssignalen.

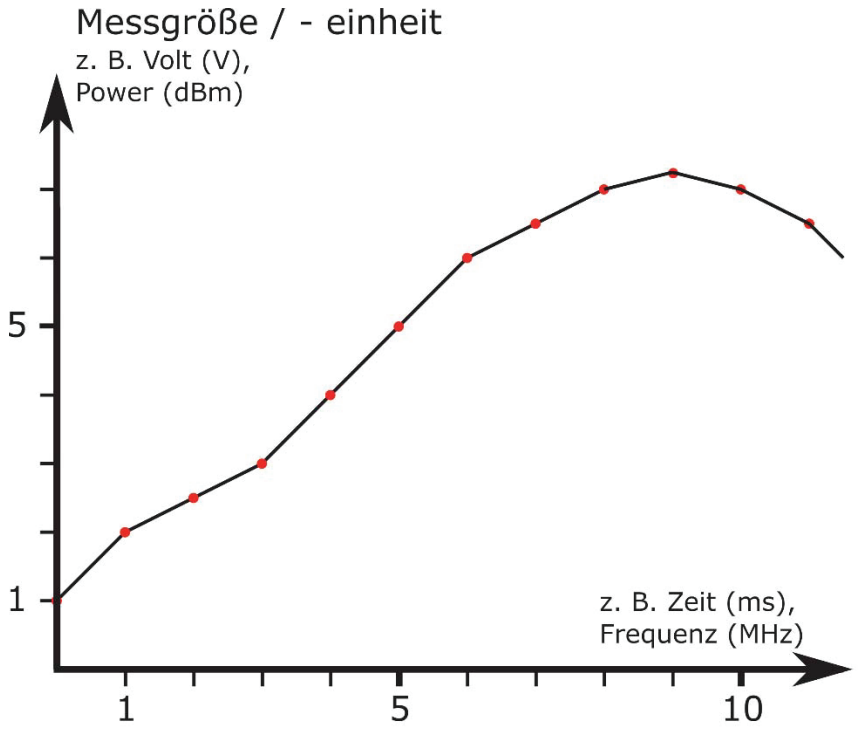


Abbildung 19. Beispiel für ein x-y-Diagramm ohne Toleranzbereich

Ein Soll-Ist-Vergleich basiert auf vorher festgelegte Toleranzgrenzen der Prüfparameter (z.B. frequenzabhängiger Spannungsverlauf auf einem Messpunkt). Innerhalb dieser Toleranzgrenzen wird ein fehlerfreies Verhalten der Schaltung angenommen. Für die Datenauswertung, also der

Analyse nach der Schaltungssimulation, ist die Analyse mittels der Toleranzbereiche eine einfache und aussagekräftige Methode, um die funktionale Korrektheit einer Schaltung zu überprüfen. Die Toleranzbereiche definieren die Grenzen zwischen dem akzeptierten Toleranzbereich (innerer Bereich) und dem nicht akzeptierten Bereich (äußerer Bereich), siehe auch Abbildung 20. Die Festlegung der Grenzen des Toleranzbereiches kann manuell oder durch automatisierte Verfahren erfolgen. In beiden Fällen müssen die jeweils zulässigen Minimal- und Maximalwerte festgelegt werden. Die Toleranzgrenzen können dabei unterschiedlichste Verläufe haben. In der folgenden Abbildung 20 ist ein beispielhafter Signalverlauf mit festgelegtem Toleranzbereich dargestellt.

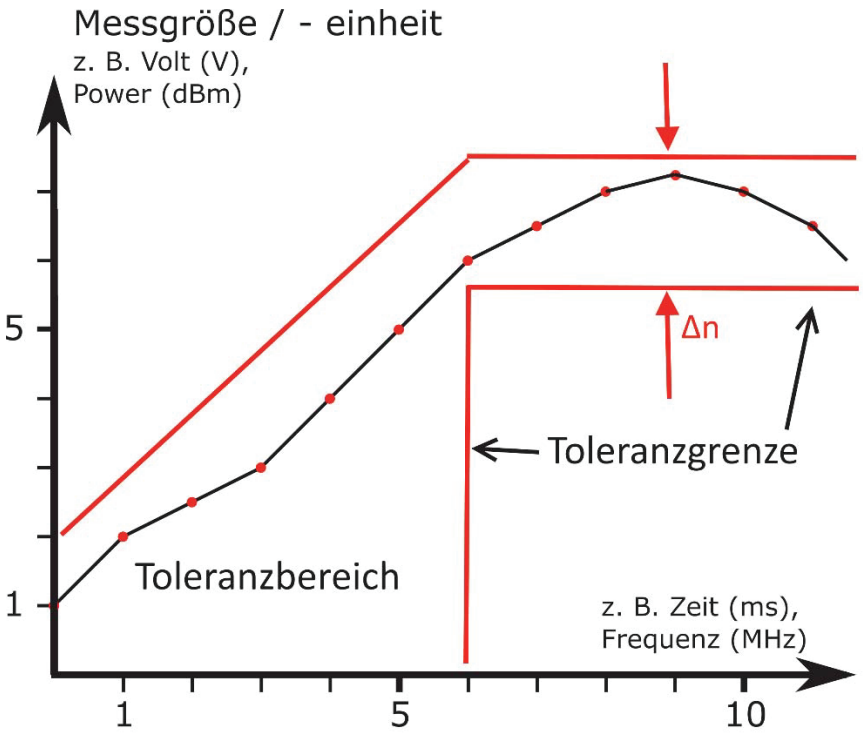


Abbildung 20. Beispiel für einen x-y Graph mit Toleranzbereich und Toleranzgrenze (rot)

Unter Zuhilfenahme der sogenannten Yield-Analyse kann der Toleranzbereich durch die Variationen der Parameterwerte von fehlerfreien Bauelementen (z.B. +/- 5 %) automatisiert bestimmt werden. Für die Yield-Analyse wird eine Vielzahl von Schaltungssimulationen mit leicht veränderten Parameterwerten durchgeführt. Aufgrund von Fertigungstoleranzen und Signalrauschen ist in der Praxis von leicht voneinander abweichenden Signalverläufen auszugehen. Dadurch ergeben sich, wie in Abbildung 21 zu sehen, überlappende Signalverläufe, die durch einen Graphen dargestellt werden. Die Grenzen des Toleranzbereiches werden

durch die Auswahl der beobachteten niedrigsten und größten Messpunkt-Wertepaare festgelegt. Selbst wenn die Variationen bei den jeweiligen Parameterwerten der Bauelemente sehr klein sind, können diese Variationen insgesamt größere Veränderungen in der Schaltung hervorrufen. Beispielsweise können Variationen, die im realen System aufgrund von Temperaturänderungen auftreten, durch eine Yield-Analyse modelliert und für die Simulation verwendet werden.

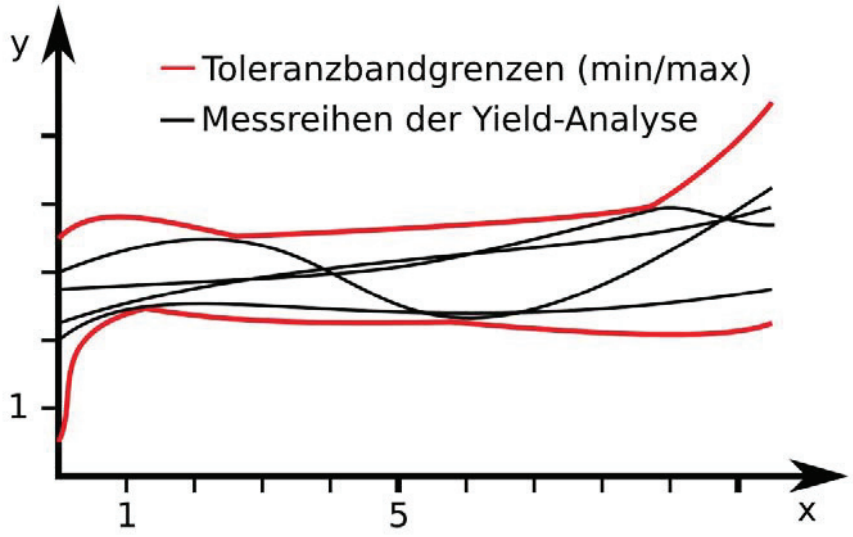


Abbildung 21. Messreihen für die Simulation mit Parametervariationen der Komponenten. Die Achsen haben hier keine festgelegte Dimension.

Außer den eindimensionalen Abhängigkeiten (siehe Diagramme aus Abbildung 19 bis Abbildung 21) können beliebig viele Toleranzbereiche zur Ausfallerkennung und Funktionalitätsprüfung von analogen Schaltungen festgelegt bzw. kombiniert werden. Bei mehreren Ein/Ausgangssignalen würde ein vieldimensionaler Toleranzbereich bzw. eine Menge von zweidimensionalen Toleranzbereichen entstehen. Dazu können die Messpunkt-Daten mittels der API des EDA-Tools ausgelesen werden und für weitere quantitative Analysen genutzt werden. Beispielsweise können Min-Max-Werte, Spitze-Tal-Werte, periodische Schwingungen oder horizontale/vertikale Signalverschiebungen (z.B. zeitliche Verzögerungen) usw. definiert werden, um durch entsprechende Toleranzbereiche das fehlerhafte vom fehlerfreien Verhalten einer Schaltung zu unterscheiden. In der Praxis genügt meist eine Menge von eindimensionalen Abhängigkeiten, dargestellt durch zweidimensionale Diagramme mit der Zeit als Abszisse.

4.2.6 ABLAUF DER FEHLERINJEKTION

Durch die Nutzung des VB-Skriptes zur Fehlermodellierung in Kombination mit der API-Schnittstelle (siehe Abbildung 16) des verwendeten Schaltungssimulators kann eine gezielte Fehlersimulation durchgeführt werden. In der folgenden Abbildung 22 ist die dazu benötigte Objektstruktur abgebildet. Über die entwickelten VB-Skript-Prozeduren ist ein Lesen, Schreiben und Steuern der Schaltungssimulation möglich.

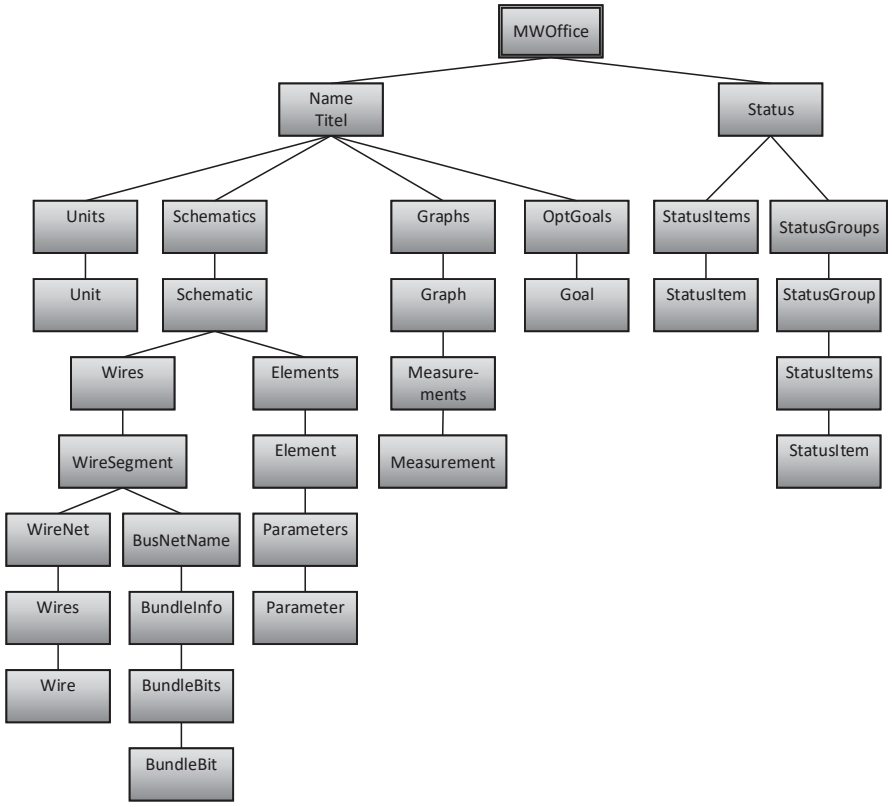


Abbildung 22. Befehlsbaum bzw. Aufrufgraph für die VB-Skript-basierte Fehlersimulation. Vollständige API siehe (NI AWR 2015)

Für die Prüfung der analogen Schaltung, mit Hilfe der Fehlersimulation benötigt man neben der fehlerfreien Schaltung auch eine Fehlerliste mit den modellierten Fehlern. Fehlerarten können erst mit der erstellten Liste systematisch injiziert werden. Ein oder mehrere Fehlerarten einer Schaltungskomponente können durch die folgenden Eigenschaften strukturiert werden. Eine Fehlerart kann einer oder mehreren verschiedenen Komponentenarten zugeordnet werden (z. B. Transistor, Widerstand, Kondensator etc.). Beim FKA werden die Fehlerklassen anhand der Fehlerarten gebildet. Der Ort des Fehlers (d.h. welche konkrete Komponente betroffen ist) ist ebenfalls zu berücksichtigen. Das modifizierte Verhalten der betreffenden Komponenten wird dann entsprechend dem Fehlermodell, das für die Fehlerart definiert ist, realisiert. Aus Tabelle 11 (Kapitel

3.6.4) wird ersichtlich, dass für (feste) Widerstände Unterbrechung (harte Fehlerart) und Parameteränderungen (weiche Fehlerart) die häufigsten Fehler sind. Wenn eine Vereinfachung der Untersuchung angestrebt wird, kann man sich bei der Fehlermodellierung auf die häufigsten Fehlerarten beschränken.

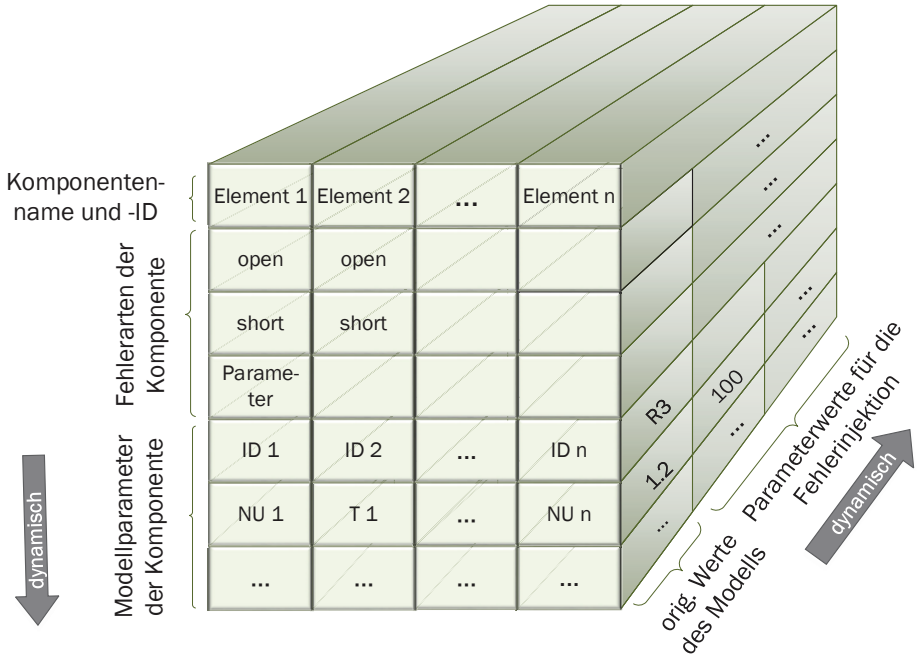


Abbildung 23. Datenmodell des entwickelten Fehlersimulators

Für den in dieser Arbeit entwickelten Fehlersimulator wird zuerst ein Datenmodell der Schaltungskomponenten erstellt (siehe Abbildung 23). Alle für die Fehlersimulation relevanten Informationen wie Komponentenname, Eigenschaften, Parameter (inkl. Datentypen und Parameterwerte) und injizierbare Fehlertypen werden in einer einheitlichen Datenstruktur abgespeichert. Aus dieser Datenstruktur wird anschließend eine Fehlerliste generiert. Die Fehlerliste speichert den Komponentennamen bzw. -ort und die zu injizierende Fehlerart.

Die Erstellung der Fehlerliste kann bei kleinen Schaltungen und bei einer Einfehlerinjektion eventuell noch manuell erfolgen. Bei den meisten praktischen Anwendungen bzw. bei Mehrfachfehlerannahme ist die Komplexität der Schaltung jedoch meist zu hoch, um die Fehlerliste manuell anzugeben. Demzufolge ist auch ein automatisiertes Verfahren für die Erstellung der Fehlerliste (Menge der Fehlerinjektionen) notwendig.

Die Fehlerarten selbst werden mit VB-Skript implementiert. So kann jede erdenkliche Fehlerart definiert werden. Diese Vorgehensweise ist flexibler im Einsatz als bisherige Ansätze der analogen Fehlersimulation (siehe

Kapitel 3.4). Die Fehlerarten werden als VB-Subroutinen definiert und während der Fehlerinjektion ausgeführt. Wie sich der injizierte Fehler lokal bzw. global auswirkt, zeigt die anschließende Schaltungssimulation und der Vergleich der Messwerte mit der fehlerfreien Schaltung.

In Abbildung 24 ist exemplarische eine skriptbasierte Fehlermodellierung gezeigt. Die Fehlerart Kurzschluss wird durch das Hinzufügen von einer weiteren Verbindungsleitung zwischen den Kontaktstellen eines Bauelementes implementiert. Damit handelt es sich um eine sehr einfach zu modellierende Fehlerart Die mit wenig Aufwand für beliebige Anzahl an Kontaktstellen bei Bauelementen verwendet werden kann. Eine weitere und komplexere Fehlermodellierung ist in Anhang B zu finden.

```
Sub FailureType_Short (ElementX As Element, SchemaX As
Schematic)
    Dim x1 As Long  Dim x2 As Long
    Dim y1 As Long  Dim y2 As Long
    n = ElementX.Nodes.Count
    For i = 1 To n Step 2
        x1 = ElementX.Nodes(i).x
        x2 = ElementX.Nodes(i+1).x
        y1 = ElementX.Nodes(i).y
        y2 = ElementX.Nodes(i+1).y
        SchemaX.Wires.Add(x1,y1,x2,y2)
    Next i
End Sub
```

Abbildung 24. Beispielimplementierung (VB-Skript Subroutine) der Fehlerart Kurzschluss bei beliebigen 2 Pin-Komponenten.

Mit der skriptbasierten Fehlermodellierung kann eine Vielzahl von Fehlerarten modelliert werden. Speziell für die analogen Schaltungen soll im Folgenden ein weiteres Fehlermodell exemplarisch präsentiert werden. Das Fehlermodell soll einen Leckstrom bei Kondensatoren modellieren, der erst ab einer bestimmten Spannung bei einem fehlerhaften Isolator auftritt. Bei einem Kondensator kann ein Leckstrom dann auftreten, wenn der Isolator fehlerbedingt eine geringe elektrische Leitfähigkeit besitzt. In Abbildung 25 wird eine Kondensatorschaltung gezeigt und ein Fehlermodell dafür definiert In Abbildung 26 sind die gemessenen Signale in Form der Spannungsverläufe dargestellt. Im fehlerfreien Fall (durchgezogene Linie) steigt die Kondensatorspannung in kurzer Zeit bis auf 5V an und im fehlerhaften Fall (gestrichelte Linie) nähert sich der Spannungsverlauf einer fehlerhaften Spannung unterhalb von 5 V an (ca. 4,6V).

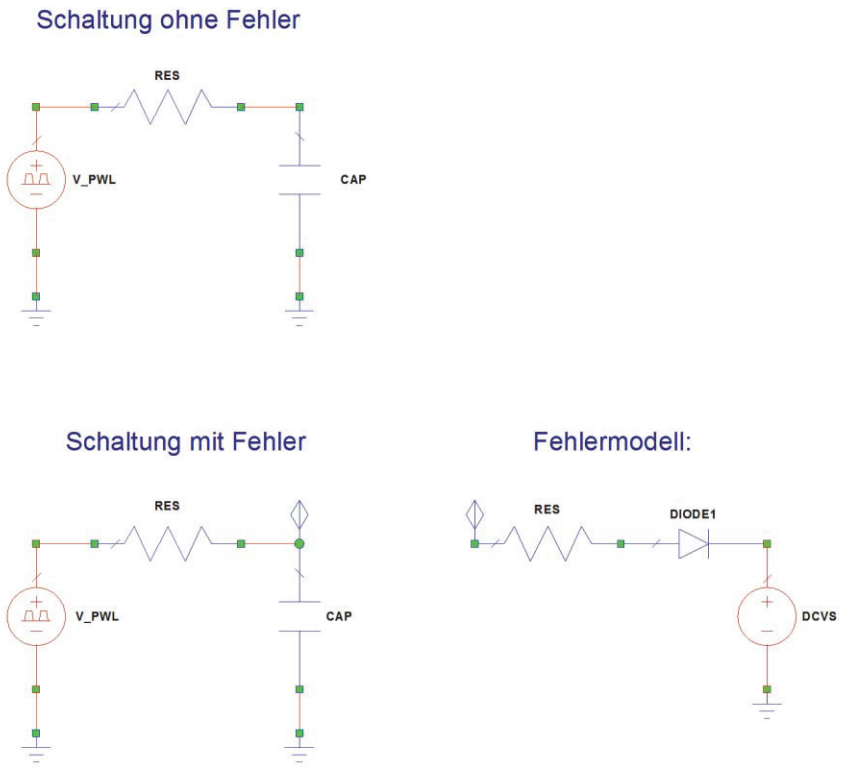


Abbildung 25. Kondensatorschaltung mit Leckstrom-Fehlermodell

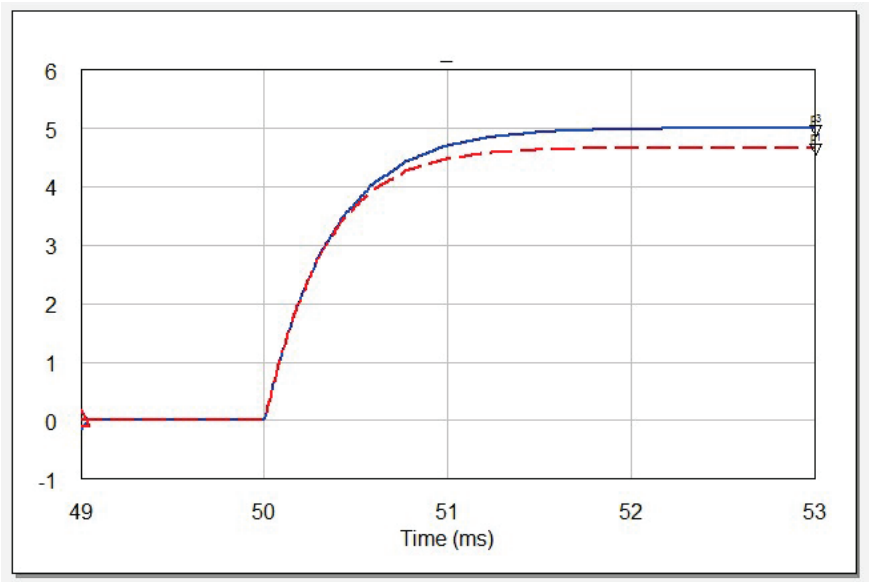


Abbildung 26. Ausgangssignale der Kondensatorschaltung. Spannungskurve der Schaltung ohne Fehler (durchgezogene Linie) und mit Fehlerinjektion (gestrichelte Linie).

Die Fehlermodellierung und -injektion wird durch das VB-Skript realisiert, das in Abbildung 27 dargestellt ist. Die grafische Darstellung des Fehlermodelles und der Fehlerinjektion ist aus Abbildung 25 ersichtlich.

```
Sub FailureType_CapacitorsLeakageCurrent (ElementX As
Element, x As Integer, y As Integer)

'Fehlermodellierung:
'Beschreiben der Fehlerart durch Standartkomponenten

'Bauelemente
Dim res, nc1, nc2, diode, gnd, dcvs

'Res erstellen und Parameter setzen
Set res = Project.Schematics.ActiveSchematic.Ele-
ments.Add("RES", x, y)
res.Parameters(2).ValueAsDouble = 6800

'DIODE1 erstellen
Set diode = Project.Schematics.ActiveSchematic.Ele-
ments.Add("DIODE1", res.Nodes(2).x, res.Nodes(2).y)

'DCVS erstellen und Parameter setzen
Set dcvs = Project.Schematics.ActiveSchematic.Ele-
ments.Add("DCVS", diode.Nodes(2).x, diode.Nodes(2).y)
dcvs.Parameters(2).ValueAsDouble = 2

'GND erstellen
Set gnd = Project.Schematics.ActiveSchematic.Ele-
ments.Add("GND", dcvs.Nodes(2).x, dcvs.Nodes(2).y)



'Fehlerinjektion:
'Verbindung zwischen fehlerhaften Bauelement (Ele-
mentX) und Fehlermodell herstellen.

'nc1 und nc2 erstellen
Set nc1 = Project.Schematics.ActiveSchematic.Ele-
ments.Add("NCONN", res.Nodes(1).x, res.Nodes(1).y)
Set nc2 = Project.Schematics.ActiveSchematic.Ele-
ments.Add("NCONN", ElementX.Nodes(1).x, Ele-
mentX.Nodes(1).y)
nc1.Parameters("Name").ValueAsString = "CONN_X"
nc2.Parameters("Name").ValueAsString = "CONN_X"


End Sub
```

Abbildung 27. Beispielimplementierung (VB-Skript Subroutine) der Fehlerart Leckstrom bei Kondensatoren.

5 BEWERTUNG

In diesem Kapitel soll die Güte des Fehlerklassen-Algorithmus (FKA) im Hinblick auf die effiziente Fehlersimulation erfasst und quantitativ bewertet werden. Für die Bewertung der Qualität des FKA und des Fehlersimulators sind:

- die Leistungsfähigkeit der Simulation in Form von Zeitaufwand bzw. Zeitersparnis und
- die Genauigkeit der Ergebnisse

wesentlich. Der Fokus dieser Arbeit liegt auf analoger Fehlersimulation mit der Analyse von Mehrfachfehlern. Die Eigenschaft der Monotonie wird bei der Simulation von analogen Schaltungen vorausgesetzt bzw. nicht in Frage gestellt. Dies bedeutet, dass bei der Injektion von mehreren Fehlern in unterschiedliche Bauelementen (z.B. ≥ 2 Fehler gleichzeitig) nur noch die Fehlerarten kombiniert werden, die bei der Simulation keinen Ausfall der Schaltung hervorgerufen haben.

Um eine ausreichend große Testmenge zu erhalten, wurden ein Dutzend analoge Schaltungen ausgewählt und auf Transistorebene betrachtet. Der Großteil der Schaltungen sind Beispielschaltungen von EDA-Tools und daher auch über die entsprechenden Literaturverweise frei verfügbar. Die Kriterien für die Auswahl der Schaltungen sind u.a.

- mittelgroße Schaltungen (≥ 10 Bauelemente)
- mit einem Schaltungssimulator simulierbare Schaltungen (Eine Schaltung ist beispielsweise nicht/schlecht simulierbar, wenn der Simulator nicht in „vernünftiger“ Zeit die aus der Schaltung gewonnenen Gleichungssystemen lösen kann)
- bestehend aus Bauelementen (z.B. Transistoren, Widerstände usw.), in welche die modellierten Fehlerarten (Kurzschluss, Unterbrechung, Parameterfehler) injiziert werden können
- die Fehlertiefe sollte mindestens 3 betragen und die Simulation mit vertretbarem Aufwand (CPU-Zeit) durchführbar sein. D.h. die ausgewählte Schaltung darf für bestimmte Dreifachfehler ausfallen. Sie muss aber auch für einige Dreifachfehler noch korrekt funktionieren (andernfalls würde FKA kein Ergebnis liefern).
- die Bewertung der Leistungsfähigkeit der Schaltung sollte über „aussagekräftige“ Messpunkte, für die sich gut Toleranzbereiche definieren lassen, möglich sein.

Es wurden zeitintensive Fehlersimulationen der ausgewählten Schaltungen für die Bewertung der Techniken (Monotonie-Regel und FKA) zur Reduktion der Simulationszeit durchgeführt. Der verwendete Schaltungssimulator wurde hinsichtlich der Simulationszeiten nicht mit anderen Schaltungssimulatoren verglichen, da ein direkter Vergleich wegen der unterschiedlichen Fähigkeiten der EDA-Tools, unterschiedlichen Modelle für Bauelemente und unterschiedlichen Entwurfsziele der Schaltungssimulatoren nur schwer möglich ist. Auch wenn ein Vergleich der Simulationszeit wichtig für die Gesamtprüfzeit mit allen Fehlerinjektionen der Schaltung ist, kann diese auf Grund der unterschiedlichen Fähigkeiten der Schaltungssimulation nicht aussagekräftig verglichen werden. Es besteht aber auch nicht die Notwendigkeit eines direkten Vergleiches, weil der FKA nicht in die einzelnen Simulationsläufe eingreift, sondern nur deren Anzahl reduziert. Daher kann der Nutzen von FKA unabhängig von der Technik der Schaltungssimulation betrachtet werden, da die Simulation selbst (außer durch die Fehlerinjektion) nicht verändert wird. Insofern stellt die festgestellte Anzahl der benötigten Simulationsläufe ein simulatorunabhängiges Maß zur Bewertung dar.

Um die Effizienz von FKA der einzelnen Schaltungen untereinander vergleichbarer zu machen, wurde die Gesamtsimulationszeit nicht in die Bewertung aufgenommen. Die Dauer einer Schaltungssimulation ist abhängig von der Anzahl der Bauelemente, der Analyseart der Schaltungssimulation. Des Weiteren ist die Anzahl der durchgeführten Simulationsläufe der Testschaltungen sehr unterschiedlich. Deshalb findet sich in den Ergebnissen die Anzahl der benötigten bzw. der eingesparten Simulationsläufe für die Bewertung des FKA wieder. Die quantitativen Ergebnisse sind in Kapitel 5.2 zusammengefasst und die Analyse der Gründe für die Wirksamkeit des FKA sind im Kapitel 5.3 erörtert.

5.1 SIMULIERTE SCHALTUNGEN

Alle Simulationen wurden auf einer Workstation (32 GB DDR3-RAM, Intel Xeon E3-1286 v3 (Haswell) 4-Core CPU, SSD SATA3 mit MS Windows 10) durchgeführt. Der in dieser Arbeit entwickelte Fehlersimulator wurde mit verschiedenen analogen Schaltungen untersucht. Für eine eingehende Untersuchung der Anwendbarkeit dieses Ansatzes wurden Schaltungen mit einer unterschiedlichen Anzahl an Bauelementen und von unterschiedlicher Schaltungsart berücksichtigt. Die analogen Schaltungen können in unterschiedliche Arten eingeteilt werden (z.B. Filter, Ein- oder Mehrstufen-Verstärker, Taktgeneratoren, A/D- und D/A-Konverter und andere). Eine Übersicht über die verwendeten Schaltungen ist in der folgenden Tabelle 13 zu finden, wobei jeweils die Bauelement-Anzahlen angegeben sind. Die verwendeten Schaltungen repräsentieren ganz typische analoge Schaltungen mit Bauelementen, die sich für die Fehlerinjektion mit realitätsnahen Fehlermodellen eignen.

	Beispielschaltungen	**Bauelemente insgesamt**	**Widerstände**	**Kondensato-**	**Transistoren**	**Spulen**	**Strom-/ Spannungsquellen**
1	LM741 Verstärker (National Semiconductor 2000)	38	13	1	20	0	3
2	VHF/UHF Verstärker (Gentzler und Leong 2003)	52	10	13	0	10	3
3	Limiter BSP (AWR Corporation 2015)	47	17	4	25	0	1
4	Spannungsstabilisator (Morgenstern 1997)	8	5	0	2	0	1
5	BJT_Amp_Complete: Small Signal (NI examples 2016a)	13	3	3	1	0	1
6	BJT_Amp_Complete: Large Signal (NI examples 2016a)	13	3	3	1	0	2
7	Gilbert_Cell (NI examples 2016b)	50	16	6	10	3	1
8	IAM_81_Gilbert_Cell_Noise (NI examples 2016c)	61	18	6	14	6	1
9	Limiter_HB_Aplac (NI examples 2016d)	68	19	4	25	0	1
10	Low_Power_Mixer (NI examples 2016e)	28	3	6	1	2	1
11	Nonlinear Noise Switch Views (NI examples 2016f)	28	3	6	1	0	1
12	UHF Power Amplifier (NI examples 2016g)	60	10	13	0	18	2
13	Impuls-Signalverstärker	167	67	9	34	0	1

Tabelle 13. Eigenschaften der untersuchten Schaltungen

5.2 ERGEBNISSE DER FEHLERSIMULATION UND DIE BEWERTUNG DES FEHLERKLASSEN-ALGORITHMUS (FKA)

Für jede Schaltung wurden Akzeptanzkriterien in Form von Toleranzbereichen festgelegt. Die verwendeten Toleranzbereiche und weitere Details der jeweiligen Schaltungen sind in Anhang C dargestellt. Die Toleranzbereiche werden in die Graphen eingeblendet, in denen auch die Messgrößen in Form von Signalverläufen dargestellt werden. Die meisten ausgewählten Schaltungen wurden mit wenigen Änderungen aus der Beispielbibliothek des verwendeten EDA-Tools übernommen. Die Ergebnisse der durchgeführten Fehlersimulationen sind Tabelle 15 zu entnehmen. Der Zweck dieser Tabelle ist die Bewertung des FKA in Bezug auf die Anzahl der eingesparten Simulationsläufe und die Genauigkeit der interpolierten Ergebnisse.

Für ein besseres Verständnis der Ergebnistabelle (Tabelle 15) wird zuvor in Tabelle 14 noch ein Bespiel für eine nicht optimierte, eine mit der Monotonie-Regel durchgeführte und eine mit FKA durchgeführten Fehlersimulation angegeben, jeweils unterschieden nach Fehlertiefe. Wie in Spalte (II) der Tabelle 14 zu sehen ist, reduziert die Fehlersimulation mit Anwendung der Monotonie-Regel den Simulationsaufwand bereits erheblich. Der FKA reduziert die Anzahl der Simulationsläufe nochmals für die höchste simulierte Fehlertiefe. In Tabelle 15 ist jeweils nur die Gesamtzahl der Simulationsläufe angegeben.

Simulierte Schaltung mit Fehlerinjektion	Fehlertiefe (Einfach- und Mehrfachfehlersimulationen)	(I) Anzahl der Simulationsläufe für alle möglichen Fehlerkombinationen	(II) Anzahl der Simulationsläufe mit Monotonie-Regel	(III) Anzahl der Simulationsläufe mit FKA
2) VHF/UHF Verstärker (Gentzler und Leong 2003)	Fehlertiefe 1:	164	164	164
	Fehlertiefe 2:	13.120	1.424	1.424
	Fehlertiefe 3:	682.240	16.598	6.045
	Summe:	695.524	18.186	7.633

Tabelle 14. Beispiel für die Anzahl der Simulationsläufe mit Fehlerklassen (FKA) und Monotonie Ansatz für unterschiedliche Fehlertiefen (FT)

Die Bedeutungen der Spalten aus Tabelle 15 sind die folgenden:

(I) Die Anzahl der Gesamtsimulationen, d.h. die Gesamtanzahl der benötigten Einfach- und Mehrfachfehlersimulationen für die unter (VI) angegebenen Fehlertiefen (z.B. Fehlertiefe 1- 3).
(II) Die Anzahl der mit dem Fehlersimulator (Kapitel 4.2) unter Annahme der Monotonie-Regel durchgeführten Fehlerinjektionen.
(III) Die Anzahl der Simulationsdurchläufe unter Anwendung des Fehlerklassen-Algorithmus (Kapitel 4.1). Durch die teilweise Interpolation der letzten angegebenen Fehlertiefe(n) und dazugehörenden Fehlerklasse(n) kann auf einen Teil der Läufe unter Fehlerinjektion verzichtet werden.
(IV) Der Beschleunigungsfaktor (Fehlerklassen vs. Monotonie-Regel) der Fehlersimulation als Quotient aus (II) und (III).
(V) Die relative Abweichung der Anwendung des FKA (III) im Vergleich zu den fehlersimulierten Ergebnissen aus (II). Der Ablauf des Experimentes ist im Kapitel 4.1.1 beschrieben.
(VI) Die untersuchte Fehlertiefe [1-n] und die Anzahl der FKA-Ketten für das FKA-Verfahren der Schaltung.

Zusammenfassung der Ergebnisse

Wie aus den durchschnittlichen Ergebnissen in der letzten Zeile der Tabelle 15 zu erkennen ist, können mit einer durchschnittlichen Abweichung von nur 1,6% zwischen den simulierten Ergebnissen der Spalte (II) und den mittels FKA interpolierten Werten der Spalte (III) insgesamt rund 45 % der Simulationsläufe eingespart werden. Der Median liegt bei einer Abweichung von nur 1%, was einer meist vernachlässigbaren Abweichung entspricht. FKA vermag den Toleranzgrad also sehr gut vorauszusagen. Es sei betont, dass die relative Abweichung stets bezüglich der Fehlertiefe angegeben wird, auf die der FKA angewendet wird. In den Fehlertiefen 1 und 2 erzielt der FKA ohnehin keine Einsparungen.

Simulierte Schaltungen mit Fehlerinjektion	(I) Anzahl der Simulationsläufe für alle möglichen Fehlerkombinationen	(II) Anzahl der Simulationsläufe mit Monotonie-Regel	(III) Anzahl der Simulationsläufe mit FKA	(IV) Beschleunigungsfaktor	(V) Die relative Abweichung von FKA in %	(VI) Fehlertiefe [1-n] und Anzahl FK insgesamt
1) LM741 Verstärker (National Semiconductor 2000)	3020664	2090	1718	1,22	0,5	[1-4] 65
2) VHF/UHF Verstärker (Gentzler und Leong 2003)	695525	18186	7633	2,38	1,0	[1-3] 30
3) Limiter BSP (AWR Corporation 2015)	1045256	1208	858	1,40	0,2	[1-4] 49
4.1) Spannungsstabilisator I (Morgenstern 1997)	8358	4088	2688	1,53	0,2	[1-3] 239
4.2) Spannungsstabilisator II (Morgenstern 1997)	317248	11173	5209	2,14	1,0	[1-4] 1001
5) BJT_Amp_Complete: Small Signal (NI examples 2016a)	72688	28061	12579	2,23	0,9	[1-4] 677
6) BJT_Amp_Complete: Large Signal (NI examples 2016a)	72688	6502	1832	3,55	2,9	[1-4] 188
7) Gilbert Cell (NI examples 2016b)	54810	26475	17300	1,53	0,4	[1-3] 7
8) IAM 81 Gilbert Cell Noise (NI examples 2016c)	2237248	8024	6678	1,20	1,3	[1-3] 72
9) Limiter_HB_Aplac (NI examples 2016d)	803440	13403	9155	1.46	0.4	[1-3] 147
10) Low_Power_Mixer (NI examples 2016e)	64400	7742	4649	1.67	5,4	[1-4] 423
11) Nonlinear Noise Switch Views (NI examples 2016f)	4520	1208	752	1,61	1,3	[1-4] 12
12) UHF VHF Power Amplifier (NI examples 2016g)	102426	12009	8005	1,50	6,0	[1-3] 7
13) Impuls-Signalverstärker	148259	7849	3335	2,35	1,0	[1-3] 30
			Durchschnitt:	1, 84	1,6	

Tabelle 15. Simulationsergebnisse mit Fehlerklassen-Algorithmus (FKA)

5.3 BEGRÜNDUNG DER WIRKSAMKEIT DES FKA

In der vorliegenden Arbeit wurde auf die Mehrfachfehleranalyse fokussiert und eine Methode entwickelt, um den dabei entstehenden Rechenaufwand für die Fehlersimulation zu reduzieren. In diesem Kapitel sollen die Hintergründe des FKA analysiert und die dem FKA zugrundeliegenden Effekte, die zur Einsparung von Simulationsläufen führen, diskutiert werden.

Die Fehlerarten beschreiben, auf welche unterschiedlichen Arten die Komponenten ausfallen können. In praktisch jedem komponentenbasierten System lassen sich dominante Fehlerarten identifizieren. Gemeint sind diejenigen Fehlerarten die eine dominante Rolle beim Ausfall des Systems spielen. Dies bedeutet, dass dominante Fehler einen geringeren Toleranzgrad aufweisen als nicht dominante Fehler. Häufig gibt es nur eine kleine Anzahl von dominanten Fehlerarten im System. Beim FKA werden die Fehlerarten zu Fehlerklassen zusammengefasst, wobei die Dominanz einer Fehlerklasse durch deren Toleranzgrad bestimmt wird.

Die durch den FKA bestimmten FK-Ketten weisen stets einen monoton steigenden oder monoton fallenden Toleranzgrad auf, bis auf wenige Ausnahmen. Dieser monotone Effekt lässt sich dadurch erklären, dass in der Kette eine Fehlerart stetig zunimmt, während eine andere abnimmt. Dominante Einzelfehlerarten sind womöglich auch in der Kombination, die eine Fehlerklasse ausmacht, dominant. Um diesen vermuteten Effekt des FKA zu überprüfen, werden die FK-Ketten aller in Kapitel 5.1 bzw. 5.2 aufgeführten Schaltungen auf die Dominanz der Fehlerarten untersucht. Für die Untersuchung werden die Fehlerarten, die in den mit dem FKA aufgestellten FK-Ketten zu- bzw. abnehmen, überprüft.

- Wenn der Toleranzgrad einer Fehlerklasse X kleiner ist als der Toleranzgrad einer Fehlerklasse Y und auch der Toleranzgrad einer kombinierten Fehlerklasse <X, Z> kleiner ist als der Toleranzgrad einer Fehlerklasse <Y, Z>, dann wird dies als T-Übereinstimmung bezeichnet („T“ weist auf „Toleranzgrad“ hin).
- Andernfalls liegt T-Abweichung vor.

Als Beispiel für eine FK-Kette sei die in Abbildung 28 dargestellte Fehlerkombination genannt. Daraus wird ersichtlich, dass in den gemäß FKA bestimmten FK-Kette eine Fehlerart (D) innerhalb der Kette abnimmt, eine Fehlerart (C) zunimmt und eine Fehlerart (A) (oder auch mehrere) in allen FK der FK-Kette gleich stark vertreten sind.

Die Analyse der Fehlerklassen-Ketten soll anhand der Abbildung 28 durch das folgende Beispiel erläutert werden. Für alle Ketten (insgesamt 908 Ketten) der Schaltungen aus Tabelle 16 wurde die Analyse in gleicher Weise durchgeführt. In der Abbildung 28 ist aus den beiden Fehlerklassen FK_2(A, D) und FK_2(A, C,) die FK-Kette < FK_3(A, D, D), FK_3(A, C, D), FK_3(A, C, C) > gebildet worden. Die FK-Kette hat eine zunehmende Fehlerart (C), eine abnehmende Fehlerart (D) und die gemeinsame Fehlerart (A). Die FK-Kette hat in diesem Beispiel einen abnehmenden Toleranzgrad, d.h. der Toleranzgrad ist bei der FK_3(A, D, D) höher als bei FK_3(A, C, C). Nun sind zwei Fragestellungen von Interesse: Zum einen, ob der Toleranzgrad der Einzelfehlerart (D) ebenfalls höher ist als der Toleranzgrad der Einzelfehlerart (C), und zum anderen, ob der abnehmende Toleranzgrad auch schon zwischen FK_2(A, D) und FK_2(A, C,) zu beobachten ist und sich in die höheren Fehlertiefen fortsetzt.

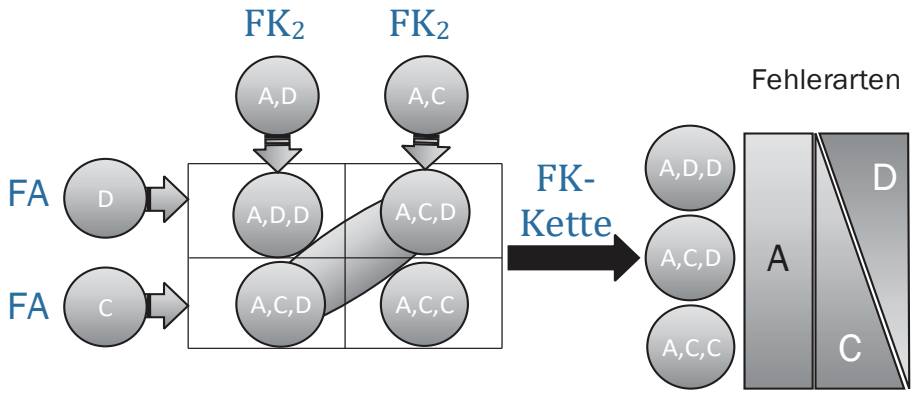


Abbildung 28. Beispiel für eine FK-Kette mit zunehmender (C), abnehmender (D) und gleichbleibender Fehlerart (A).

Die Ergebnisse der Analyse sind in der folgenden Tabelle 16 zu finden.

- Die Spalte (I) ist die Fehlertiefe (1-x) und die Anzahl der interpolierten FK-Ketten der Fehlertiefe (x) zu finden.
- Die Spalte (II) zeigt den Anteil und die Anzahl der T-Übereinstimmungen, wenn Einzelfehler mit den Fehlern der höchsten Fehlerklasse (z.B. Dreifachfehler) verglichen werden (Die Fehlertiefen der verglichenen Klassen sind daher 1 und x).
- Die Spalte (III) zeigt den Anteil und die Anzahl der T-Übereinstimmungen, wenn die Fehlerklassen, aus denen die Ketten gebildet worden sind, mit den Fehlerklassen der Ketten verglichen werden (Die Fehlertiefen der Klassen sind x-1 und x).

Bezogen auf das Beispiel in Abbildung 28 gehören die Fehlerklassen (FK_2(A, D) und FK_2(A, C,)) zu der nächsthöheren Fehlertiefe der FK-Kette < FK_3(A, D, D), FK_3(A, C, D), FK_3(A, C, C) >. Der relative Wert in den Spalten (II) und (III) wird im Verhältnis zu dem Wert in Spalte (I) gebildet.

In der letzten Zeile der Tabelle 16 ist die Summe der FK-Ketten der durchgeführten Untersuchungen in Absolut- und in Prozentwerten zu finden. Zusammenfassend hat sich gezeigt, dass sich die T-Übereinstimmung bei der Gegenüberstellung der Fehlertiefen 1 und x (Spalte II in Tabelle 16) nicht bestätigt hat. Nur bei ca. 47 % der FK-Ketten stimmt der zunehmende bzw. abnehmende Toleranzgrad mit der Zunahme bzw. Abnahme des Toleranzgrads in der untersuchten FK-Ketten überein. Die zweite Untersuchung (Spalte (III) in Tabelle 16) zeigt mit über 80% Übereinstimmung ein sehr deutliches Ergebnis hinsichtlich der T-Übereinstimmung bei Gegenüberstellung der Fehlertiefen x-1 und x. Wird jedes Ergebnis der Spalte (III) einzeln betrachtet, so liegt bei 11 von 13 fehlerinjizierten Schaltungen die Übereinstimmung bei deutlich über 80 % und bei der Hälfte der Schaltungen sind es annähernd100%.

Die T-Übereinstimmung einer Fehlerart setzt sich meist nicht über beliebig viele Mehrfachfehler hinweg. Wenn eine Fehlerart für Einfachfehler dominiert, dann dominiert sie für Dreifachfehler bzw. Vierfachfehler häufig nicht mehr. Jedoch ist die T-Übereinstimmung von einer Fehlertiefe zur nächsthöheren Fehlertiefe häufig sehr hoch, beispielsweise beim Übergang von Doppel- zu Dreifachfehlern und von Dreifach- zu Vierfachfehlern. Dies erklärt, warum die Interpolation des FKA meist sehr gute Ergebnisse liefert, die nahe an den tatsächlichen Werten liegen.

Simulierte Schaltungen	(I) Fehlertiefe [1 bis x] und Anzahl der FK-Ketten (interpolierte Fehlerklassen) in Fehlertiefe x.	(II) Übereinstimmende Richtung des Toleranzgrades der FK-Ketten zur abnehmenden und zunehmenden Fehlerart in % und als Absolutwert	(III) Monotonie des Toleranzgrades der FK-Ketten identisch zur höheren FK (Fehlertiefe (x-1)) in % und als Absolutwert
1) LM741 Verstärker	[1-4] 5	0,00% 0	100,00% 5
2) VHF/UHF Verstärker	[1-3] 10	40,00% 4	50,00% 5
3) Limiter BSP	[1-4] 8	100,00% 8	100,00% 8
4.1) Spannungsstabilisator I	[1-3] 29	41,38% 12	93,10% 27
4.2) Spannungsstabilisator II	[1-4] 407	53,07% 216	71,99% 293
5) BJT_Amp_Complete: Small Signal	[1-4] 209	22,49% 47	80,38% 168
6) BJT_Amp_Complete: Large Signal	[1-4] 64	45,31% 29	93,75% 60
7) Gilbert Cell	[1-3] 1	0,00% 0	100,00% 1
8) IAM 81 Gilbert Cell Noise	[1-3] 7	85,71% 6	0,00% 0
9) Limiter_HB_Aplac	[1-3] 41	48,78% 20	100,00% 41
10) Low_Power_Mixer	[1-4] 114	68,42% 78	99,12% 113
11) Nonlinear Noise Switch Views	[1-4] 2	50,00% 1	100,00% 2
12) UHF VHF Power Amplifier	[1-3] 1	0,00% 0	100,00% 1
13) Impulse-Signalverstärker	[1-3] 10	60,00% 6	80,00% 8
Summe FK-Ketten:	908	427 **47,03%**	732 **80,62%**

Tabelle 16. Vergleich der Analyseergebnisse zur T-Übereinstimmung von Fehlerklassen

6 ZUSAMMENFASSUNG

Die Fehlersimulation analoger Schaltungen mit Hilfe der Schaltungssimulation ist ein wichtiges und zugleich komplexes Vorgehen. In der vorliegenden Arbeit wurden mehrere Methoden der effizienten Fehlersimulation beschrieben. Auch mit den neuen Ansätzen und Fehlermodellierungsmöglichkeiten, die aus der Literatur bekannt sind, bleibt die analoge Fehlersimulation ein herausfordernder Forschungsbereich mit Potenzial für weitere Ansätze zur Reduktion des Rechenaufwandes. Für einige Systeme ist die Fehlersimulation eine sehr gut durchdachte und weit entwickelte Methode (z.B. Software-Systeme), für andere Systeme (z.B. hoch integrierte analoge Schaltungen) wurden bisher nur einfache technische Verfahren für die Fehlerinjektion und -modellierung publiziert. Dabei hat sich jedoch nach wie vor kein Standard-Fehlermodell herauskristallisiert. Komponentenspezifisch sind recht unterschiedliche Fehlermodelle mit unterschiedlichem Detaillierungsgrad bekannt, was einen Vergleich zwischen den einzelnen Ansätzen schwierig macht. Die vorliegende Arbeit hat sich das Ziel gesetzt, den Aufwand für die Fehlersimulation bei der Berücksichtigung von Mehrfachfehlern zu reduzieren. Eine nicht optimierte analoge Fehlersimulation kann kaum eine akzeptable Prüfzeit (CPU-Zeit) erreichen. Die Prüfung analoger Schaltungen mit Mehrfachfehlersimulation und mehreren Fehlerarten pro Komponente wurde untersucht und daraus eine neue Fehlermodellierungsmethode entwickelt und erfolgreich angewendet. Die skriptbasierte Fehlermodellierung bietet eine sehr viel bessere Flexibilität im Sinne der denkbaren Fehlerarten. Diese unterstützt die Zuverlässigkeitsanalyse in frühen Phasen der Schaltungsentwicklung, noch bevor ein Prototyp der Schaltung verfügbar ist.

Die in dieser Arbeit entwickelte Methode (FKA) identifiziert Fehlerklassen bei Mehrfachfehlerinjektion, die zu FK-Ketten vereinigt werden, so dass der Toleranzgrad zum Teil ohne Simulation mit sehr guter Näherung vorausgesagt werden können.

Mehrfachfehlersimulation mittels FKA wurde anhand einer Auswahl analoger Schaltungen überprüft. Dabei wurde festgestellt, dass sie die Simulation von Mehrfachfehlern signifikant beschleunigt. Die erzielte Beschleunigung und die Genauigkeit des FKA ist von der untersuchten Schaltung abhängig. Die vorgestellte Methode wurde bei rechenintensiven Untersuchungen durch die Fehlersimulation mit der Monotonie-Re-

gel verknüpft und kann bei Bedarf auch mit weiteren Methoden zur Beschleunigung der analogen Fehlersimulation kombiniert werden, da in die Ausführung der einzelnen Simulationsläufe nicht eingegriffen wird. Die Möglichkeit der effizienten Fehlersimulation (mit Mehrfachfehlern) und akzeptabler CPU-Laufzeit verbessert die Qualität der Prüfmethoden von analogen Schaltungen. Die Methode der effizienten Fehlersimulation mit dem FKA und seine Analyse stellt den wissenschaftlichen Hauptbeitrag der vorliegenden Arbeit dar.

7 AUSBLICK

Es sind noch weitere Varianten des FKA denkbar. Eine mögliche Variante könnte die Fehlerarten nach den Komponentenarten unterscheiden, auf die die Fehler anwendbar sind. Der Kurzschluss in einem Transistor wäre dann beispielsweise eine andere Fehlerart als der Kurzschluss in einem Kondensator. Dies würde in den meisten Fällen zu kleineren Fehlerklassen führen, da die Anzahl der berücksichtigten Komponenten geringer wäre. Die FK-Ketten werden dann aus diesen verfeinerten Fehlerarten gebildet. Dies würde die Gesamtzahl der FK mit insgesamt kleineren FK erhöhen. Ob sich dadurch die Effizienz des FKA erhöht oder die Abweichung der Genauigkeit des FKA verringert, lässt sich nicht allgemein aus den vorliegenden Ergebnissen ableiten. Durch sehr kleine FK könnte sich der Genauigkeit durch die Anwendung des FKA eher verschlechtern. Bei einer höheren Anzahl von FK ist der Beschleunigungsfaktor nicht höher als bei einer kleineren Anzahl von FK (siehe auch Tabelle 15), weil stets das mittlere Glied einer Kette aus der Simulation herausgenommen und durch Interpolation behandelt wird. Ein Vorteil der Trennung von FK nach Komponentenarten liegt in der Tatsache, dass dann die Fehler innerhalb einer Fehlerklasse näherungsweise gleich wahrscheinlich sind. Dies ermöglicht bessere Aussagemöglichkeiten über das Zuverlässigkeitsverhalten.

Der FKA kann grundsätzlich auch längere als Dreierketten bilden. Es lassen sich ebenfalls Vierer- oder auch längeren Ketten finden. Zum einen werden dadurch mehr FK interpoliert, wodurch der Aufwand für die Fehlersimulation reduziert wird. Zum anderen ergeben sich durch längere Ketten auch mehr Überschneidungen mit anderen gefundenen FK-Ketten. Durch eine längere Kette und eine dadurch erhöhte Anzahl von Interpolationen könnte auch die Ungenauigkeit der erhaltenen Ergebnisse zunehmen. Ob es insgesamt ein Vorteil ist, möglichst lange FK-Ketten zu bilden, hängt sicherlich von der untersuchten Schaltung ab.

Die Fehlerklassenbildung kann ggf. nicht nur auf die Komponentenarten unterscheiden, sondern auch den Ort eines Fehlers in der Schaltung (z.B. Widerstand hängt an der Versorgungsleitung, dient der Spannungsstabilisierung oder ist für die Steuerung der Betriebspunkte notwendig usw.) berücksichtigen. Dies ist eine wichtige Information für die Prüfung der Schaltung und kann unterschiedliche Folgen für den lokalen bzw. globalen Ausfall der Schaltung haben. Darum könnte die Unterscheidung des Fehlerortes auch für die Anwendung des FKA relevant sein. Dies würde

wie auch bei anderen Differenzierungen zu einer größeren Anzahl von FK-Ketten führen.

Bei der Optimierung des entwickelten Fehlersimulators sind punktuelle Verbesserungen denkbar. Zu den möglichen Optimierungen des Software-Tools gehört die nebenläufige Berechnung der Fehlerkombinationen, die zu injizieren sind. So könnte die Steuerung der Simulation und die Suche nach der Kombination von weiteren Mehrfachfehlern auch auf separaten CPU-Kernen ausgeführt werden. Dadurch können die Simulationspausen zwischen den Fehlertiefen bei der Fehlersimulation reduziert bzw. ganz vermieden werden. Des Weiteren könnte die Simulation von verschiedenen Fehlerfällen parallel auf mehreren Rechnern bzw. durch ein Rechennetz ausgeführt werden, weil die einzelnen Fehlerinjektionen unabhängig voneinander sind.

8 LITERATURVERZEICHNIS

Agrawal, V. D. (1990): Fault sampling revisited. In: *IEEE Des. Test. Comput.* 7 (4), S. 32–35. DOI: 10.1109/54.57911.

Avizienis, A.; Laprie, J.-C.; Randell, B.; Landwehr, C. (2004a): Basic concepts and taxonomy of dependable and secure computing. In: *IEEE Trans.Dependable and Secure Comput.* 1 (1), S. 11–33. DOI: 10.1109/TDSC.2004.2.

Avizienis, Algirdas; Laprie, Jean-Claude; Randell, Brian (2001): Fundamental concepts of dependability: University of Newcastle upon Tyne, Computing Science (Technical report series, no CS-TR-739).

Avizienis, Algirdas; Laprie, Jean-Claude; Randell, Brian (2004b): Dependability and Its Threats: A Taxonomy. In: Renè Jacquart (Hg.): Building the Information Society, Bd. 156. Boston, MA: Springer US (IFIP International Federation for Information Processing), S. 91–120.

AWR Corporation (2015): Bipolar Limiting Amplifier Circuit. Hg. v. AWR Corporation. Online verfügbar unter https://awrcorp.com/download/faq/english/docs/Getting_Started/Tonal_Analysis.html, zuletzt geprüft am 05.03.2015.

Bell, Ian (2007): ANTICS manual PDF version. University of Hull VLSI Design and Test Group. Online verfügbar unter http://www.hull.ac.uk/edtm/antics/antman.pdf, zuletzt aktualisiert am 02.10.2007, zuletzt geprüft am 15.12.2015.

Birolini, Alessandro (2010): Reliability engineering. Theory and practice. 6th ed. Heidelberg, New York: Springer. Online verfügbar unter http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10412202.

Birolini, Alessandro (2014): Reliability engineering. Theory and practice. 7th ed. Heidelberg, New York: Springer. Online verfügbar unter http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10412202.

Burns, Mark; Roberts, Gordon W. (2001): An introduction to mixed-signal IC test and measurement. New York, NY: Oxford University Press (The Oxford series in electrical and computer engineering). Online verfügbar unter http://www.loc.gov/catdir/enhancements/fy0610/00042770-d.html.

Crama, Yves; Hammer, P. L. (2011): Boolean functions. Theory, algorithms, and applications. Cambridge, New York: Cambridge University

Press (Encyclopedia of mathematics and its applications, 142). Online verfügbar unter http://site.ebrary.com/lib/academiccompletetitles/home.action.

Dhifi, Mustapha (2003): Novel approaches for accelerated analog fault simulation. Univ., Diss.--Hannover, 2002. Aachen: Shaker (Berichte aus der Elektronik).

El-Gamal, Mohamed A. (1990): Fault location and parameter identification in analog circuits. Ohio University.

Gentzler, C. G.; Leong, S. K. (2003): Broadband VHF/UHF amplifier design using coaxial transformers. In: *High Frequency Electronics*, S. 42–51. Online verfügbar unter http://www.polyfet.com/HFE0503_Leong.pdf, zuletzt geprüft am 01.03.2015.

Harvey, R.J.A.; Richardson, A.M.D.; Bruls, E.M.J.G.; Baker, K. (1994): Analogue fault simulation based on layout dependent fault models. In: International Test Conference. Washington, DC, USA, 2-6 Oct. 1994, S. 641–649.

Heidtmann, Klaus (1997): Zuverlässigkeitsbewertung technischer Systeme. Modelle für Zuverlässigkeitsstrukturen und ihre analytische Auswertung. Wiesbaden: Vieweg+Teubner Verlag (TEUBNER-TEXTE zur Informatik, 21). Online verfügbar unter http://dx.doi.org/10.1007/978-3-322-95379-7.

Hewlett Packard (1999): AB-0013: SPICE Models for the IAM- 81 and IAM-82 Active Mixers. Online verfügbar unter http://www.hp.woodshot.com/hprfhelp/5_downld/lit/ic-lit/ab0013.pdf.

Hoffmann, C. (2002): A new design flow and testability measure for the generation of a structural test and BIST for analogue and mixed-signal circuits, 2002.

HSPICE (2010): HSPICE The Gold Standard for Accurate Circuit Simulation. Online verfügbar unter http://www.synopsys.com/Tools/Verification/AMSVerification/CircuitSimulation/HSPICE/Documents/hspice_ds.pdf, zuletzt geprüft am 30.11.2015.

IEEE Xplore (2015). Online verfügbar unter http://ieeexplore.ieee.org/search/searchresult.jsp?queryText=analog%20fault%20simulation&newsearch=true, zuletzt geprüft am 30.11.2015.

IEC 50 (191), 1995: Internationales elektrotechnisches Wörterbuch.

James Gleick (1996): A Bug and a Crash. Online verfügbar unter http://www.around.com/ariane.html, zuletzt aktualisiert am 27.01.2003, zuletzt geprüft am 21.10.2015.

Junwei Hou (1998): CONCERT: a concurrent transient fault simulator for nonlinear analog circuits. New York, NY: Association for Computing Machinery.

Junwei Hou; Chatterjee, A. (2003): Concurrent transient fault simulation for analog circuits. In: *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* 22 (10), S. 1385–1398. DOI: 10.1109/TCAD.2003.818129.

Kabisatpathy, Prithviraj; Barua, Alok; Sinha, Satyabroto (2005): Fault Diagnosis of Analog Integrated Circuits. Boston, MA: Springer (Frontiers in Electronic Testing, 30). Online verfügbar unter http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10139739.

Knight, John (2012): Fundamentals of dependable computing for software engineers. Boca Raton: CRC Press. Online verfügbar unter QA76.9.F38 K55 2012.

KNKT (2015): Aircraft Accident Investigation Report (Flight PK-AXC), Final Version. Komite Nasional Keselamatan Transportasi. Hg. v. Indonesia. Ministry of Transportation. Online verfügbar unter http://kemhubri.dephub.go.id/knkt/ntsc_aviation/baru/Final%20Report%20PK-AXC.pdf, zuletzt geprüft am 01.12.2015.

Kooli, Maha; Di Natale, Giorgio (2014): A survey on simulation-based fault injection tools for complex systems, S. 1–6. DOI: 10.1109/DTIS.2014.6850649.

Lazzaroni, Massimo; Cristaldi, Loredana; Peretto, Lorenzo; Rinaldi, Paola; Catelani, Marcantonio (2011): Reliability Engineering. Basic Concepts and Applications in ICT. Berlin, Heidelberg: Springer Berlin Heidelberg. Online verfügbar unter http://dx.doi.org/10.1007/978-3-642-20983-3.

DIN EN 62741:2015-10: Leitfaden zur Darlegung von Zuverlässigkeitsanforderungen - Der Zuverlässigkeitsnachweis (IEC 62741:2015). Online verfügbar unter http://www.beuth.de/en/standard/din-en-62741/237771319.

Leveson, N. G.; Turner, C. S. (1993): An investigation of the Therac-25 accidents. In: *Computer* 26 (7), S. 18–41. DOI: 10.1109/MC.1993.274940.

Leveson, Nancy (1995): Medical Devices: The Therac-25. Online verfügbar unter https://www.cs.ucf.edu/~dcm/Teaching/COP4600-

Fall2010/Literature/Therac25-Leveson.pdf, zuletzt geprüft am 12.02.2017.

Lions, Jacques-Louis (1996): ARIANE 5: Flight 501 Failure. Online verfügbar unter http://www-users.math.umn.edu/~arnold/disasters/ariane5rep.html und http://www.cs.ucsb.edu/~cappello/courses/cs10/lectures/ariane5.html.

Lipinski, H. (1990): FAUSTUS-Fault Simulation Tool Using Spice. In: *Institut für Theoretische Elektrotechnik, Universität Hannover*.

Liu, Ruey-Wen (1991): Testing and diagnosis of analog circuits and systems. New York, N.Y.: Van Nostrand Reinhold.

Machado, Gerson A. S. (1996): Low-power HF microelectronics. A unified approach. London, UK: Institution of Electrical Engineers (IEE circuits and systems series, 8).

Mei-Chen Hsueh; Tsai, T. K.; Iyer, R. K. (1997): Fault injection techniques and tools. In: *Computer* 30 (4), S. 75–82. DOI: 10.1109/2.585157.

Mentor Graphics (2016): Tessent DefectSim. Online verfügbar unter http://s3.mentor.com/public_documents/datasheet/products/silicon-yield/tessent-defectsim-ds.pdf.

Microwave Office Element Catalog (2016): Gummel-Poon NPN BJT: GBJT. Online verfügbar unter https://awrcorp.com/download/faq/english/docs/Elements/GBJT.htm, zuletzt aktualisiert am 11.03.2016, zuletzt geprüft am 09.06.2016.

MIL-HDBK-338B. ELECTRONIC RELIABILITY DESIGN HANDBOOK (01 OCT 1998) (1998). Online verfügbar unter http://everyspec.com/MIL-HDBK/MIL-HDBK-0300-0499/MIL-HDBK-338B_15041/, zuletzt geprüft am 09.05.2016.

Morgenstern, Bodo (1997): Elektronik-Aufgaben Analoge Schaltungen. Wiesbaden: Vieweg+Teubner Verlag (Aus dem Programm Grundstudium). Online verfügbar unter http://dx.doi.org/10.1007/978-3-322-89568-4.

Multisim (2012): NI Multisim: Powerful Circuit Design and Teaching Software - National Instruments. Online verfügbar unter http://www.ni.com/multisim/, zuletzt aktualisiert am 03.02.2012, zuletzt geprüft am 30.11.2015.

Nagi, Naveena; Chatterjee, Abhijit; Abraham, Jacob A.: DRAFTS: discretized analog circuit fault simulator. In: Alfred E. Dunlop (Hg.): the 30th international. Dallas, Texas, United States, S. 509–514.

National Instruments (2015): AWRDE Scripting Landing Page. Online verfügbar unter https://awrcorp.com/download/faq/english/scripts/AWRDE_Scripting_Wiki_Start_Page.aspx, zuletzt geprüft am 25.11.2015.

National Semiconductor (2000): LM741 Operational Amplifier. Online verfügbar unter http://web.mit.edu/6.301/www/LM741.pdf, zuletzt geprüft am 05.03.2015.

NI AWR (2015): APLAC Circuit Simulation Technology. Online verfügbar unter https://encrypted.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0ahUKEwjh2OfO__HTAhVSY1AKHeiAASMQFggiMAA&url=http%3A%2F%2Fwww.awrcorp.com%2Fsites%2Fdefault%2Ffiles%2Fcontent%2Fattachments%2FDS-APLAC-2015.2.10.pdf&usg=AFQjCNEYQgVvs3qjHZ2geWSUgwhjNjxsQ&sig2=YNSDLp9s0dQVKVcNTpJaJw, zuletzt geprüft am 01.02.2017.

NI AWR Design Environment (2016). Online verfügbar unter http://www.awrcorp.com/de/products/microwave-office, zuletzt geprüft am 01.02.2016.

NI examples (2016a): BJT Amp. Online verfügbar unter https://awrcorp.com/download/faq/english/examples/BJT_Amp_Complete.aspx, zuletzt geprüft am 16.08.2016.

NI examples (2016b): Gilbert Cell. Online verfügbar unter https://awrcorp.com/download/faq/english/examples/Gilbert_Cell.aspx, zuletzt geprüft am 16.08.2016.

NI examples (2016c): IAM 81. Online verfügbar unter https://awrcorp.com/download/faq/english/examples/IAM_81_Gilbert_Cell_Noise.aspx, zuletzt geprüft am 17.08.2016.

NI examples (2016d): Limiter HB Aplac. Online verfügbar unter https://awrcorp.com/download/faq/english/examples/Limiter_HB_Aplac.aspx, zuletzt geprüft am 17.08.2016.

NI examples (2016e): Low_Power_Mixer. Online verfügbar unter https://awrcorp.com/download/faq/english/examples/Low_Power_Mixer.aspx, zuletzt geprüft am 17.08.2016.

NI examples (2016f): Nonlinear_Noise_Switch_Views. Online verfügbar unter https://awrcorp.com/download/faq/english/examples/Nonlinear_Noise_Switch_Views.aspx, zuletzt geprüft am 17.08.2016.

NI examples (2016g): UHF_VHF_Power_Amplifier. Online verfügbar unter https://awrcorp.com/download/faq/english/examples/UHF_VHF_Power_Amplifier.aspx, zuletzt geprüft am 17.08.2016.

Olbrich, T.; Perez, J.; Grout, I. A.; Richardson, A.M.D.; Ferrer, C. (1996): Defect-oriented vs schematic-level based fault simulation for mixed-signal ICs. In: International Test Conference 1996. Test and Design Validity. Washington, DC, USA, 20-25 Oct. 1996, S. 511–520.

Pham, Hoang (Hg.) (2003): Handbook of Reliability Engineering. London: Springer-Verlag London Limited. Online verfügbar unter http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10130027.

Piscitelli, Roberta; Bhasin, Shivam; Regazzoni, Francesco (2015): Fault attacks, injection techniques and tools for simulation, S. 1–6. DOI: 10.1109/DTIS.2015.7127352.

Poeh, Frank; Demmerle, Frank; Alt, Juergen; Obermeir, Hermann: Production test challenges for highly integrated mobile phone SOCs — A case study. In: 2010 15th IEEE European Test Symposium (ETS). Praha, Czech Republic, S. 17–22.

Rakowsky, Uwe Kay; Richardson, Nicole (2001): Wörterbuch der Zuverlässigkeit. Hagen/Westfalen, [Norderstedt]: Life-Long-Learning; Libri.

Rausand, Marvin; Høyland, Arnljot (2004): System reliability theory. Models, statistical methods, and applications. 2nd ed. Hoboken, NJ: Wiley-Interscience (Wiley series in probability and statistics. Applied probability and statistics).

Redmill, Felix; Dale, Chris (1997): Life cycle management for dependability. London, New York: Springer.

Roberts, Gordon; Taenzler, Friedrich; Burns, Mark (2012): An introduction to mixed-signal IC test and measurement. 2. ed. New York: Oxford Univ. Press.

Rodriguez-Montanes, R.; Bruls, E. M. J. G.; Figueras, J. (1992): Bridging defects resistance in the metal layer of a CMOS process. In: *J Electron Test* 8 (1), S. 35–46. DOI: 10.1007/BF00136074.

Ross, Sheldon M. (2010): Introduction to probability models. 10th ed. Amsterdam, Boston: Academic Press.

Schwarz, Andreas (2015): Schaltungssimulation. Mikrocontroller.net. Online verfügbar unter http://www.mikrocontroller.net/articles/Schaltungssimulation, zuletzt geprüft am 18.11.2015.

Sebeke, C.; Teixeira, J. P.; Ohletz, M. J. (1995): Automatic fault extraction and simulation of layout realistic faults for integrated analogue circuits. In: the European Design and Test Conference. ED&TC 1995. Paris, France, 6-9 March 1995, S. 464–468.

Spectre (2013): Cadence Spectre Circuit Simulator . Online verfügbar unter http://www.cadence.com/rl/Resources/datasheets/virtuoso_mmsim.pdf#page=4, zuletzt geprüft am 30.11.2015.

Spinks, S. (1997): ANTICS analogue fault simulation software. In: IEE Colloquium on Testing Mixed Signal Circuits and Systems. London, UK, 23 Oct. 1997, S. 13.

Straube, Bernd; Müller, Bert; Vermeiren, Wolfgang; Hoffmann, Christoph; Sattler, Sebastian (2000): Analogue Fault Simulation by aFSIM.

Straube, Bernd; Vermeiren, Wolfgang; Spenke, Volker (2002): Multi-level hierarchical analogue fault simulation. In: *Microelectronics Journal* 33 (10), S. 815–821. DOI: 10.1016/S0026-2692(02)00099-X.

Sunter, Stephen (2016): ANALOG FAULT SIMULATION CHALLENGES AND SOLUTIONS. Mentor Graphics. Online verfügbar unter http://s3.mentor.com/public_documents/whitepaper/resources/mentorpaper_98144.pdf.

Sunter, Stephen; Jurga, Krzysztof; Majidi, Rabeeh; Laidler, Andrew (2015): Practical Analog Fault Simulation. DOI: 10.1109/TEST.2014.7035281.

Variyam, P. N.; Chatterjee, A. (1997): FLYER: fast fault simulation of linear analog circuits using polynomial waveform and perturbed state representation. In: Tenth International Conference on VLSI Design. Hyderabad, India, S. 408–412.

Wunderlich, Hans-Joachim (Hg.) (2010): Models in Hardware Testing. Lecture Notes of the Forum in Honor of Christian Landrault. Dordrecht: Springer Science+Business Media B.V (Frontiers in Electronic Testing, 43). Online verfügbar unter http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10351641.

Ziade, Haissam; Ayoubi, Rafic A.; Velazco, Raoul (2004): A Survey on Fault Injection Techniques. In: *Int. Arab J. Inf. Technol.* 1 (2), S. 171–186.

DIN EN 60300-1: 2015-01: Zuverlässigkeitsmanagement - Teil 1: Leitfaden für Management und Anwendung (IEC 60300-1:2014). Online verfügbar unter http://www.beuth.de/de/norm/din-en-60300-1/224807213?SearchID=955696827, zuletzt geprüft am 07.10.2015.

ANHANG

ANHANG A: GRUNDLAGEN ZUR ZUVERLÄSSIGKEITSTHEORIE

Zur Zuverlässigkeitstheorie bzw. zu Modellen der Systemzuverlässigkeitstheorie existieren sehr gute Bücher und Veröffentlichungen. In diesem Abschnitt soll eine Einführung in die wichtigsten Bestandteile und Ergebnisse der Zuverlässigkeitstheorie gegeben werden. Umfangreichere Einführungen sind bei (Rausand und Høyland 2004; Ross 2010; Lazzaroni et al. 2011) zu finden. Diese primären Quellen wurden verwendet, um diesen Anhang zu erstellen. Um eine einfache Einführung zu geben, müssen mehrere Begriffe genannt und erläutert werden. Dazu gehören das Experiment, der Stichprobenraum (engl. sample space, S), das Ereignis (engl. event, E) bzw. Vorkommnis und das Ergebnis (e) bzw. der Ausgang eines Experiments.

Zuverlässigkeitsbewertungen auf System-, Baugruppen- oder Komponentenebene sind für die Analyse von Fehlern und/oder Leistungsengpässen in möglichst frühen Entwicklungsphasen nicht nur aus sicherheitstechnischen und finanziellen Gesichtspunkten relevant. In diesen Abschnitt werden Methoden für das Bestimmen der Fehlerraten bis hin zur Systemebene und die Analyse der komponentenbasierten Fehlerarten vorgestellt. Zur Untersuchung eines komplexen Systems wird anhand der kalkulierten Fehlerraten der einzelnen verwendeten Bauelemente/Komponenten und der verbundenen Struktur der Komponenten (die eine Systemstruktur beschreiben) untersucht. Bei der Systemstruktur muss zwischen der physikalischen und der zuverlässigkeitsbasierten Struktur unterschieden werden. So kann die physikalische Anordnung von Komponenten (z. B. eine elektronische Schaltung aus Bauelementen) auch wenn es eindeutig redundante Komponenten sind, nicht unbedingt der logischen Struktur im Zuverlässigkeitsblockdiagram entsprechen. Wegen unterschiedlicher Annahmen und Unsicherheiten in der Bewertung der Zuverlässigkeit kann die (anhand eines aufgestellten Modells) tatsächliche Zuverlässigkeit nur mit einer bedingten Genauigkeit angegeben werden. Es existieren drei primäre Ansätze bzw. Zweige der Zuverlässigkeitsanalyse: die Analyse der Hardware, der Software und der menschlichen Zuverlässigkeit. Heutige technische Systeme sind zum Großteil eine Verknüpfung aus Hardware, Software und Menschen (z.B. Entwickler und Bediener eines Systems). In dieser Arbeit wird auf den

ersten der drei Zweige, also die Zuverlässigkeit von technischen (Hardware-) Komponenten und Systemen fokussiert. Insbesondere wird die Analyse von analogen Schaltungen im Detail betrachtet.

Beispiel: Es wird ein Zufalls-Experiment durchführt, dessen Ergebnis nicht vorhersagbar ist. Während das Ergebnis des Experiments nicht im Voraus bekannt ist, kann die Menge aller möglichen Ergebnisse bzw. Ausgänge des Experimentes bekannt sein. Dieser Satz von allen möglichen Ergebnissen eines Experiments wird als der Stichprobenraum S des Experiments bezeichnet. Beliebige Teilmengen des Stichprobenraums S können als Ereignis E bezeichnet werden. Ein Ereignis ist eine angegebene Menge möglicher einzelner Ergebnisse bzw. Punkte aus S. Notiert wird dies als $E \subset S$, wobei $\subset$ die Teilmenge bezeichnet. Das komplementäre Ereignis wird mit E^k angegeben. Das komplementäre Ereignis E^k eines Ereignisses E ist die Teilmenge aller Ereignisse aus S, die nicht in E sind. Das komplementäre Ereignis des Stichprobenraums S, ist per Definition leer und wird als Nullmenge bezeichnet (gekennzeichnet durch ∅). Um den Stichprobenraum S und die verschiedenen Ereignisse (E_1, E_2 usw.) zu illustrieren, ist es üblich, ein Venn-Diagramm zu verwenden (siehe folgende Abbildung 29). In einem Venn-Diagramm steht das Rechteck für den Stichprobenraum S. Alle möglichen Ergebnisse eines Experimentes werden als einzelne Punkte im Rechteck abgebildet. Die Ellipsen (E_1 oder E_2) umfasst alle Ergebnisse der jeweiligen Ereignisse.

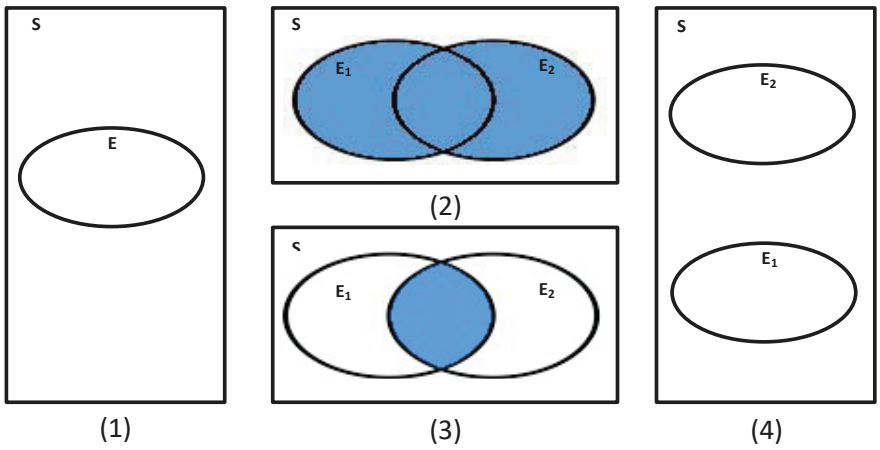


Abbildung 29. Venn-Diagramme, in (1) wird ein Ereignis E abgebildet, in (2) sind zwei überlappende (Vereinigungsmenge) unabhängige Ereignisse E_1 und E_2 dargestellt, in (3) eine Schnittmenge der unabhängigen Ereignisse E_1 und E_2 und (4) zeigt zwei sich ausschließende Ereignisse (ohne Schnittmenge).

Wie in Abbildung 29 zu sehen, können mehrere Mengenbeziehungen, die insbesondere für die Zuverlässigkeitstheorie interessant sind, unterschieden werden. So zeigt das erste Diagramm (1) eine Teilmenge ($E \subset S$). Das zweite Venn-Diagramm (2) eine Vereinigungsmenge ($E_1 \cup E_2$), mit allen Ergebnissen, die E_1 oder E_2 bzw. beiden Ereignissen zuzuordnen sind. Das dritte Venn-Diagramm (3) zeigt eine Schnittmenge aus E_1 und E_2 ($E_1 \cap E_2$). Das vierte Venn-Diagramm (4) der Abbildung 29 bildet zwei Ereignisse ab, die disjunkt sind ($E_1 \cap E_2 = \emptyset$), d.h. es handelt sich um zwei sich ausschließende Ereignisse, die nicht zugleich stattfinden können.

Die grundsätzlichen Regeln zur Berechnung der Wahrscheinlichkeit von Ereignissen werden im Folgenden in ihren Grundzügen beschrieben. Betrachtet man ein Experiment, dessen Stichprobenraum S ist, wird für jedes unabhängig auftretende Ereignis E eine Zahl P(E) (unabhängige Wahrscheinlichkeit) definiert, die die folgenden 3 Bedingungen erfüllt.

(I) $0 \leq P(E) \leq 1$

(II) $P(S) = 1$

(III) Für jede Sequenz von Ereignissen $E_1, E_2, \ldots$, die sich gegenseitig ausschließen, d. h. disjunkte Ereignisse ($E_x \cap E_y = \emptyset$ wenn $x \neq y$), gilt

$$P(\bigcup_{x=1}^{\infty} E_x) = \sum_{x=1}^{\infty} P(E_x)$$

Aus der ersten Bedingung (I) kann abgeleitet werden, dass die Wahrscheinlichkeit stets zwischen 0 und 1 liegen muss. Ein Ereignis E mit einer Wahrscheinlichkeit $P(E) = 0$ tritt nicht auf (oder ist unmöglich). Ist die Wahrscheinlichkeit $P(E) = 1$, so ist von dem sicheren Auftreten des Ereignisses auszugehen.

Wie zuvor bereits erwähnt schließen sich das komplementäre Ereignis E^k und das Ereignis E gegenseitig aus. Es gilt:

$E^k \cap E = \emptyset$

$E^k \cup E = S.$

$P(E^k) = 1 - P(E)$

Als Nächstes soll die Auftrittswahrscheinlichkeit für die unabhängigen Ereignisse E_1 und E_2 ($E_1 \cup E_2$) berechnet werden, siehe Venn-Diagramm (2) der Abbildung 29. Mit $P(E_1) + P(E_2)$ erhalten wir die Auftrittswahrscheinlichkeit der Ereignisse E_1 und E_2. Da Punkte von $E_1 \cap E_2$ sowohl in

E_1 als auch in E_2 liegen, werden diese doppelt gezählt. Folglich muss zwecks Richtigstellung $P(E_1 \cap E_2)$ von $P(E_1 \cup E_2)$ abgezogen werden:

$$P(E_1 \cup E_2) = P(E_1) + P(E_2) - P(E_1 \cap E_2)$$

wobei $P(E_1 \cap E_2) = P(E_1) * P(E_2)$ gilt (allgemeine Produktregel für Wahrscheinlichkeiten von unabhängigen Ereignissen).

Für drei Ereignisse (E_1, E_2, E_3) gilt:

$$\begin{aligned} &P(E_1 \cup E_2 \cup E_3) \\ &\quad = P(E_1) + P(E_2) + P(E_3) - P(E_1 \cap E_2) - P(E_1 \cap E_3) \\ &\quad - P(E_2 \cap E_3) + P(E_1 \cap E_3 \cap E_3) \end{aligned}$$

Die allgemeine Gleichung für unabhängige Ereignisse wird auch als Additionsregel für Wahrscheinlichkeiten bezeichnet:

$$(E_1 \cup E_2 \cup \dots E_m) = 1 - \prod_{i=1}^{m}(1 - P(E_i)) \qquad \text{(A.1)}$$

Schließen sich die Ereignisse E_1 und E_2 aus ($E_1 \cap E_2 = \emptyset$, siehe auch Venn-Diagramm (4) der Abbildung 29), muss die Schnittmenge nicht abgezogen werden und es gilt:

$$P(E_1 \cup E_2) = P(E_1) + P(E_2)$$

allgemein:

$$P(\bigcup_{x=1}^{\infty} E_x) = \sum_{x=1}^{\infty} P(E_x) \qquad \text{(A.2)}$$

Neben der Wahrscheinlichkeit von unabhängigen Ereignissen kann auch die bedingte Wahrscheinlichkeit bestimmt werden. Diese ist von Interesse, wenn zwischen dem Auftreten zweier Ereignisse ein Zusammenhang (z.B. Folgeausfälle) besteht. Die bedingte Wahrscheinlichkeit $P(E_1/E_2)$ ist die Wahrscheinlichkeit für das Eintreten eines Ereignis E_1 unter der Bedingung, dass das Ereignis E_2 eingetreten ist. In diesem Fall ist der relative Anteil von E_1 interessant, d.h. die Schnittmenge von $E_1 \cap E_2$. Die bedingte Wahrscheinlichkeit von E_1 in Abhängigkeit von E_2 ist definiert als

$$P(E_1/E_2) = \frac{P(E_1 \cap E_2)}{P(E_2)} \quad \text{für } P(E_2) > 0$$

Die Wahrscheinlichkeit für das Eintreten der Ereignisse E_1 und E_2 wird auch als Produktregel für Wahrscheinlichkeiten bezeichnet.

$$P(E_1 \cap E_2) = P(E_1/E_2) * P(E_2) = P(E_2/E_1) * P(E_1)$$

Das Eintreten der Ereignisse E_1 und E_2 ist unabhängig voneinander, wenn beispielsweise beim Eintreten von E_1 keine Informationen darüber vorhanden sind, ob ein anderes Ereignis eintritt oder nicht, d.h. die Ereignisse keinen Einfluss aufeinander haben. Dann gilt:

$$P(E_2/E_1) = P(E_2) \text{ sowie } P(E_1/E_2) = P(E_1).$$

In diesem Fall gilt: $P(E_1 \cap E_2) = P(E_1) * P(E_2)$

Die allgemeine Produktregel für Wahrscheinlichkeiten für eine Folge unabhängiger Ereignisse lautet

$$(E_1 \cap E_2 \cap \dots E_m) = P(E_1) * P(E_2) \dots * P(E_m)$$

oder in alternativer Schreibweise:

$$P(\bigcap_{x=1}^{\infty} E_x) = \prod_{i=1}^{m} P(E_i) \tag{A.3}$$

Aufbauend auf den oben ausgeführten Grundlagen kann ein sehr einfaches gleichverteiltes Wahrscheinlichkeitsmodell definiert werden. Sei S ein endlicher, nicht leerer Stichprobenraum mit n möglichen Ausgängen (e) mit $S = \{e_1, e_2, \dots, e_n\}$. So wird das Wahrscheinlichkeitsmodell auf S folgendermaßen definiert.

$$P(e) = \frac{1}{n} \quad f\ddot{u}r\ alle\ e\ P(E_2) > 0$$

Jedes Ereignis E hat n_E unterscheidbare Ergebnisse/Ausgänge. Durch die Gleichverteilung des Wahrscheinlichkeitsmodells ist die Wahrscheinlichkeit jedes Ereignisses aus S gleich

$$P(E) = \frac{n_E}{n}$$

Die kumulative Verteilungsfunktion F(x) (KVF, engl. probability distribution function / cumulative distribution function, CDF) der diskreten Zufallsvariablen X wird festgelegt als

$$F(x) = P(X \leq x) = \sum_{x_n \leq x} P(X = x_n) \tag{A.4}$$

Die KVF einer kontinuierlichen Zufallsvariablen X kann durch das Integral der Wahrscheinlichkeitsdichtefunktion $f(x)$ (engl. probability density function, PDF) ausgedrückt werden, siehe folgende Formel:

$$F(x) = \int_{-\infty}^{t} f(x)\,dx \qquad \text{mit } F(-\infty) = 0, \quad F(+\infty) = 1 \tag{A.5}$$

Die Wahrscheinlichkeitstheorie kann eingesetzt werden, um die Zuverlässigkeit eines Systems zu beschreiben. In diesem Fall gibt die Verteilungsfunktion an, ab ein System zum Zeitpunkt t ausgefallen ist. Da der Betriebsbeginn zum Zeitpunkt 0 angenommen wird, ist die untere Integralgrenze 0. Außerdem wird das Komplement $\overline{F}(x) = 1 - F(x)$ der Verteilungsfunktion als Überlebenswahrscheinlichkeit R(t) aufgefasst (engl. reliability function).

$$F(x) = \int_{0}^{t} f(x)\,dx \text{ und es gilt } \overline{F}(x) = R(x) = 1 - F(x) \qquad \text{mit } F(0^{-}) = 0, F(+\infty) = 1 \tag{A.6}$$

Die Funktion der Fehlerrate bzw. Ausfallrate h(x) (engl. hazard rate bzw. failure rate (function)) eines Objektes (z.B. eines Systems bzw. einer Komponente) wird definiert als $h(x) = f(x)/\overline{F}\,(x)$ wenn F(x) eine Dichtefunktion f(x) besitzt und für Werte von x: F(x) < 1 gilt.

Unterschiedliche Verteilungsfunktionen werden verwendet und eignen sich je nach Ausfallursache für die Beschreibung der Fehlerverteilung, der Wahrscheinlichkeitsdichte f(t), der kumulativen Verteilungsfunktion F(t) und der Ausfallrate h(t). Eine Vielzahl von Distributionen wurden in der Vergangenheit angewandt, um die Lebensdauer (und daraus abgeleitet MMTF, MTBF usw.) von elektronischen und mechanischen Komponenten zu beschreiben. Typische Fehlerverteilungen sind in Tabelle 18 zu finden. Die Fehlerrate für eine hohe Anzahl von identischen, nicht reparierbaren und unabhängigen Komponenten wird häufig mit einer Badewanne-Kurve dargestellt (siehe folgende Abbildung 30), die drei aufeinander folgende Betriebsphasen unterscheidet: In der ersten Phase zu Beginn der Nutzungsperiode (engl. burn-in period) fällt die Fehlerrate mit der Zeit ab, weil sich Herstellungsfehler während der Fabrikation vorwiegend am Anfang und danach nicht mehr so stark auswirken. In der zweiten Phase bleibt die Fehlerrate (nahezu) konstant und ist über die gesamte Lebensdauer gesehen die geringste Fehlerrate. Die zweite Phase stellt die Haupt-Nutzungsphase (engl. useful life period) eines Systems dar. Die dritte und letzte Phase ist die Abnutzungsperiode (engl. wear-out period) mit steigender Fehlerrate aufgrund von Alterungseffekten.

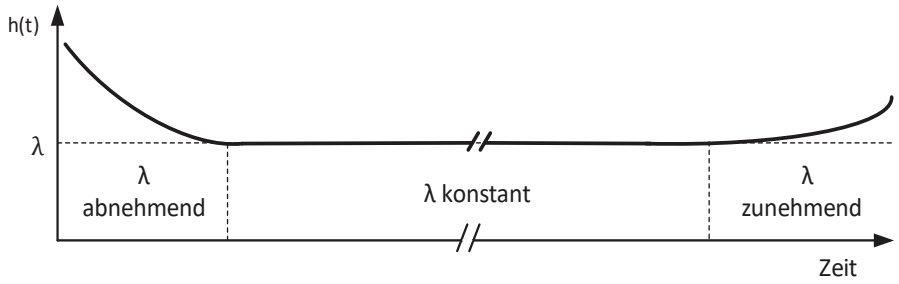


Abbildung 30. Typischer Verlauf der „Badenwannenkurve“

Wie in Tabelle 18 zu sehen, können für unterschiedliche Ausfallverläufe unterschiedliche Familien der Verteilungen genutzt werden. Entsprechend dem unterschiedlichen Verlauf der Fehlerrate $h(t)$ können drei primäre Ausfallverläufe unterschieden werden. Verteilungen mit einer steigenden, einer fallenden und einer konstanten Fehlerrate. Die Gamma- und die Weibull-Distribution können für einzelne Parameterwerte sowohl eine steigende als auch eine fallende Fehlerverteilung beschreiben. Die steigende Fehlerrate ist intuitiv so zu verstehen, dass eine Abnutzung der Komponente stattfindet, weil mit zunehmendem Alter Verschließeffekte auftreten. Eine Ausnahme bildet die logarithmische Normalverteilung, deren Fehlerrate $h(t)$ zu Beginn ansteigt und im weiteren Verlauf wieder fällt. Die einfachste Beschreibung der Fehlerrate über der Zeit ist die konstante Fehlerrate, die einer exponentiellen Verteilung entspricht. Die Beziehungen zwischen F(t), f(t), R(t) und h(t) werden in der folgenden Tabelle 17 zusammengefasst.

	F(t)	f(t)	R(t)	h(t)
F(t) =	-	$\int_0^t f(u)\,du$	$1 - R(t)$	$1 - \exp\left(-\int_0^t h(u)\,du\right)$
f(t)=	$\frac{d}{dt} F(t)$	-	$-\frac{d}{dt} R(t)$	$h(t) * \exp\left(-\int_0^t h(u)\,du\right)$
R(t)	$1 - F(t)$	$\int_t^\infty f(u)\,du$	-	$\exp\left(-\int_0^t h(u)\,du\right)$
h(t)	$\frac{dF(t)/dt}{1 - F(t)}$	$\frac{f(t)}{\int_t^\infty f(u)\,du}$	$-\frac{d}{dt} \ln R(t)$	-

Tabelle 17. Beziehung und Zusammenhang zwischen den Funktionen F(t), f(t), R(t) und h(t). Basierend auf (Rausand und Høyland 2004, S. 20).

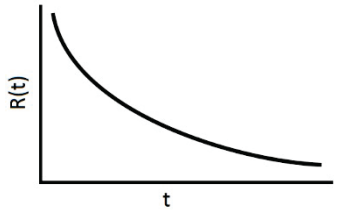
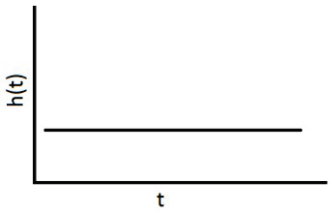
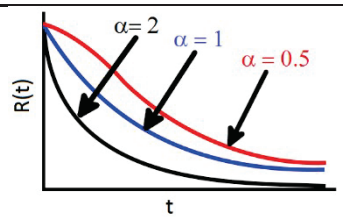
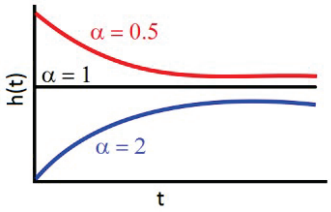
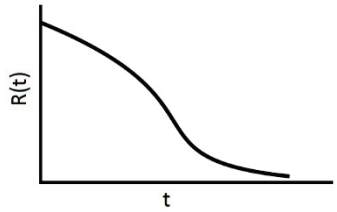
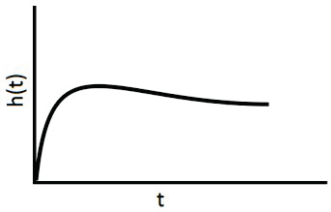
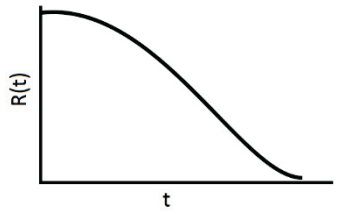
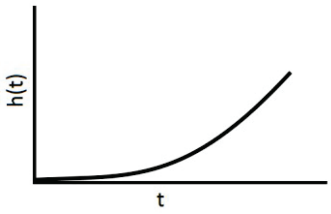
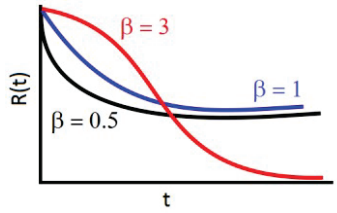
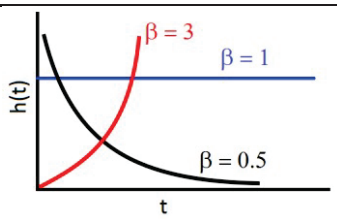
	Zuverlässigkeitsfunktion: $R(t) = 1 - F(t)$	Funktion der Fehlerrate: $h(t) = \frac{f(t)}{R(t)}$
Exponentiell	 $R(t) = e^{-\lambda t}$	 $h(t) = \lambda$
Gamma	 $R(t) = \frac{\lambda^{\alpha}}{\Gamma(\alpha)} \int_{t}^{\infty} \lvert t^{\alpha-1} e^{-\lambda t} dt$	 $h(t) = \frac{t^{\alpha-1} e^{-\lambda t}}{\int_{t}^{\infty} t^{\alpha-1} e^{-\lambda t} dt}$
Log-Normal	 $R(t) = 1 - \phi\left(\frac{\ln(t) - \mu}{\sigma}\right)$	 $h(t) = \frac{f(t)}{1 - \phi\left(\frac{\ln(t) - \mu}{\sigma}\right)}$
Normal	 $R(t) = 1 - \phi\left(\frac{t - \mu}{\sigma}\right)$	 $h(t) = \frac{f(t)}{1 - \phi\left(\frac{t - \mu}{\sigma}\right)}$
Weibull	 $R(t) = e^{-\left[\frac{(t-\gamma)^{\beta}}{\eta}\right]}$	 $h(t) = \frac{\beta}{\eta}\left(\frac{t - \gamma}{\eta}\right)^{\beta-1}$

Tabelle 18. Unterschiedliche Formen der Zuverlässigkeitsfunktion und der Fehlerrate der am häufigsten verwendeten Distributionen. Basierend auf (MIL-HDBK-338B 1998, S. 5–9).

ANHANG B: ERWEITERTES FEHLERMODELL FÜR DIE FEHLERINJEKTIONEN

Im Folgenden wird beispielshaft die Implementierung eines weiteren Fehlermodells und Fehlerinjektion mit VB-Skript dargestellt.

```
'Hier werden die einzelnen Kontaktkombinationen kurzgeschlossen. Es werden
Leitungen zwischen Kontakten hinzugefügt um die Fehlerart Kurzschluss zu er-
zeugen.
Sub add_wires_short_sequentially (ByRef schalttabelle, ele As Element, sch As
Schematic)
    Dim xmin As Long    Dim xmax As Long    Dim ymax As Long
    Dim ymin As Long    Dim xmid As Double  Dim ymid As Double

    Select Case ele.Nodes.Count
        Case 2  ' Elemente mit nur zwei Kontakten
            If ele.Nodes(1).Connected = True And ele.Nodes(2).Connected =
                 True
                 Then
                If ele.Nodes(1).x = ele.Nodes(2).x Or ele.Nodes(1).y =
                          ele.Nodes(2).y Then
                    sch.Wires.Add(ele.Nodes(1).x , ele.Nodes(1).y ,
                                  ele.Nodes(2).x , ele.Nodes(2).y)
                End If
            End If

        Case Else   ' alle Elemente mit mehr als drei Kontakten

            If ele.Nodes.Count > 1 Then

                For i = 1 To ele.Nodes.Count
                    If i = 1 Then
                        xmin = ele.Nodes(i).x          xmax = xmin
                        ymax = ele.Nodes(i).y          ymin=ymax
                    Else
                        If ele.Nodes(i).x > xmax Then
                            xmax = ele.Nodes(i).x
                        End If
                        If ele.Nodes(i).y > ymax Then
                            ymax = ele.Nodes(i).y
                        End If
                        If ele.Nodes(i).x < xmin Then
                            xmin = ele.Nodes(i).x
                        End If
                        If ele.Nodes(i).y < ymin Then
                            ymin = ele.Nodes(i).y
                        End If
                    End If
                Next i

                xmid = Round((xmax+xmin)/200)*100
                ymid = Round((ymax+ymin)/200)*100

                For i = 1 To  ele.Nodes.Count
                        If schalttabelle(i-1,ks_nummer) = 1 Then
                            If xmid = ele.Nodes(i).x Then
                             sch.Wires.Add(xmid,ymid,xmid,ele.Nodes(i).y)
                            ElseIf ymid = ele.Nodes(i).y Then
                             sch.Wires.Add(xmid,ymid,ele.Nodes(i).x,ymid)
                            Else
                             sch.Wires.Add(xmid,ymid,xmid,ele.Nodes(i).y)
         sch.Wires.Add(xmid,ele.Nodes(i).y,ele.Nodes(i).x,ele.Nodes(i).y)
                            End If
                        End If
                Next i
            End If
    End Select

    If ks_nummer < UBound(schalttabelle,2) Then     ks_nummer = ks_nummer + 1
    Else       ks_nummer = 0
    End If

End Sub
```

Abbildung 31. Beispielimplementierung (VB-Skript Subroutine) der Fehlerart Kurzschluss mit mehr als zwei Kontaktstellen

ANHANG C: DETAILS ZU UNTERSUCHTEN SCHALTUNGEN

Im Folgenden werden die mit dem hier vorgestellten Fehlersimulator untersuchten Schaltungen dargestellt.

Nr.	Beispielschaltungen	Bauelemente insgesamt	Widerstände	Kondensatoren	Transistoren	Spulen	Strom-/ Spannungquellen
1	LM741 Verstärker (National Semiconductor 2000)	38	13	1	20	0	3
2	VHF/UHF Verstärker (Gentzler und Leong 2003)	52	10	13	0	10	3
3	Limiter BSP (AWR Corporation 2015)	47	17	4	25	0	1
4	Spannungsstabilisator (Morgenstern 1997)	8	5	0	2	0	1
5	BJT_Amp_Complete: Small Signal (NI examples 2016a)	13	3	3	1	0	1
6	BJT_Amp_Complete: Large Signal (NI examples 2016a)	13	3	3	1	0	2
7	Gilbert_Cell (NI examples 2016b)	50	16	6	10	3	1
8	IAM_81_Gilbert_Cell_Noise (NI examples 2016c)	61	18	6	14	6	1
9	Limiter_HB_Aplac (NI examples 2016d)	68	19	4	25	0	1
10	Low_Power_Mixer (NI examples 2016e)	28	3	6	1	2	1
11	Nonlinear Noise Switch Views (NI examples 2016f)	28	3	6	1	0	1
12	UHF Power Amplifier (NI examples 2016g)	60	10	13	0	18	2
13	Impuls-Signalverstärker	167	67	9	34	0	1

Tabelle 19. Eigenschaften der exemplarischen Schaltungen

Schaltung 1: LM741 Operationsverstärker

Schaltung:

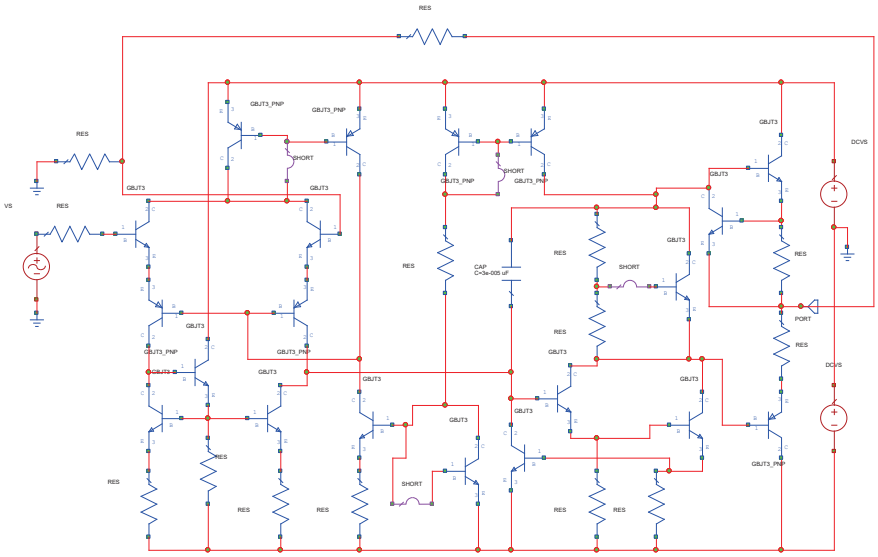


Abbildung 32. Schaltung des LM741 Operationsverstärkers. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 34 Bauelemente:

- Kurzschluss, Unterbrechung bei CAP, RES, GBJT3
- Jeweils 2 Parameterfehler bei CAP, RES

Relevante Graphen und Toleranzbereiche:

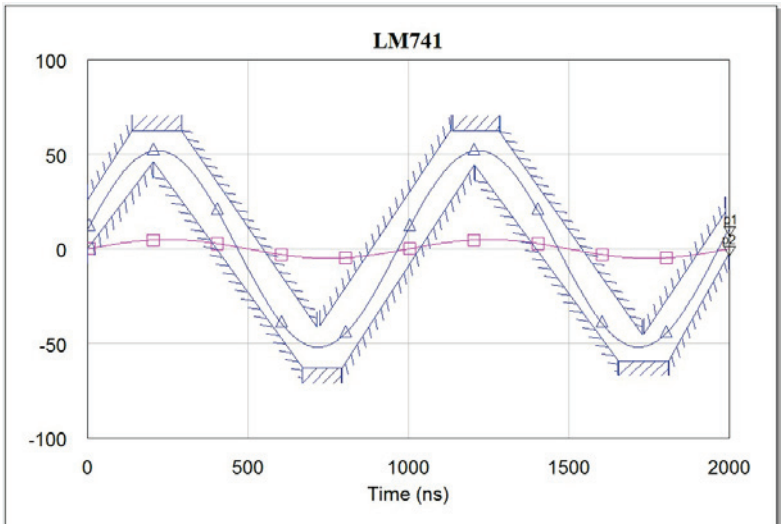


Abbildung 33. Ausgangssignale und Toleranzbereich des LM741 Operationsverstärkers. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 2: VHF/UHF Verstärker

Schaltung:

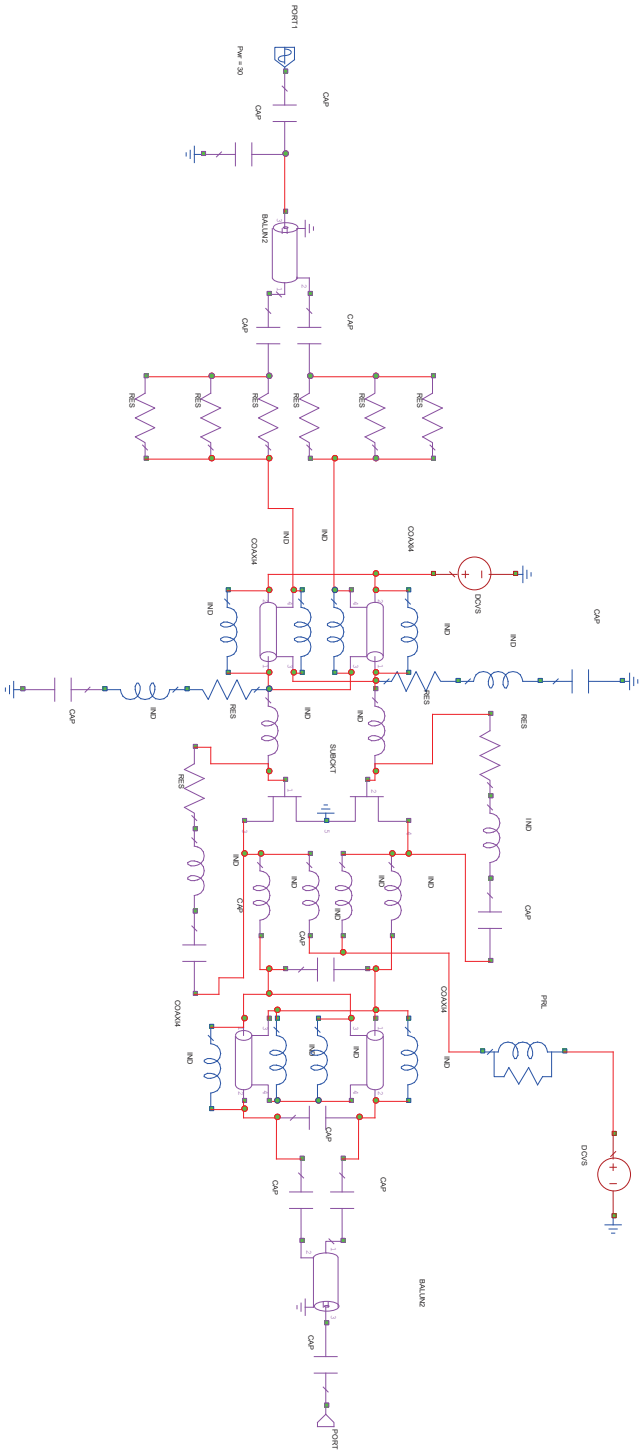


Abbildung 34. Schaltung des VHF/UHF Verstärkers. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 41 Bauelemente:

- Kurzschluss, Unterbrechung und 2 Parameterfehler bei CAP, RES, GBJT3, IND

Relevante Graphen und Toleranzbereiche:

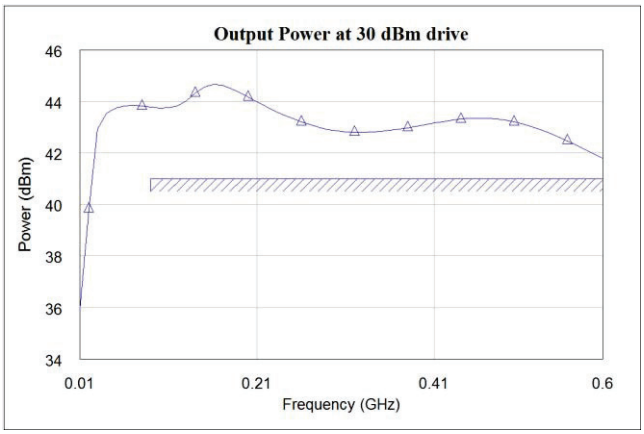


Abbildung 35. Ausgangssignale und Toleranzbereich des VHF/UHF Verstärkers. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 3: Limiter BSP (Bipolar Limiting Amplifier Circuit).
Schaltung:

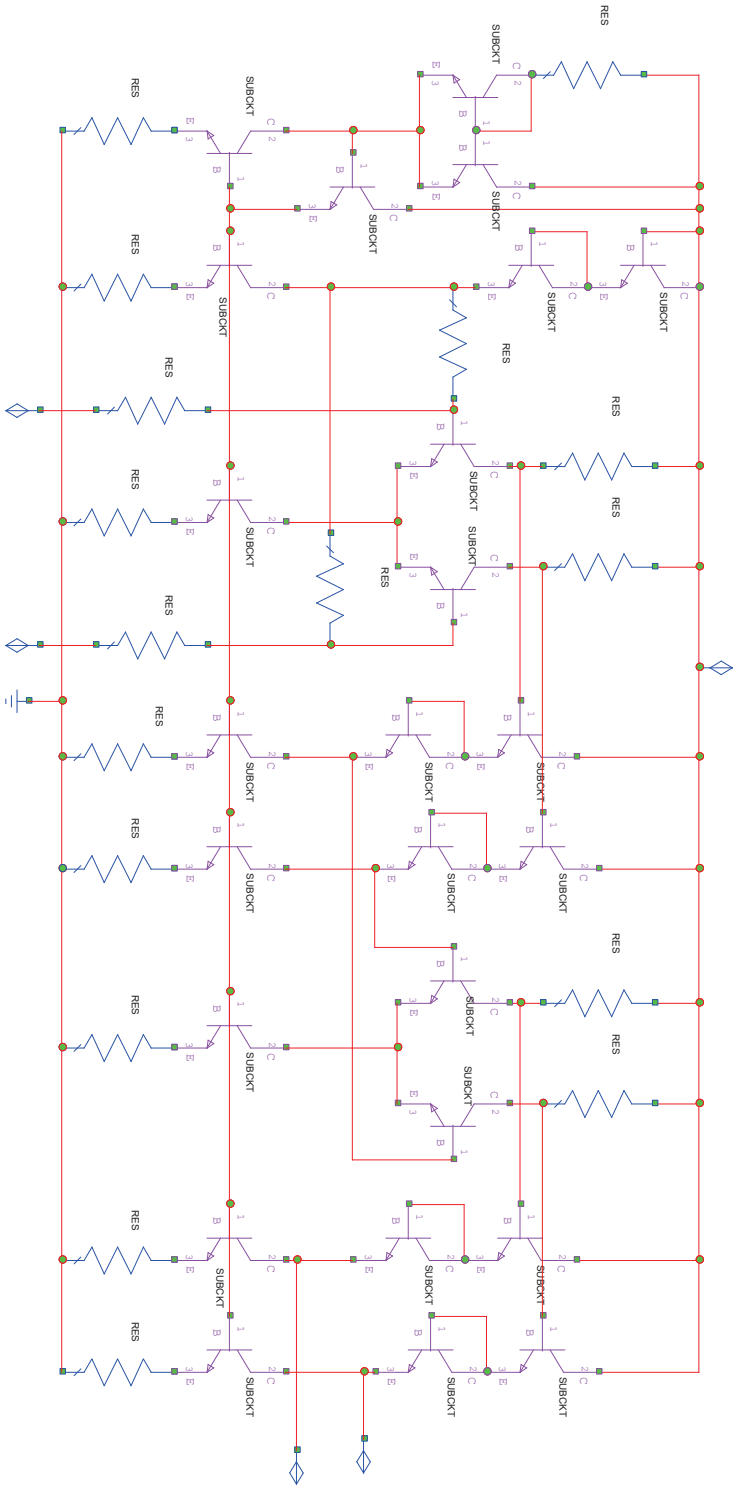


Abbildung 36. Schaltung des Limiter BSP. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 41 Bauelemente:

- Kurzschluss, Unterbrechung und 2 Parameterfehler bei CAP, RES, GBJT3, IND

Relevante Graphen und Toleranzbereiche:

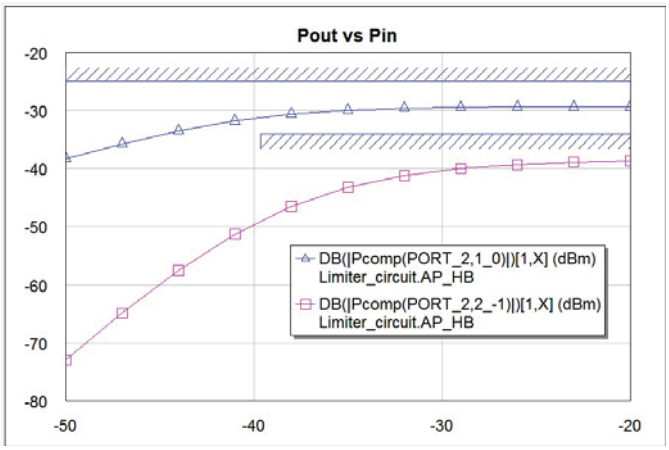


Abbildung 37. Ausgangssignale und Toleranzbereich des Limiter BSP. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 4: Spannungsstabilisator (Voltage Stabilizer)

Schaltung:

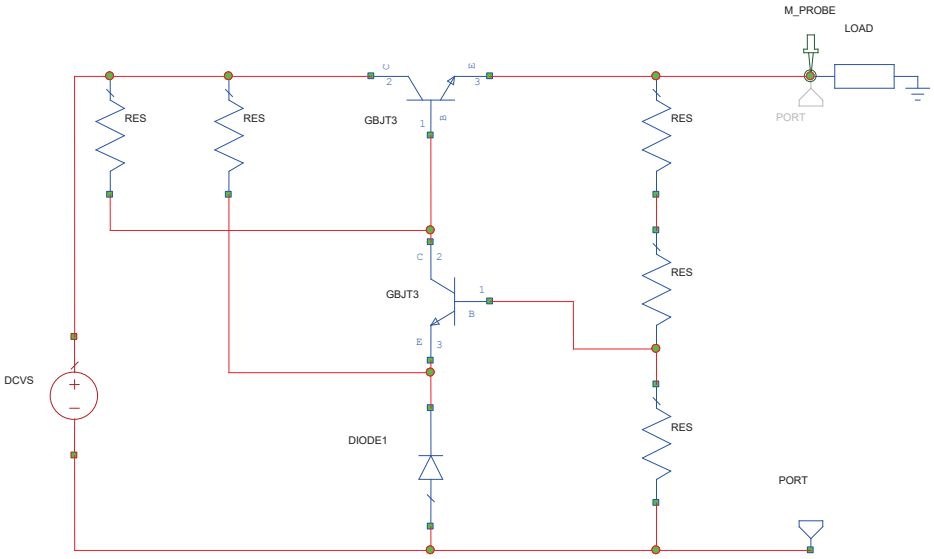


Abbildung 38. Schaltung des Spannungsstabilisators. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 6 Bauelemente:

- Kurzschluss, Unterbrechung und 4 Parameterfehler bei RES
- Kurzschluss, Unterbrechung und 12 Parameterfehler bei DIODE

Relevante Graphen und Toleranzbereiche:

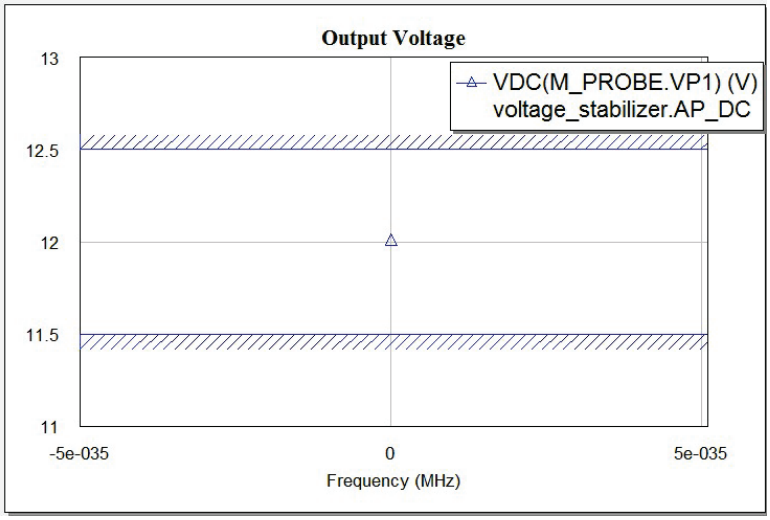


Abbildung 39. Ausgangssignale und Toleranzbereich des Spannungsstabilisators. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 5: BJT_Amp_Complete: Small Signal

Schaltung:

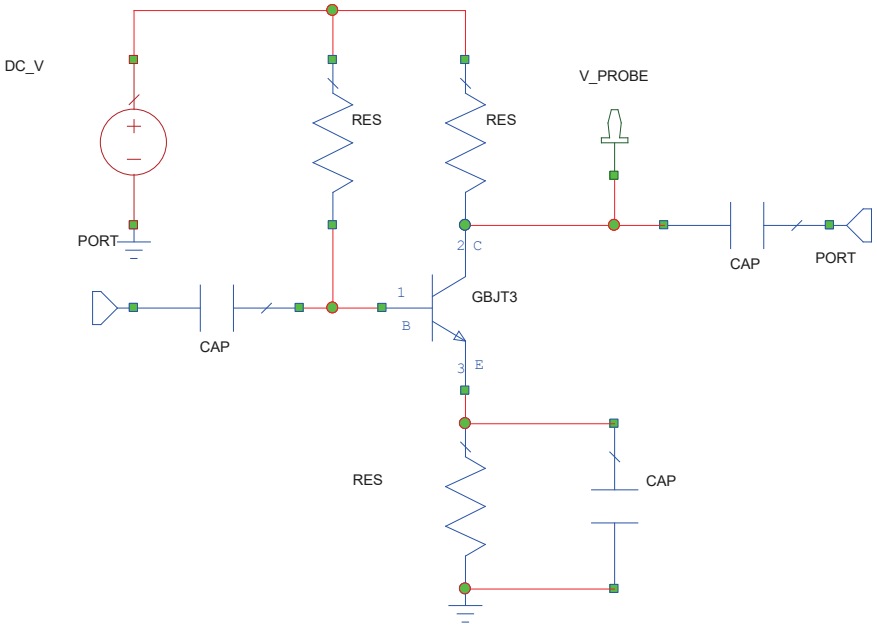


Abbildung 40. Schaltung des BJT_Amp_Complete: Small Signal. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 7 Bauelemente:

- Kurzschluss und Unterbrechung bei GBJT3
- Kurzschluss, Unterbrechung und 6 Parameterfehler bei CAP und RES

Relevante Graphen und Toleranzbereiche:

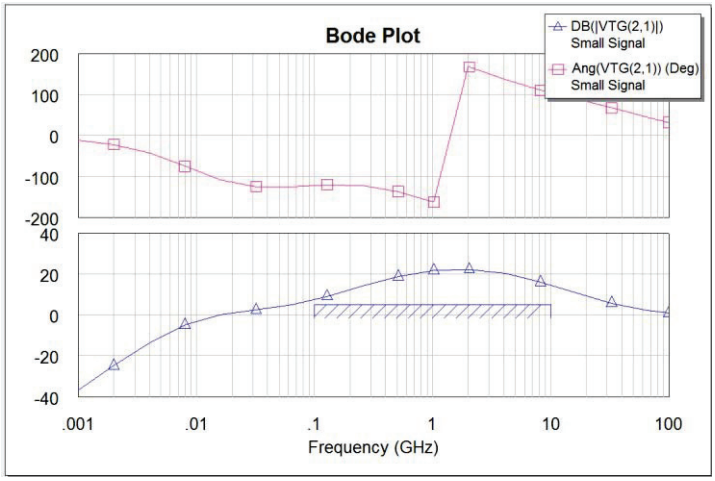


Abbildung 41. Ausgangssignale und Toleranzbereich des BJT_Amp_Complete: Small Signal. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 6: BJT_Amp_Complete: Large Signal

Schaltung:

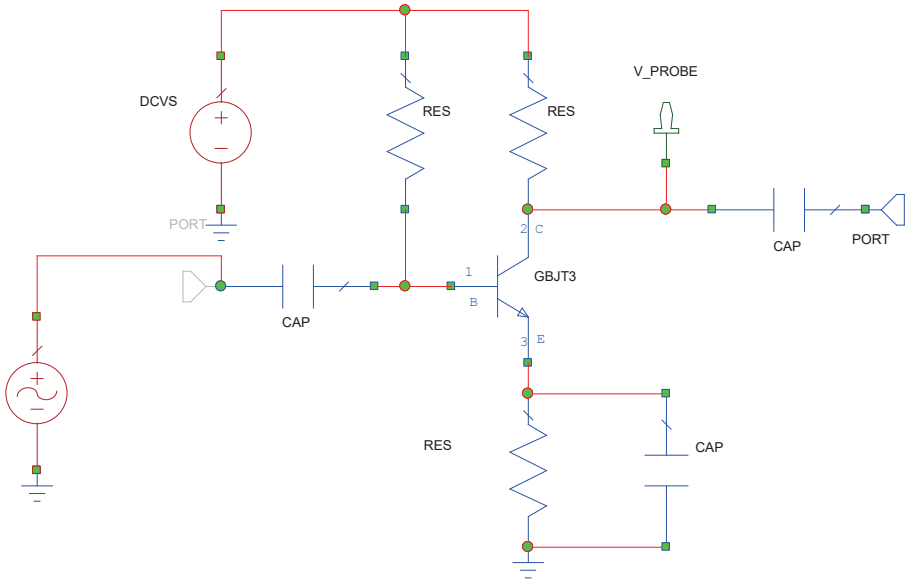


Abbildung 42. Schaltung des BJT_Amp_Complete: Large Signal. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 7 Bauelemente:

- Kurzschluss und Unterbrechung bei GBJT3, CAP und RES
- 6 Parameterfehler bei CAP und RES

Relevante Graphen und Toleranzbereiche:

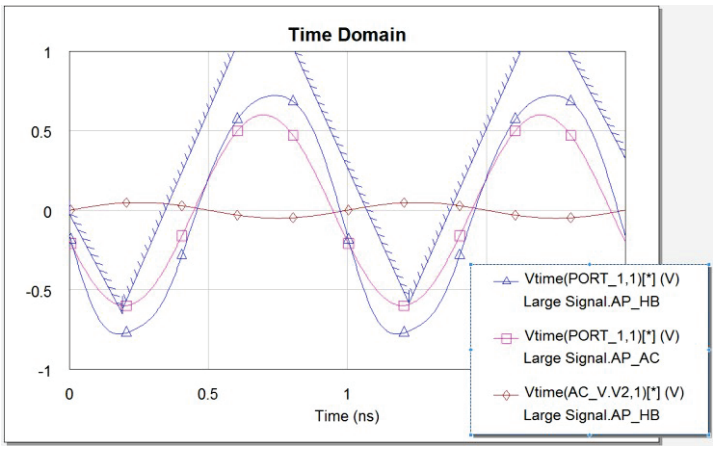


Abbildung 43. Ausgangssignale und Toleranzbereich des BJT_Amp_Complete: Large Signal. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 7: Gilbert Cell

Schaltung: Design für eine Bipolar-Transistor-Mixer mit Gilbert Cell-Anordnung

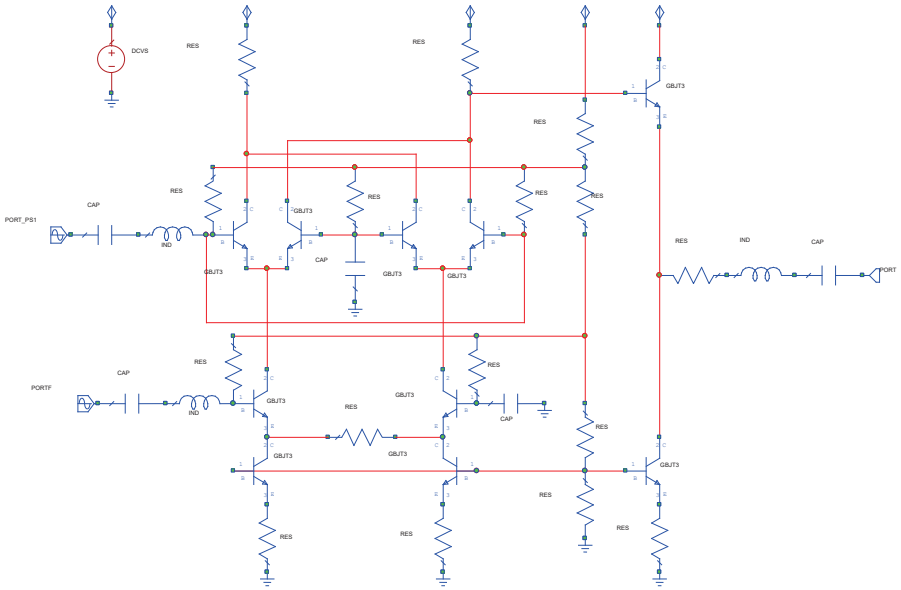


Abbildung 44. Schaltung der Gilbert Cell. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 35 Bauelemente:

- Kurzschluss und Unterbrechung bei RES, IND, DCVS, GBJT3, CAP

Relevante Graphen und Toleranzbereiche:

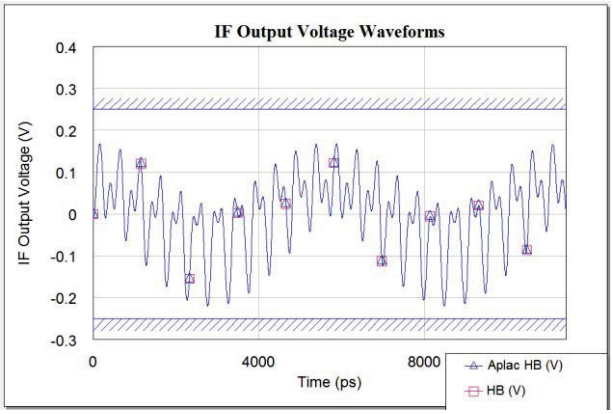


Abbildung 45. Ausgangssignale und Toleranzbereich der Schaltung Gilbert Cell. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 8: IAM_81_Gilbert_Cell_Noise

Schaltung: Design für eine Mixer-Rausch-Analyse für einen Gilbert-Cell Mixer. Beispiel entnommen aus (Hewlett Packard 1999)

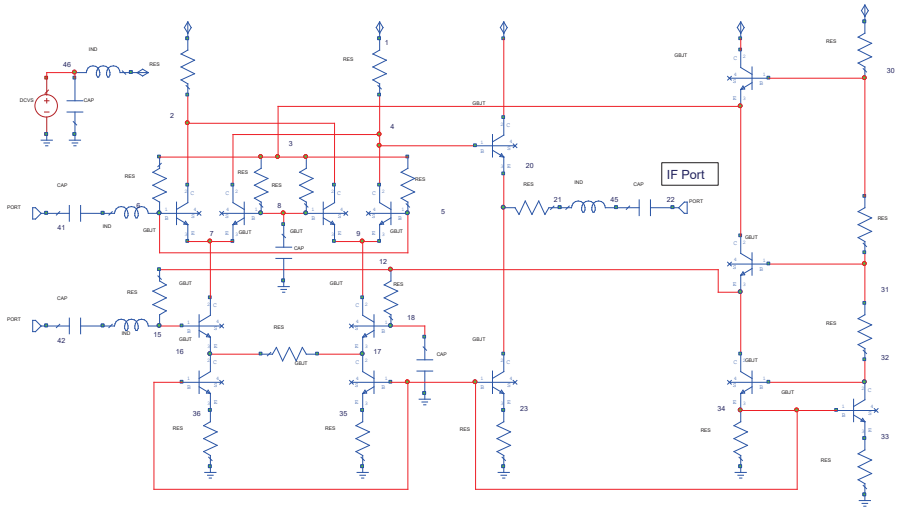


Abbildung 46. Schaltung der IAM_81_Gilbert_Cell_Noise. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 43 Bauelemente:

- Kurzschluss, Unterbrechung und 6 Parameterfehler bei RES, IND, CAP
- Kurzschluss und Unterbrechung bei DCVS, GBJT

Relevante Graphen und Toleranzbereiche:

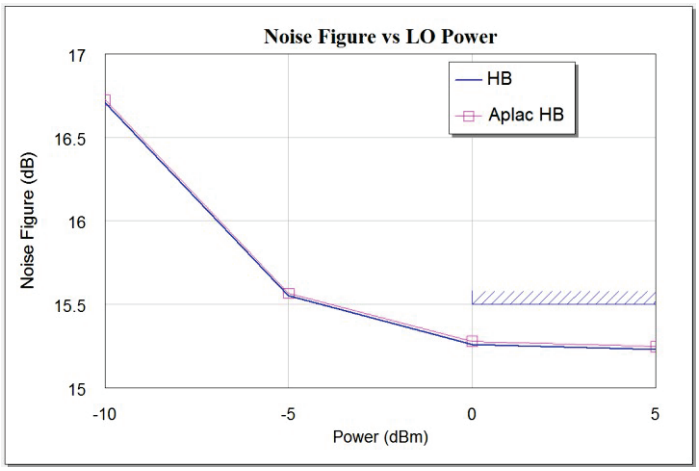


Abbildung 47. Ausgangssignale und Toleranzbereich bei IAM_81_Gilbert_Cell_Noise. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 9: Limiter_HB_Aplac

Schaltung:

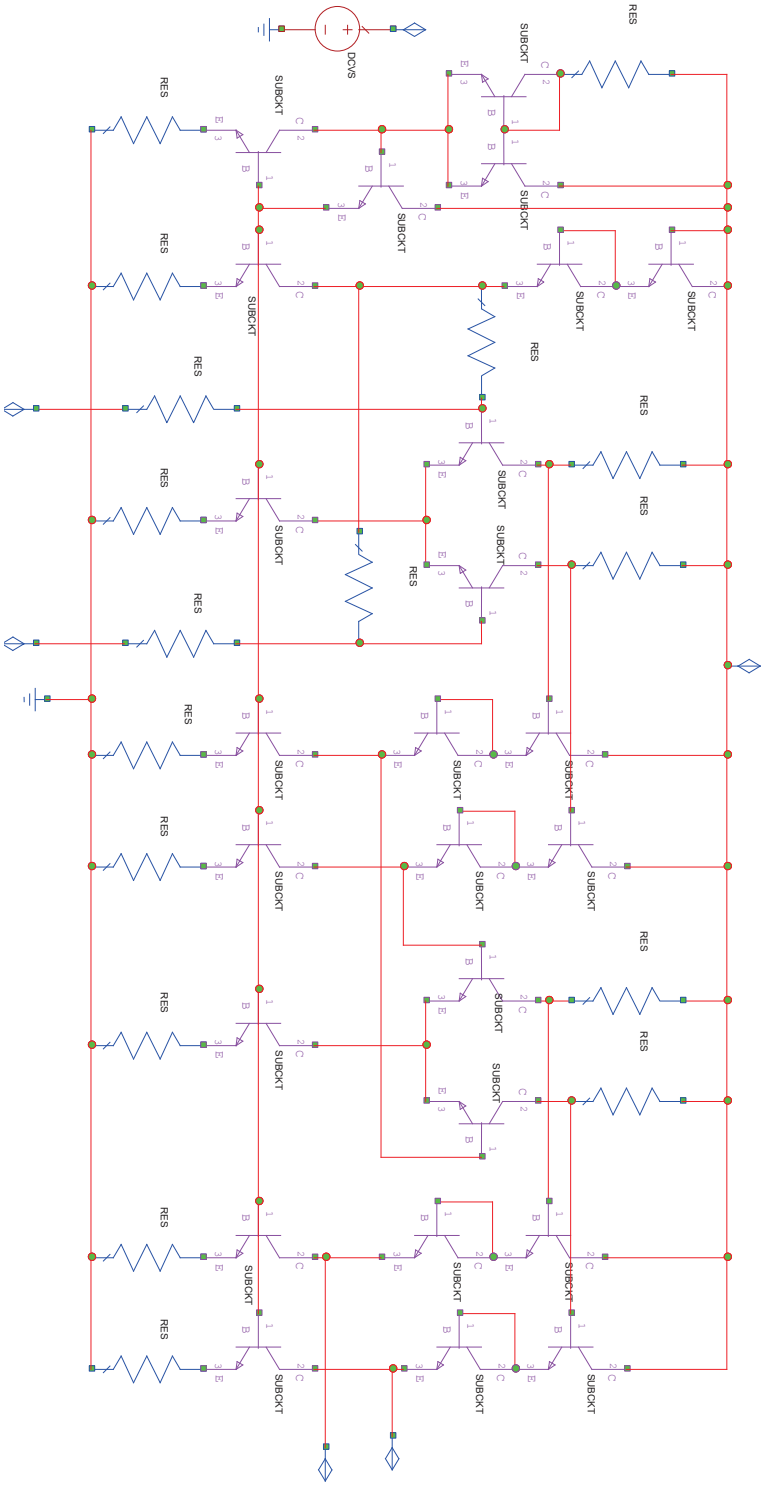


Abbildung 48. Schaltung Limiter_HB_Aplac. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 22 Bauelemente:

- Kurzschluss, Unterbrechung und 6 Parameterfehler bei RES, DCVS, CAP

Relevante Graphen und Toleranzbereiche:

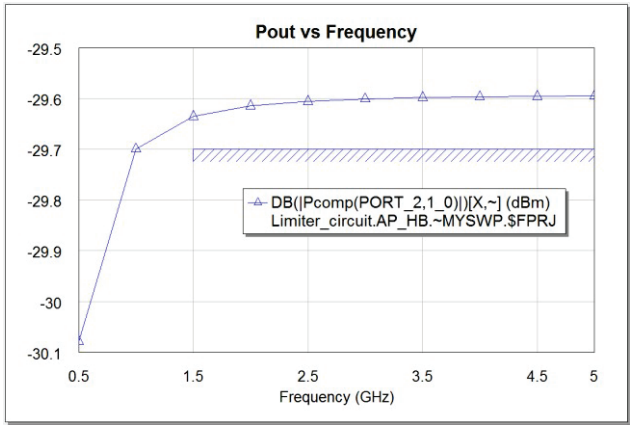


Abbildung 49. Ausgangssignale und Toleranzbereich der Schaltung Limiter_HB_Aplac. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 10: Low_Power_Mixer

Schaltung:

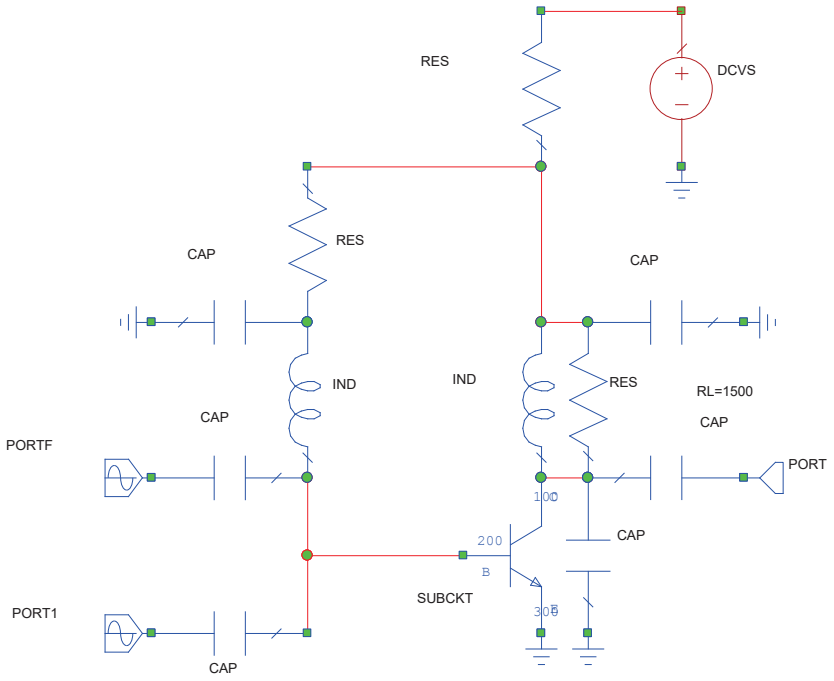


Abbildung 50. Schaltung des Low_Power_Mixer. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 12 Bauelemente:

- Kurzschluss, Unterbrechung und 6 Parameterfehler bei RES, IND, CAP, DCVS

Relevante Graphen und Toleranzbereiche:

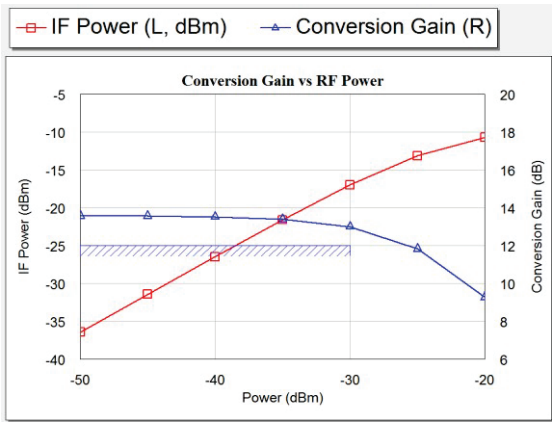


Abbildung 51. Ausgangssignale und Toleranzbereich der Schaltung Low_Power_Mixer. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 11: Nonlinear Noise Switch Views

Schaltung:

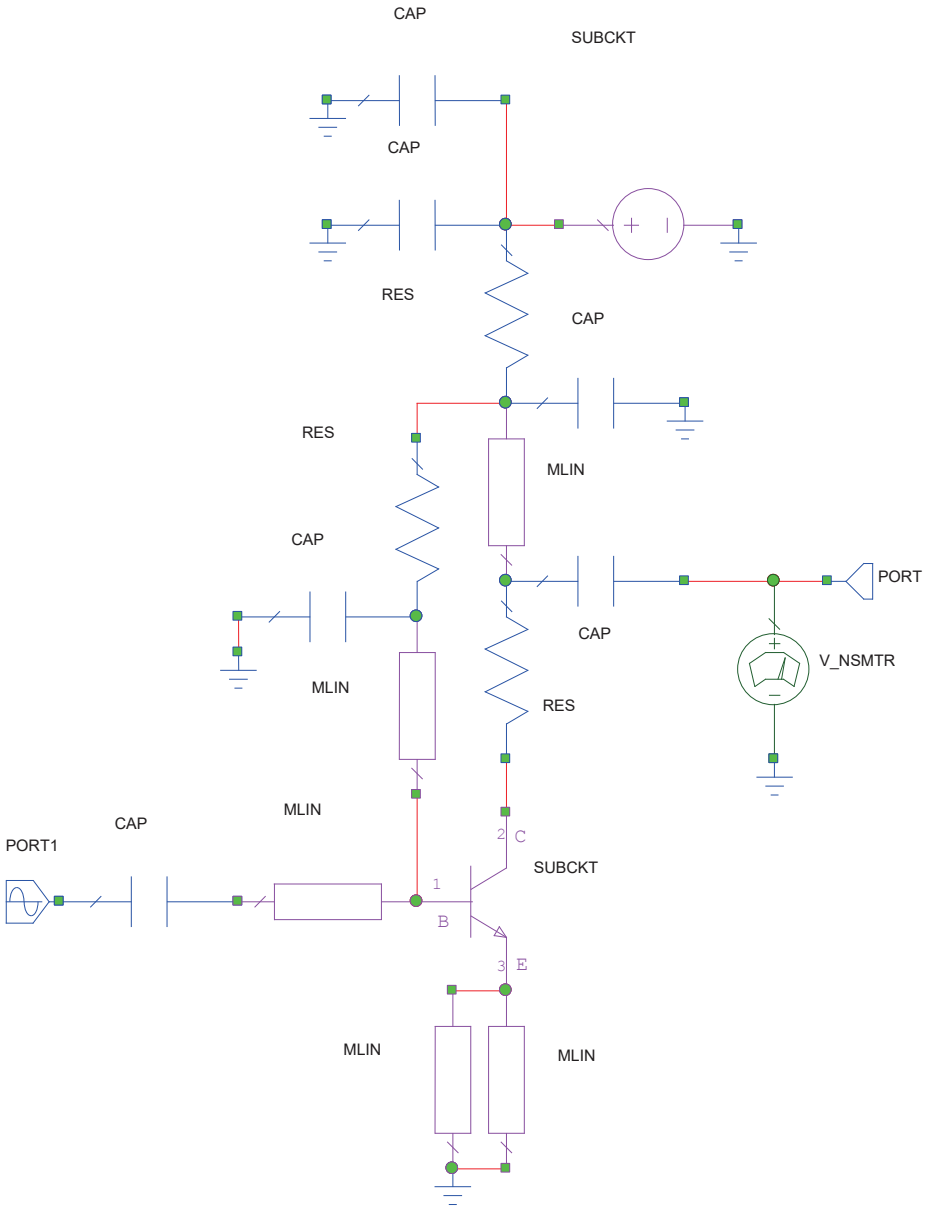


Abbildung 52. Schaltung des Nonlinear Noise Switch Views. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 10 Bauelemente:

- Kurzschluss, Unterbrechung bei RES, CAP

Relevante Graphen und Toleranzbereiche:

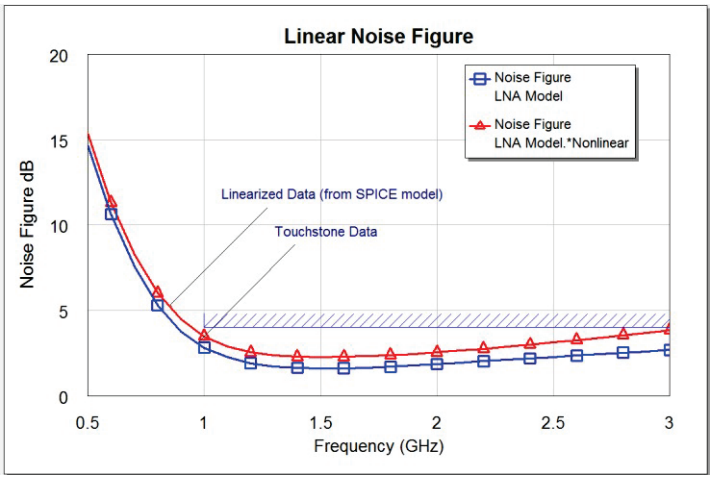


Abbildung 53. Ausgangssignale und Toleranzbereich der Schaltung Nonlinear Noise Switch Views. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 12: UHF VHF Power Amplifier

Schaltung:

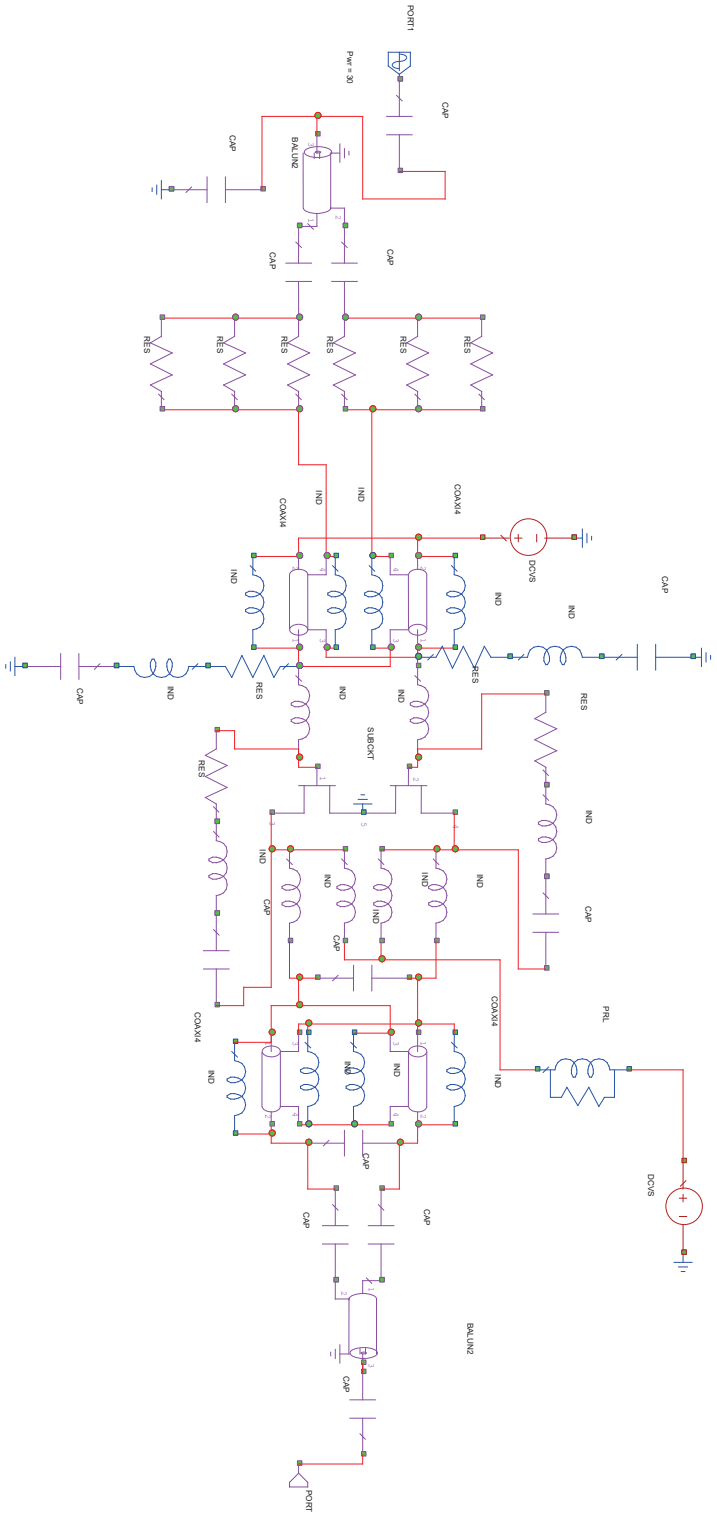


Abbildung 54. Schaltung des UHF VHF Power Amplifiers. Entnommen aus (NI AWR Design Environment 2016)

Fehlermodelle und Fehlerinjektion in 10 Bauelemente:

- Kurzschluss, Unterbrechung bei RES, CAP, IND, DCVS

Relevante Graphen und Toleranzbereiche:

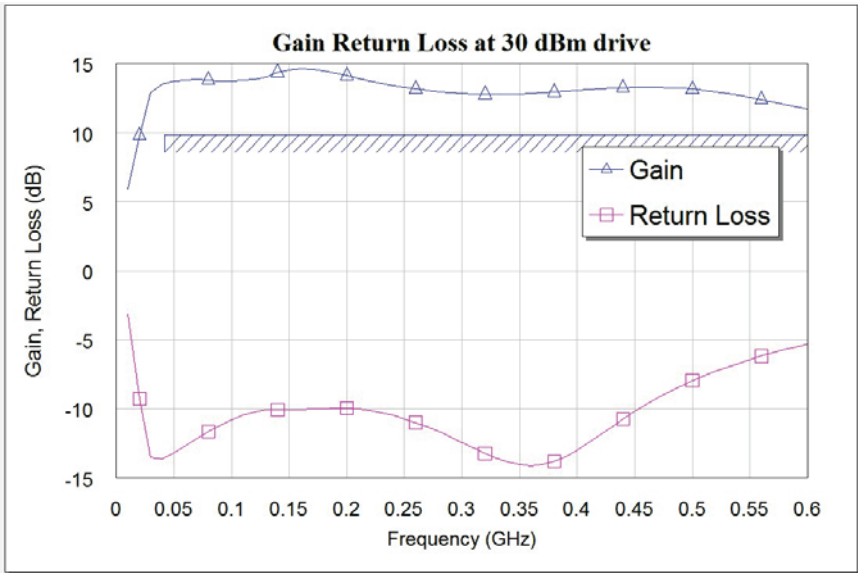


Abbildung 55. Ausgangssignale und Toleranzbereich des UHF VHF Power Amplifier. Entnommen aus (NI AWR Design Environment 2016)

Schaltung 13: Impuls-Signalverstärker

Schaltung:

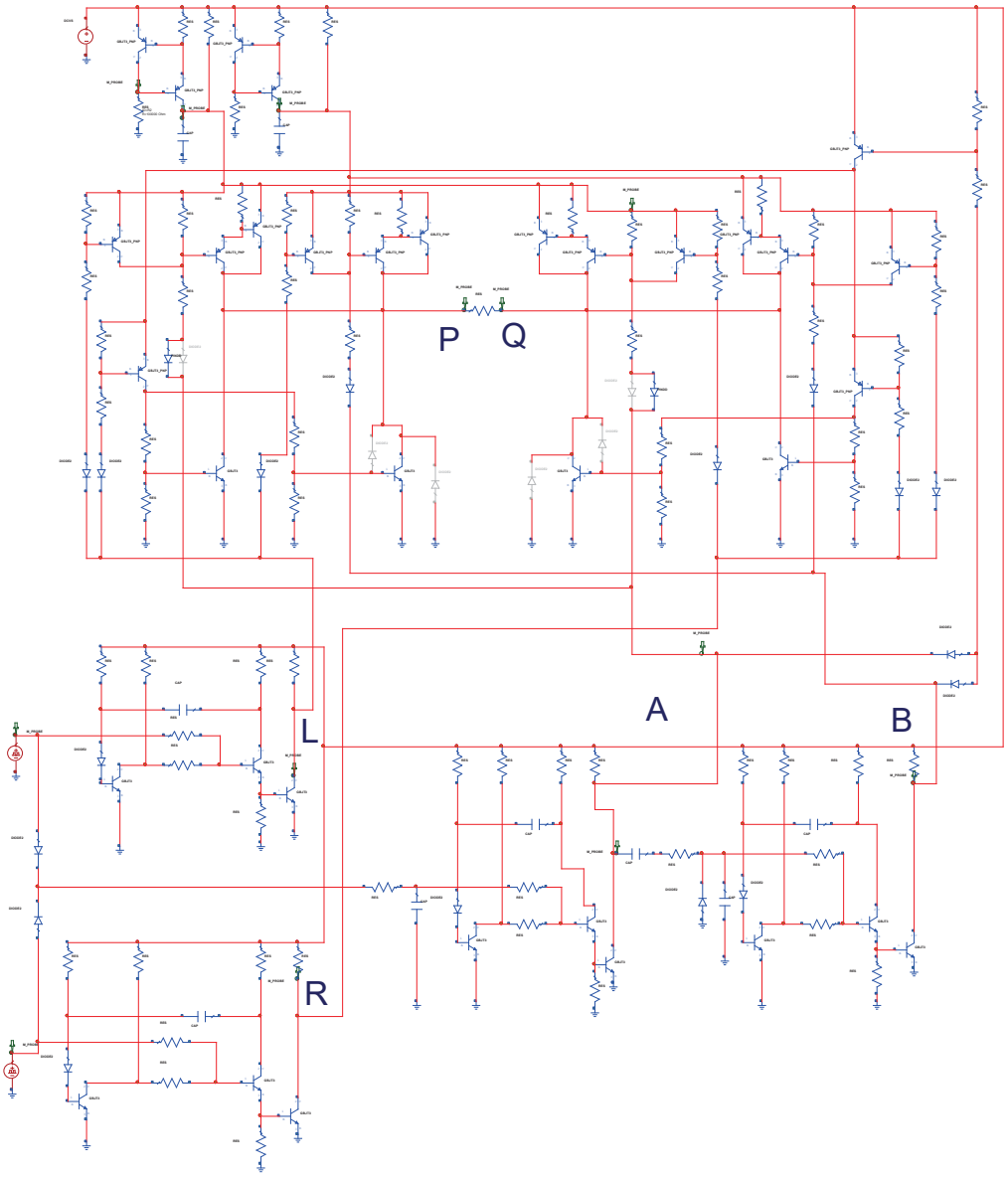


Abbildung 56. Schaltung des Impuls-Signalverstärker

Fehlermodelle und Fehlerinjektion in 10 Bauelemente:

- Kurzschluss, Unterbrechung und 2 Parameterfehler bei CAP.

Relevante Graphen und Toleranzbereiche:

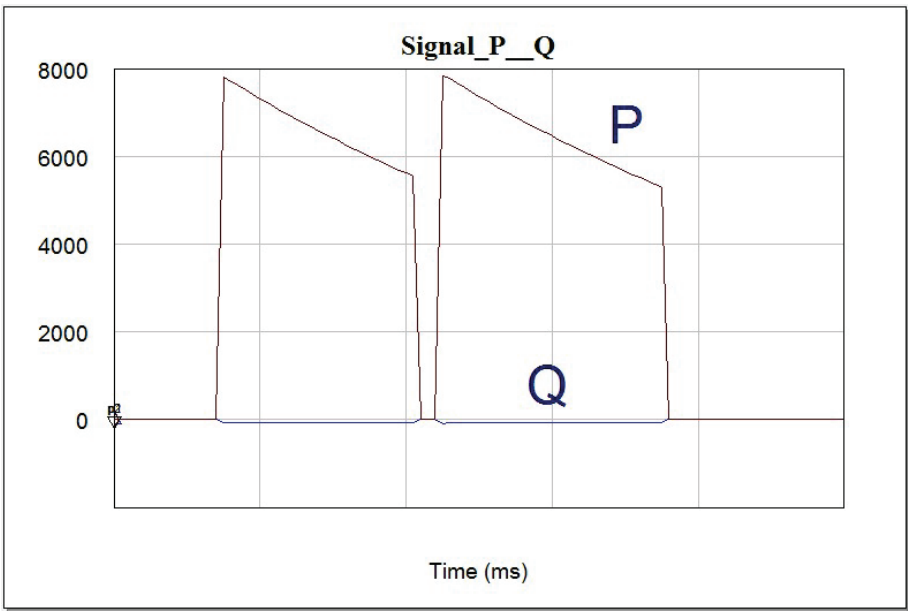


Abbildung 57. Ausgangssignale P und Q des Impuls-Signalverstärkers. Gezeigt wird die am Ausgang (P und Q) gemessene elektrische Leistung (in mW).

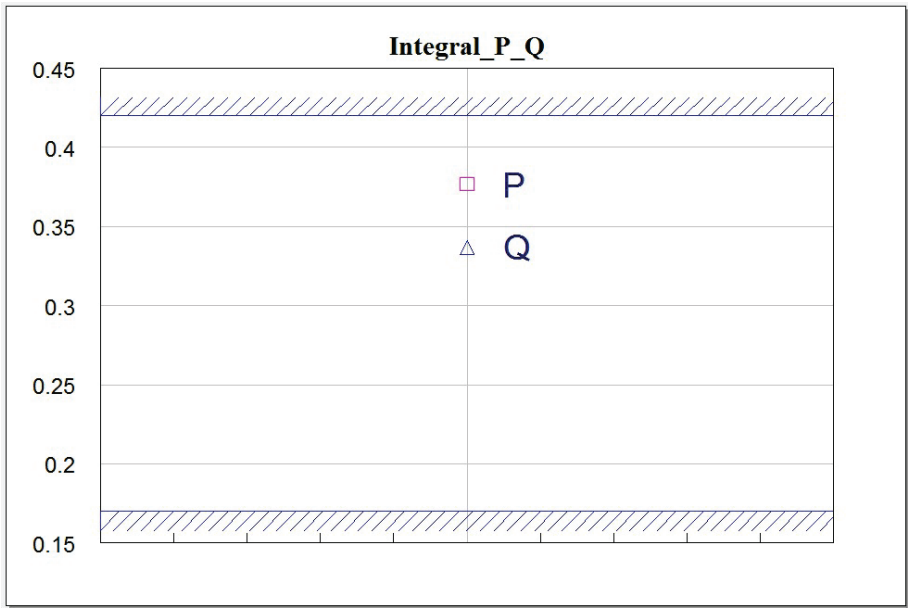


Abbildung 58. Ausgangssignale und Toleranzbereich des Impuls-Signalverstärkers. Dargestellt wird die elektrische Energie, die sich als Zeit-Integral der gemessenen elektrischen Leistung von Q und P aus Abbildung 57 berechnet.