Universität Rostock

Traditio et Innovatio

Beiträge zur Steigerung der Energieeffizienz FPGA-basierter Anwendungen

Der Fakultät für Informatik und Elektrotechnik der Universität Rostock
zur Erlangung des akademischen Grades eines

Doktor-Ingenieur (Dr.-Ing.)

vorgelegte Dissertation von
Christoph Niemann
geb. am 05.08.1988 in Potsdam
aus Rostock

Rostock, 12. September 2022

https://doi.org/10.18453/rosdok_id00004204

Dissertation
Fakultät für Informatik und Elektrotechnik

Christoph Niemann
Institut für Angewandte Mikroelektronik und Datentechnik

Gutachter:

- Prof. Dr.-Ing. Dirk Timmermann
  Universität Rostock
  Fakultät für Informatik und Elektrotechnik
  Institut für Angewandte Mikroelektronik und Datentechnik
  Albert-Einstein-Str. 26, 18059 Rostock

- Prof. Dr.-Ing. Albrecht Rothermel
  Universität Ulm
  Institut für Mikroelektronik
  Albert-Einstein-Allee 43, 89081 Ulm

Tag der Einreichung: 12.09.2022
Tag der Verteidigung: 30.01.2023

Lebenslauf

Christoph Niemann
geb. am 05.08.1988 in Potsdam, Deutschland

Februar 2015 – Mai 2022	Wissenschaftlicher Mitarbeiter an der Universität Rostock, Institut für angewandte Mikroelektronik und Datentechnik
April 2013 – Januar 2015	Studium M. Sc. Elektrotechnik an der Universität Rostock
Oktober 2009 – September 2012	Studium B. Sc. Wirtschaftsingenieurwesen an der Universität Rostock
August 2001 – Juli 2008	Gymnasium und Abitur am Espengrundgymnasium in Potsdam

Selbständigkeitserklärung

Hiermit versichere ich, dass ich die von mir vorgelegte Arbeit mit dem Titel

Beiträge zur Steigerung der Energieeffizienz FPGA-basierter Anwendungen

selbständig und ohne unerlaubte fremde Hilfe angefertigt, keine anderen als die angegebenen Quellen und Hilfsmittel verwendet und die den verwendeten Quellen und Hilfsmitteln wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Rostock, den 12. September 2022

_____

Christoph Niemannn

Danksagung

Es ist eine schöne Tradition, nach der Fertigstellung einer umfangreichen Arbeit in sich zu gehen und zu reflektieren, ohne wen diese Arbeit so nicht zu Stande gekommen wäre. Ich möchte die Gelegenheit nutzen, hier den Menschen zu danken, die mich auf meinem Weg unterstützt und begleitet haben.

Mein großer Dank gilt meinem Doktorvater Professor Dirk Timmermann für das mir entgegengebrachte Vertrauen. Die Einstellung am Institut für Angewandte Mikroelektronik und Datentechnik und seine langjährige Begleitung, Unterstützung und Förderung haben diese Arbeit erst möglich gemacht. Auch Professor Albrecht Rothermel möchte ich für die Begutachtung dieser Dissertation danken. Es war eine große Freude, seine sehr tiefschürfenden Fragen zur Arbeit in der Disputation zu diskutieren.

Ich möchte mich ganz herzlich bei allen Koleg:innen am Institut für Angewandte Mikroelektronik und Datentechnik für die tolle Arbeitsatmosphäre und die gegenseitige Unterstützung bedanken. Für das Lektorat dieser Arbeit möchte ich Danielle Gluns, Martin Kasparick und insbesondere Michael Rethfeldt danken, der auch viele meiner Publikationen korrigiert und mit vielen konstruktiven Hinweisen vorangebracht hat.

Abschließend möchte ich auch den Menschen meinen Dank aussprechen, die mich privat begleitet haben. Ich danke meinen Eltern Dorothea und Ulrich Niemann. Ganz besonders möchte ich „rike“ und meinem Sohn Jori danken. Darüber hinaus habe ich das große Glück, dass mich viele weitere Menschen unterstützen, deren Aufzählung den Rahmen dieser Danksagung sprengen würde. Stellvertretend möchte ich hier Sebastian Unger, Martin Kasparick, Bastian Sommerfeld, Johannes Rutkowsky und Ole Erik Schulz nennen. Danke für eure großartige Unterstützung und guten Rat in schwierigen Situationen.

Zusammenfassung

Über mehrere Jahrzehnte gelang es der Halbleiterindustrie, durch die Skalierung der Prozesstechnologie enorme Leistungssteigerungen zu erreichen. Auch der Energiebedarf für Rechenoperationen wurde dabei extrem gesenkt. Aktuell verlangsamt sich dieser Trend jedoch deutlich. Wichtige Zukunftstechnologien wie künstliche Intelligenz und moderne Kommunikationssysteme wie 5G und 6G verlangen wiederum nach einer wachsenden Rechenleistung. Gleichzeitig darf der Energiebedarf jedoch nicht beliebig steigen. Weltweit betrachtet lag der Energiebedarf der Informationsverarbeitungs- und Kommunikationssysteme 2012 bei 920 TWh. Dies entspricht etwa 5 % des gesamten Verbrauchs an elektrischer Energie. 2018 lag der Verbrauch bereits bei 2000 TWh. Die Herausforderung liegt also darin, trotz einer weniger dynamischen Entwicklung der Prozesstechnologie mehr Rechenleistung pro Energieeinheit bereitzustellen.

Eine Möglichkeit dazu ist die Verwendung spezialisierter Hardware. Dabei spielen Field Programmable Gate Arrays (FPGAs) aufgrund ihrer Flexibilität eine wichtige Rolle. Mit ihnen kann, verglichen mit Central Processing Units (CPUs), eine um mehrere Größenordnungen höhere Energieeffizienz erreicht werden. Im Vergleich zu Application-specific Integrated Circuits (ASICs) wiederum sind FPGAs bezüglich der Energieeffizienz unterlegen. ASICs bieten jedoch nicht die häufig benötigte Flexibilität eines FPGAs. Es werden also Verfahren benötigt, um die Energieeffizienz von FPGAs zu steigern. Wie beschrieben genügt es nicht, nur auf die weitere Skalierung der Complementary Metal–Oxide–Semiconductor (CMOS)-Technologie zu vertrauen. Vielmehr werden einerseits Methoden benötigt, um das Potential der gegebenen Prozesstechnologie eines FPGAs besser auszuschöpfen. Andererseits müssen Anstrengungen unternommen werden, um Anwendungen energieeffizienter auf die Hardwareressourcen von FPGAs abzubilden.

Die vorliegende Arbeit verschreibt sich dieser Aufgabe und leistet dazu zwei Beiträge. Zum einen wird ein Adaptive Voltage Scaling (AVS)-System vorgestellt, das gezielt auf die Besonderheiten von FPGAs abgestimmt ist. Dabei wird insbesondere die Sensorkalibrierung und -platzierung in den Mittelpunkt der Betrachtung gestellt. Eine Implementierung und experimentelle Evaluation zeigen das erhebliche Potential des erarbeiteten Konzepts zur Einsparung von Energie auf.

Zum anderen wird Approximate Computing als eine Möglichkeit, Anwendungen energieeffizienter umzusetzen, untersucht. Dazu werden zunächst verschiedene Gütemaße und Optimierungsziele für approximierte arithmetische Strukturen diskutiert. Um die Vergleichbarkeit der vielfältigen vorgeschlagenen Lösungen in diesem Bereich zu verbessern, wird zudem eine Evaluationsmethodik erarbeitet. Für das gewählte Optimierungsziel wird ein gezielt für moderne FPGA-Architekturen optimierter approximierter Multiplizierer entworfen. Dieser ist bezüglich der Leistungsaufnahme in Relation zu mehreren Gütemaßen für die Qualität der Approximation paretooptimal.

Abstract

Over several decades, the semiconductor industry has succeeded in achieving enormous performance increases by process technology scaling. The energy required for computing operations has also been extremely reduced. However, this trend is currently slowing down considerably. Important future technologies like artificial intelligence and modern communication systems such as 5G and 6G require increasing computing capabilities. Meanwhile, the energy consumption needs to be limited. Viewed globally, the energy demand of information processing and communications systems was 920 TWh in 2012. This corresponds to around 5% of the total consumption of electrical energy. In 2018, the power consumption has increased to 2000 TWh. The challenge therefore lies in processing more computational workload per unit of energy despite a less dynamic evolution of process technology.

One way to do this is to use specialized hardware. FPGAs are an important technology for this strategy. They enable improvements in energy efficiency of multiple orders of magnitude when compared to CPUs. But, compared to ASICs, FPGAs are inferior in terms of energy efficiency. However, ASICs do not offer the required flexibility of an FPGA. Hence, there is an urgent need for approaches to increase the energy effiiency of FPGAs. As described, it is not sufficient to rely on further scaling of the CMOS technology. Therefore, methods are needed to better exploit the potential of the given process technology of an FPGA. Moreover, efforts must be made to map applications more energy-efficiently to the hardware resources of FPGAs. The present work is dedicated to this task and makes two contributions.

On the one hand, an AVS system is presented, which is specifically taylored towards the properties of FPGAs. In particular, sensor calibration and placement are the focus of attention. An implementation and experimental evaluation show the considerable potential of the developed concept for saving energy.

On the other hand, approximate computing is investigated as a possibility to implement applications more energy-efficiently. For this purpose, different measures of quality and optimization goals for approximate arithmetic circuits are discussed. In order to improve the comparability of approaches in this area, an evaluation methodology is developed. For the chosen optimization objective, an approximate multiplier specifically optimized for modern FPGA architectures is proposed. The evaluation shows, that it is pareto-optimal regarding its power consumption and the quality of the approximation.

Inhaltsverzeichnis

Abbildungsverzeichnis **XVII**

Tabellenverzeichnis **XIX**

Abkürzungsverzeichnis **XXI**

1 Einleitung **1**
1.1 Problemstellung und Zielsetzungen der Arbeit 1
1.2 Aufbau der Arbeit 2
1.3 Konventionen und Fachtermini 5

2 Grundlagen **7**
2.1 Energieverbrauch von VLSI-Systemen 7
2.1.1 Dynamische Verlustleistung 7
2.1.2 Statische Verlustleistung 9
2.1.3 Zusammenhang zwischen Versorgungsspannung und Verlustleistung . . 11
2.2 Timing in digitalen VLSI-Systemen 11
2.2.1 Synchronität digitaler VLSI-Systeme 11
2.2.2 Timing in synchronen Designs 12
2.2.3 Timing-Analyse 13
2.3 Variable Einflussgrößen auf das Timing 14
2.3.1 Einfluss von Modellbildung und Electronic Design Automation (EDA)-Tools 15
2.3.2 Einfluss der Prozessvariation 16
2.3.3 Einfluss der Bedingungen zur Systemlaufzeit 18
2.3.3.1 Versorgungsspannung 18
2.3.3.2 Temperatur 18
2.3.3.3 Cross Talk 19
2.4 Alterungseffekte 19
2.4.1 Bias Temperature Instability (BTI) 20
2.4.2 Hot Carrier Injection (HCI) 21
2.4.3 Elektromigration 21
2.5 Clock Tree 21
2.6 Dynamic Voltage and Frequency Scaling (DVFS) 24
2.7 Besonderheiten von Field Programmable Gate Arrays (FPGAs) 25
2.7.1 Architektur 25
2.7.2 Technische Umsetzung auf Transistorlevel 26

3 Adaptive Voltage Scaling (AVS) **29**
3.1 Motivation 29
3.2 Stand der Wissenschaft und Technik 31
3.2.1 Offline-Ex-Situ-Messung 31
3.2.2 Offline-In-Situ-Messung 33
3.2.3 Online-Ex-Situ-Messung 34

3.2.4 Online-In-Situ-Messung . . . 36
3.3 Datenabhängigkeit von auf Monitoring-Flipflops basierenden AVS-Ansätzen ohne Fehlerkorrektur . . . 40
3.3.1 Grundsätzliche Funktionsweise . . . 40
3.3.2 Simulation zur quantitativen Bewertung der Datenabhängigkeit Monitoring-Flipflop-basierter AVS-Ansätze . . . 42
3.3.3 Ergebnisse und Diskussion . . . 43
3.3.4 Nicht detektierte Fehler . . . 46
3.3.5 Übertragbarkeit der Simulationsergebnisse auf reale Systeme . . . 47
3.3.6 Schlussfolgerung . . . 48
3.4 Analyse sensorbasierter AVS-Systeme . . . 49
3.5 Kalibrierung von AVS-Sensoren . . . 51
3.5.1 Anforderungen an eine AVS-Sensorkalibrierung . . . 51
3.5.2 Konzeption einer Kalibrierungsmethode für AVS-Sensoren . . . 53
3.6 Online-Sensoren . . . 58
3.7 Bemessung des residualen Guard Bands . . . 62
3.8 Implementierung und experimentelle Auswertung . . . 66
3.8.1 Testplattform . . . 66
3.8.2 Data Retention Voltage . . . 67
3.8.3 Heater-Design . . . 68
3.8.4 Experimentelle Validierung der Online-Sensoren . . . 71
3.8.5 Implementierung des AVS-Systems . . . 75
3.8.6 Anwendungsdesigns . . . 78
3.9 Ergebnisse und Diskussion . . . 79
3.9.1 Reduktion der Verlustleistung durch AVS . . . 82
3.9.2 Untersuchung des Einflusses der Prozessvariation . . . 83
3.9.3 Zuverlässigkeit . . . 86
3.9.4 Einordnung im Vergleich zum Stand der Technik . . . 88
3.10 Zwischenfazit . . . 90

4 Für FPGA optimierte approximierte Multiplizierer 93
4.1 Motivation . . . 93
4.2 Anforderungen und Gütemaße für Approximate Computing . . . 94
4.3 Stand der Wissenschaft und Technik . . . 96
4.4 Approximierte Multiplizierer zur optimalen Nutzung von FPGA-Logikressourcen 98
4.4.1 Approximierte Kompressoren . . . 99
4.4.2 Für FPGA-Logikressourcen optimierte Partialproduktbildung . . . 101
4.4.3 Statistische Optimierung des Kompressionsbaumes . . . 104
4.5 Evaluationsmethodik . . . 105
4.5.1 Gütemaße der Approximation . . . 107
4.5.2 Definition des Ressourcenbedarfs . . . 107
4.5.3 Definition des Delays . . . 107
4.5.4 Ermittlung der Leistungsaufnahme . . . 109
4.6 Evaluation und Analyse . . . 110
4.6.1 Ergebnisse . . . 111

4.6.2 Skalierbarkeit 123
4.6.3 Anwendungsbeispiel: approximierter Faltungskern für digitale Bildverarbeitung 123
4.7 Zwischenfazit 125

5 Zusammenfassung **127**
5.1 Adaptive Voltage Scaling (AVS) 127
5.2 Approximate Computing 128
5.3 Ausblick 129

A Literaturverzeichnis **I**

B Liste der Veröffentlichungen (peer-reviewed) **XVII**

C Liste weiterer Veröffentlichungen und Konferenzbeiträge **XIX**

D Liste der betreuten studentischen Arbeiten **XXI**

Abbildungsverzeichnis

1.1 Struktur der vorliegenden Arbeit 4
2.1 Durch Schaltvorgänge hervorgerufene Stromflüsse in einem CMOS-Inverter . 8
2.2 Komponenten der statischen Verlustleistung (Abbildung nach [A 19]) 9
2.3 Einteilung von Logik-Schaltnetzen [A 23] 11
2.4 Aufbau eines synchronen Schaltnetzes aus kombinatorischer Logik und Speicherelementen 12
2.5 Setup und Hold Time (Abbildung nach [A 26]) 13
2.6 Zur Entwurfszeit unbekannte Einflussgrößen auf das reale Delay; „OCV“ steht für „On-Chip Variation“. 15
2.7 Schnitt durch einen Die; Durch die unterschiedlichen mechanischen Eigenschaften von Kupfer und Dielektrikum sind beim Chemical Mechanical Polishing (CMP) räumlich korrelierte On-Chip Variation (OCV) entstanden. (Abbildung nach [A 31]) 17
2.8 Launch und Capture Path (Abbildung nach [A 26]) 22
2.9 Launch und Capture Path unter dem Einfluss lokaler Variabilität 23
2.10 Spannungs-Taktfrequenz-Arbeitspunkte bei angewandtem Dynamic Voltage and Frequency Scaling (DVFS); 7nm Fin Field-Effect Transistor (FinFET) Technologie (Prozessdaten aus [A 38], [A 59]) 24
2.11 Prinzipieller Aufbau eines FPGAs aus Logic Blocks, Connection Blocks und Switch Blocks (Abbildung nach [A 61]) 26
2.12 16:1-Multiplexer (MUX) auf Basis von N-type Metall-Oxid-Semiconductor (NMOS)-Pass-Transistoren, wie er in den FPGAs aus Intels Stratix-10 Serie verwendet wird [A 67] 27
3.1 Vergleichende Darstellung von DVFS und AVS 30
3.2 Aufbau eines In-Situ-Monitoring-Flipflops nach Nunez-Yanez [A 94] 40
3.3 Wave Form für eine vom Monitoring-Flipflop erkannte späte Transition 41
3.4 Gate-Level-Simulation zur quantitativen Bestimmung der durch datenabhängige Delays verursachten Fehler bei Monitoring-Flipflop-basierten AVS-Systemen . 44
3.5 Beispielhafte Wave Form für einen nicht erkannten Timing-Fehler; In diesem Fall wird statt des korrekten Signalwertes der falsche Wert ’Übergangszustand B’ in das Register am Ende des Pfades übernommen, ohne dass das Monitoring-Flipflop eine späte Signaltransition erkennt. 47
3.6 Nicht detektierte Fehler in Abhängigkeit von der Verteilung der Eingangsvektoren und des Evaluationszeitraums 48
3.7 Steuerung von Taktfrequenz und Versorgungsspannung durch Auswahl aus vordefinierten VF-Arbeitspunkten in einem Dynamic Voltage and Frequency Scaling (DVFS)-System 50
3.8 Blockdiagramm der charakteristischen geschlossenen Regelschleife von AVS-Systemen (abgeändert nach [B 2]) 50
3.9 Verteilung des Delays eines 8-Bit-Ripple-Carry-Addierers bei gleichverteilten, unkorrelierten Eingangsvektoren (in Intervallen zu je 5 ps Breite) 51
3.10 Kritische Versorgungsspannung verschiedener Dies (Abbildung nach [B 1]; Original ©2019 IEEE) 53

3.11 Begrenzte Auflösung der Kalibrierung durch Schrittweite der Spannungssenkung. Hier beispielhaft dargestellt für drei verschiedene Pfade eines Designs . 56
3.12 Ablauf der Sensorkalibrierung 57
3.13 Schematischer Vergleich der Sensorplatzierung mittels Hard Macro und der in dieser Arbeit vorgeschlagenen freien Platzierung innerhalb eines Bereichs . . . 60
3.14 Partielle Guard-Bands mit und ohne AVS (Abbildung übersetzt aus [B 2]) . . . 62
3.15 Einordnung verschiedener Störgrößen bezüglich der zeitlichen Größenordnung, innerhalb derer sie das Timing beeinflussen. Daten aus [A 19], [A 120] sowie eigenen Messungen 63
3.16 Geschlossene AVS-Regelschleife mit residualem Guard Band (abgeändert aus [B 2]) 63
3.17 Testplattform für die experimentelle Evaluation 67
3.18 In dieser Arbeit verwendetes Heater-Design 70
3.19 Ring-Oszillator-Implementierung 72
3.20 Beispielhafte Auswertung einer Sensorregion; Die betragsmäßig größte relative Abweichung jedes Sensors wurde rot hervorgehoben. 73
3.21 Prozess zum automatisierten Erstellen von Test Patterns für FPGA-Anwendungen (Abbildung übersetzt aus [B 1]; Original ©2019 IEEE) 76
3.22 Gesamtablauf des vorgeschlagenen AVS-Systems für FPGAs (Abbildung übersetzt aus [B 2]) 77
3.23 Verlauf der Spannung während der Kalibrierung und beginnendem Anwendungsbetrieb (Design A, FPGA A) 80
3.24 Verlauf der Spannung während der Kalibrierung und beginnendem Anwendungsbetrieb; Alle FPGAs wurden mit demselben Bitfile (Design A) initialisiert, das auch für Abbildung 3.23 verwendet wurde. 81
3.25 Versorgungsspannung, Strom und Leistungsaufnahme bei angewandtem AVS und Kalibrierungsprozedur (Design B, FPGA E) 83
3.26 Bei angewandtem AVS stellen sich auf Instanzen von baugleichen FPGAs verschiedene Versorgungsspannungen ein. 84
3.27 Leistungsaufnahme verschiedener Instanzen des gleichen FPGA-Modells mit und ohne AVS (Design A) 85
4.1 Kritischer Pfad eines 4:2-Kompressors (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE) 99
4.2 Anordnung der Partialprodukt-Blöcke (PPBs) für einen 8x8-Bit-Multiplizierer (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE) 101
4.3 Zuordnung der PPB-Ausgänge zur ersten Kompressionsstufe (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE) 104
4.4 Ablauf der automatisierten Evaluation von approximierten Multiplizierern . . . 106
4.5 Delay bei Vivado-Timing-Analyse mit kombinatorischer Logik als Pipeline-Stufe zwischen Registern 108
4.6 Input- und Output-Buffer dominieren das Ergebnis der Vivado-Timing-Analyse, wenn eine rein kombinatorische Logik als Toplevel-Design implementiert wird. 108
4.7 Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Mean Relative Error (MRE) (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE) . 112

4.8 Vergleich mit dem Stand der Technik bezüglich Delay und Mean Relative Error (MRE) (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE) 112
4.9 Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und MRE . 113
4.10 Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Worst-Case Relative Error (WCRE) 114
4.11 Vergleich mit dem Stand der Technik bezüglich Delay und Worst-Case Relative Error (WCRE) 114
4.12 Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und WCRE 115
4.13 Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Normalized Mean Error Distance (NMED) 116
4.14 Vergleich mit dem Stand der Technik bezüglich Delay und Normalized Mean Error Distance (NMED) 116
4.15 Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und NMED 117
4.16 Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Worst-Case Normalized Error Distance (WCNED) 118
4.17 Vergleich mit dem Stand der Technik bezüglich Delay und Worst-Case Normalized Error Distance (WCNED) 118
4.18 Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und WCNED 119
4.19 Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Error Rate (ER) 120
4.20 Vergleich mit dem Stand der Technik bezüglich Delay und Error Rate (ER) . . 120
4.21 Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und ER . . 121
4.22 Ergebnisbilder: exakter Multiplizierer, hier vorgestellter approximierter Multiplizierer mit exakten 4:2-Carry-Save-Kompressoren, Ullah et al. [A 155] Design Cc (von links nach rechts) (Abbildung aus [B 4]; Original ©2021 IEEE) . 124

Tabellenverzeichnis

2.1 Entwicklung von Versorgungsspannung und V_{INS} über mehrere Technologieknoten nach [A 39] . . . 19
3.1 Übersicht über die zur Bewertung der Datenabhängigkeit des Monitorings genutzten Constrained-Random-Eingangsvektoren . . . 43
3.2 Auswertung über alle Sensorregionen und FPGAs, die zur Evaluation der LOC-Sensoren untersucht wurden . . . 74
3.3 Messwerte für Anwendungsdesign A (Leakage dominiert die Verlustleistung); Alle Werte sind Durchschnittswerte. . . . 87
3.4 Messwerte für Anwendungsdesign B (dominante dynamische Verlustleistung). Alle Werte sind Durchschnittswerte. . . . 87
3.5 Vergleich mit ausgewählten Arbeiten aus dem Stand der Technik [✓ = ja, o = mit Einschränkungen, ✗ = nein] . . . 89
4.1 Wahrheitstabelle des approximierten 4:2-Kompressors [B 4] . . . 100
4.2 Anzahl der PPB_A Ausgangsbits im Vergleich mit einer 'konventionellen' Partialproduktbildung basierend auf AND-Gattern für dieselben Multiplizierer-Eingänge [B 4] . . . 102
4.3 Anzahl der PPB_B Ausgangsbits im Vergleich mit einer 'konventionellen' Partialproduktbildung basierend auf AND-Gattern für dieselben Multiplizierer-Eingänge [B 4] . . . 102
4.4 Ressourcenbedarf zur Erzeugung der Partialprodukte eines 8x8-Bit-Multiplizierers [B 4] . . . 103
4.5 PPB_A Statistik [B 4] . . . 105
4.6 PPB_B Statistik [B 4] . . . 105
4.7 Vergleich mit dem Stand der Technik von approximierten 8-Bit-Multiplizierern. Die Ergebnisse wurden, wo nicht anders gekennzeichnet, durch Implementierung auf einer identischen Toolchain und erschöpfende Simulation ermittelt. . . . 122
4.8 Ergebnisse für die Anwendung verschiedener approximierter Multiplizierer in einem 3x3-Faltungskern [B 4] . . . 124

Abkürzungsverzeichnis

ACU	AVS Control Unit
AES	Advanced Encryption Standard
API	Application Programming Interface
ASIC	Application-specific Integrated Circuit
ATPG	Automatic Test Pattern Generation
AVFS	Adaptive Voltage and Frequency Scaling
AVS	Adaptive Voltage Scaling
BEOL	Back End of Line
BIST	Built-in Self-Test
BIT	Built-in Test
BRAM	Block Random-Access Memory
BTI	Bias Temperature Instability
CEO	Chief Executive Officer
CLB	Configurable Logic Block
CMOS	Complementary Metal–Oxide–Semiconductor
CMP	Chemical Mechanical Polishing
CPU	Central Processing Unit
CVD	Chemical Vapour Deposition
DCM	Digital Clock Manager
DIBL	Drain-induced Barrier Lowering
DNN	Deep Neural Network
DSP	Digital Signal Processing
DVFS	Dynamic Voltage and Frequency Scaling
EDA	Electronic Design Automation
ER	Error Rate
Fab	Semiconductor Fabrication Plant
FF	Flipflop
FFGB	Fast Fluctuation Guard Band
FinFET	Fin Field-Effect Transistor
FIR	Finite Impulse Response
FIT	Failure in Time
FIVR	Fully Integrated Voltage Regulator
FN Tunneling	Fowler–Nordheim Tunneling
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GB	Guard Band

GIDL	Gate Induced Drain Leakage
GND	Ground (das gemeinsame Bezugspotential in einer Schaltung)
HCI	Hot Carrier Injection
HKMG	High-k/Metal Gate
IC	Integrated Circuit
IO	Input/Output
IP Core	Intellectual Property Core
IR-Drop	Einbruch der Versorgungsspannung durch ohmsche Widerstände
LER	Line Edge Roughness
LFSR	Linear Feedback Shift Register
LOC-Sensor	Mittels `LOC...RANGE`-Constraints platzierter Sensor
LUT	Lookup Table
MAC	Multiply-Accumulate
MAE	Mean Absolute Error
MED	Mean Error Distance
ML	Machine Learning
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
MRE	Mean Relative Error
MUX	Multiplexer
NBTI	Negative-Bias Temperature Instability
NMED	Normalized Mean Error Distance
NMOS	N-type Metall-Oxid-Semiconductor
OCV	On-Chip Variation
P&R	Place and Route
PBTI	Positive-Bias Temperature Instability
PC	Personal Computer
PCB	Printed Circuit Board
PDN	Power Delivery Network
PLL	Phase-locked Loop
PMBus	Power Management Bus
PMOS	P-type Metall-Oxid-Semiconductor
PPB	Partialprodukt Block
PSNR	Peak Signal-to-Noise Ratio
PVT	Process, Voltage and Temperature
RATeMAP	Reliability Aware Task Mapping
RCA	Ripple-Carry Addierer
RE	Relative Error

RO	Ringoszillator
ROM	Read-only Memory
RTL	Register Transfer Level
SAIF	Switching Activity Interchange Format
SNR	Signal-to-Noise Ratio
SoC	System-on-a-Chip
SR Latch	Set-Reset Latch
SRAM	Static Random-Access Memory
STA	Static Timing Analysis
T-FlipFlop	Toggle-Flipflop
TDDB	Time-dependent Dielectric Breakdown
TPU	Tensor Processing Unit
$\mathbf{V_{DD}}$	Positive Versorgungsspannung
$\mathbf{V_{GS}}$	Gate-Source-Spannung
$\mathbf{V_{TH}}$	Schwellenspannung
VF-Arbeitspunkt	Spannungs-Taktfrequenz-Arbeitspunkt
VHDL	Very High Speed Integrated Circuit Hardware Description Language
VLSI	Very Large Scale Integration
VRM	Voltage Regulator Module
WCNED	Worst-Case Normalized Error Distance
WCRE	Worst-Case Relative Error
WGN	White Gaussian Noise
WNS	Worst Negative Slack

1 Einleitung

1.1 Problemstellung und Zielsetzungen der Arbeit

Über mehrere Dekaden gelang es der Halbleiterindustrie, den wachsenden Bedarf an Rechenleistung durch ein exponentielles Wachstum der auf einem Chip integrierten Transistoren zu befriedigen. Diese schon 1965 vom späteren Intel-CEO Gordon Moore vorausgesagte Entwicklung wurde als Mooresches Gesetz bekannt [A 1]. Zuletzt verlangsamte sich die Skalierung der Strukturgrößen jedoch deutlich und etliche Experten erwarten ihr nahendes Ende [A 2], [A 3], [A 4]. Wichtige Zukunftstechnologien, wie künstliche Intelligenz, autonomes Fahren sowie fortschrittliche Kommunikationssysteme wie 5G und 6G basieren jedoch auf der Verarbeitung enormer Datenmengen. Um den steigenden Bedarf an Rechenleistung trotz der langsameren Entwicklung der Halbleiter-Prozesstechnologie decken zu können, wird in zunehmenden Maße spezialisierte Hardware benötigt [A 5]. Eine Möglichkeit dafür sind Hardwarebeschleuniger. Diese können als Application-specific Integrated Circuit (ASIC) oder Field Programmable Gate Array (FPGA) realisiert werden und ermöglichen im Vergleich zu Central Processing Units (CPUs) eine um ein bis drei Größenordnungen [1.1] größere Rechenleistung und Energieeffizienz [A 2], [A 6]. Deshalb gelten sie als eine der Schlüsseltechnologien für eine weitere Steigerung der Energieeffizienz in der Informationstechnologie [A 7].

Als ASIC gefertigte, dedizierte Hardwarebeschleuniger kommen jedoch nur in Frage, wenn sowohl die Art des Workloads als auch die Art seiner Verarbeitung über die gesamte Produktlebensdauer bekannt sind. Dies ist insbesondere bei aktuellen Technologien wie Machine Learning (ML), die einer rasanten Entwicklung unterliegen, häufig nicht der Fall. Vielmehr gibt beispielsweise Microsoft als großer Cloud-Betreiber an, dass sich die Workloads im Wochen- und Monatsrhythmus ändern [A 5]. FPGAs ermöglichen sowohl die Verarbeitung großer Datenmengen in Hardware als auch die flexible Anpassung an sich ändernde Anforderungen. Folgerichtig wächst ihr Marktanteil seit ihrer Markteinführung im Jahr 1984 kontinuierlich [A 8]. So wird allein für den Zeitraum von 2021 bis 2026 ein Wachstum des Marktes um 47 % auf über neun Milliarden US-Dollar erwartet [A 9]. Ihr Anwendungsfeld reicht von Cloud-Instanzen wie beispielsweise *Amazon EC2 F1* [A 10], *IBM CAPI* und dem *Project Catapult* von Microsoft [A 11] über Anwendungen in Smart Devices bis hin zur Telekommunikationsinfrastruktur [A 9]. Den Vorteilen im Vergleich zu ASICs wie großer Flexibilität und kürzerer Time-to-Market steht jedoch eine um bis zu eine Größenordnung schlechtere Energieeffizienz gegenüber [A 12], [A 6]. Zudem sind ihre rekonfigurierbaren Grundbausteine deutlich langsamer als die Standardzellen und das fest verdrahtete Routing eines ASIC.

Insbesondere die schlechtere Energieeffizienz ist dabei als kritisch zu betrachten. Zum einen sind hier finanzielle Gründe ausschlaggebend, da der Energiebedarf einen erheblichen Teil der Kosten eines Rechenzentrums verursacht. Microsoft gibt die direkt für Elektrizität anfallenden Kosten mit 15 % der Gesamtkosten für den Betrieb eines großen Cloud-Rechenzentrums an [A 13]. Noch höhere Kosten fallen jedoch mittelbar durch die große Leistungsaufnahme an: die Infrastrukturkosten für die Transformation und Verteilung der Energie, die Sicherstellung der Zuverlässigkeit der Energieversorgung sowie die Abführung der entstehenden Abwärme sum-

[1.1] Im Rahmen dieser Arbeit ist mit „Größenordnung“ immer die dezimale Größenordnung, also die Potenz zur Basis 10, gemeint.

mieren sich auf 25 % der Gesamtkosten [A 13]. Somit verursacht der Energiebedarf insgesamt etwa 40 % der Kosten für ein modernes Rechenzentrum.

Zum anderen summiert sich der Energiebedarf der Informationsverarbeitung insgesamt auf eine Größe, die auch aus einer gesamtgesellschaftlichen Perspektive nicht vernachlässigt werden kann. So verbrauchten im Jahr 2017 allein die Rechenzentren in Deutschland 13,2 TWh Energie. Dies entspricht etwa 2,5 % des gesamten Verbrauchs an elektrischer Energie [A 14]. Der weltweite Energiebedarf aller Rechenzentren wird für das Jahr 2018 in einer eher vorsichtigen Schätzung mit etwa 200 TWh angegeben [A 7], [A 15]. Der weltweite Gesamtenergiebedarf für Informationsverarbeitung und Kommunikation hingegen lag im Jahr 2012 bei 920 TWh. Dies entspricht 4,7 % des weltweiten Stromverbrauchs [A 16]. Im Jahr 2018 wurde er bereits auf etwa 2 000 TWh geschätzt [A 15]. Gelänge es, die Energieeffizienz in diesem Bereich um nur ein Prozent zu steigern, entspricht die Ersparnis mehr als dem Anderthalbfachen des jährlichen Stromverbrauches von Berlin [A 17]. Angesichts dieser Größenordnung ist eine Steigerung der Energieeffizienz der Datenverarbeitung auch als Beitrag zum Klimaschutz von Bedeutung.

Durch das, wie beschrieben, zuletzt deutlich langsamere Complementary Metal–Oxide–Semiconductor (CMOS)-Scaling kann dabei das Problem der schlechten Energieeffizienz und geringeren Performance nicht allein durch Abwarten und die Hoffnung auf künftige Technologieknoten gelöst werden. Vielmehr werden hierzu erhebliche Anstrengungen auf weiteren Gebieten wie der Architektur und der optimalen Nutzung der zur Verfügung stehenden Prozesstechnologie benötigt [A 18], [A 3]. Die vorliegende Arbeit verschreibt sich dieser Aufgabe.

Es werden zwei mögliche Ansatzpunkte für eine energieeffizientere Datenverarbeitung behandelt. Zum einen wird die Optimierung des Spannungs-Taktfrequenz-Arbeitspunktes (VF-Arbeitspunktes) mittels Adaptive Voltage Scaling (AVS) untersucht. Dies ist eine Technik zur optimalen Ausschöpfung des Potentials einer gegebenen Prozesstechnologie, die von der konkreten Anwendung und deren Architektur unabhängig ist. Zum anderen wird Approximate Computing als eine Möglichkeit zur energiesparenden Umsetzung geeigneter, fehlertoleranter Anwendungen in ein Hardware-Design betrachtet. Konkret wird ein explizit für die Zielplattform FPGA optimierter approximierter Multiplizierer entwickelt.

Bezüglich beider Ansatzpunkte gibt es bereits sehr viele Forschungsarbeiten. Jedoch gilt für beide Gebiete, dass sich die für ASIC entwickelten Vorgehensweisen nicht unmittelbar auf FPGAs übertragen lassen. So ist beispielsweise das Einsparen von Logikgattern im Bereich des Approximate Computings auf ASICs ein übliches Optimierungsziel. FPGAs dagegen basieren nicht auf einzelnen Logikgattern, sondern bilden die Logik in Lookup Tables (LUTs) ab. Folglich werden zur Steigerung der Energieeffizienz von FPGA-basierten Anwendungen auf die Eigenheiten dieser Plattform angepasste, dedizierte Verfahren benötigt. Ebensolche werden in dieser Arbeit entwickelt.

1.2 Aufbau der Arbeit

Die vorliegende Arbeit gliedert sich in sechs Abschnitte. Zunächst werden in Kapitel 2 die für die nachfolgenden Abschnitte bedeutenden Grundlagen erläutert. Gemäß der Zielstellung dieser Arbeit werden zunächst die Bestandteile und Einflussgrößen der Verlustleistung von Very Lar-

ge Scale Integration (VLSI)-Systemen beleuchtet, um daraus mögliche Ansatzpunkte zu deren Verringerung abzuleiten. Anschließend werden VLSI-Systeme anhand ihrer zeitlichen Synchronisation kategorisiert. Für die in dieser Arbeit schwerpunktmäßig betrachteten synchron getakteten Systeme werden anschließend das Timing sowie die verschiedenen Einflussgrößen auf das Timing diskutiert. Nachfolgend werden Alterungseffekte von auf aktuellen Technologieknoten gefertigten Chips betrachtet. Schließlich werden Besonderheiten von FPGAs herausgearbeitet, die benötigt werden, um Ansätze zur Steigerung der Energieeffizienz zielgerichtet entwickeln zu können.

Das dritte Kapitel beinhaltet mehrere Beiträge für die Anwendung von Adaptive Voltage Scaling (AVS) auf FPGAs. Zunächst werden existierende Ansätze diskutiert und kategorisiert. Anschließend werden die auf Monitoring-Flipflops basierenden AVS-Systeme näher analysiert. Dabei werden Einflussgrößen, die sich auf die Zuverlässigkeit solcher Systeme auswirken, theoretisch erläutert und mittels einer geeigneten Simulation quantitativ bestimmt. Nachfolgend wird ein sensorbasierter AVS-Ansatz entwickelt. Dabei wird der Schwerpunkt darauf gelegt, drei Teilaspekte näher zu beleuchten, die in vielen Arbeiten zum Thema nur am Rande betrachtet oder komplett vernachlässigt werden. Diese sind: die notwendige Kalibrierung der Sensoren, die Bemessung eines residualen Guard Bands für Einflussgrößen, die nicht ausgeregelt werden können und eine Platzierung der Sensoren, die sich gut in bestehende Electronic Design Automation (EDA)-Arbeitsabläufe einfügt.

Das folgende Kapitel grenzt sich deutlich von den vorhergehenden Kapiteln ab. Standen bisher die vom konkreten Anwendungsdesign weitestgehend unabhängigen realen, physischen Eigenschaften der Bestandteile eines VLSI-Systems im Vordergrund, befasst sich Kapitel 4 mit der energiesparenden Umsetzung von Anwendungen in ein Hardwaredesign. Konkret wird ein approximierter Multiplizierer[1,2] als Anwendung des Approximate-Computing-Paradigmas vorgeschlagen. Dieser ermöglicht es, Anwendungen, die ein gewisses Maß an Fehlertoleranz aufweisen, sehr energieeffizient auf FPGAs zu implementieren. Der Beitrag in diesem Kapitel ist somit gewissermaßen orthogonal zu den vorhergehenden Kapiteln. Er lässt sich unabhängig von diesen einsetzen, aber auch beliebig mit ihnen kombinieren.

Kapitel 5 schließt die Arbeit mit einer Zusammenfassung und Diskussion ab und gibt einen Ausblick auf mögliche Folgearbeiten.

Abbildung 1.1 illustriert den Aufbau dieser Arbeit. Wesentliche eigene Beiträge sind gelb hervorgehoben.

[1,2] „Approximiert“ wird in dieser Arbeit als Übertragung des englischen „Approximate“ aus „Approximate Computing“ genutzt.

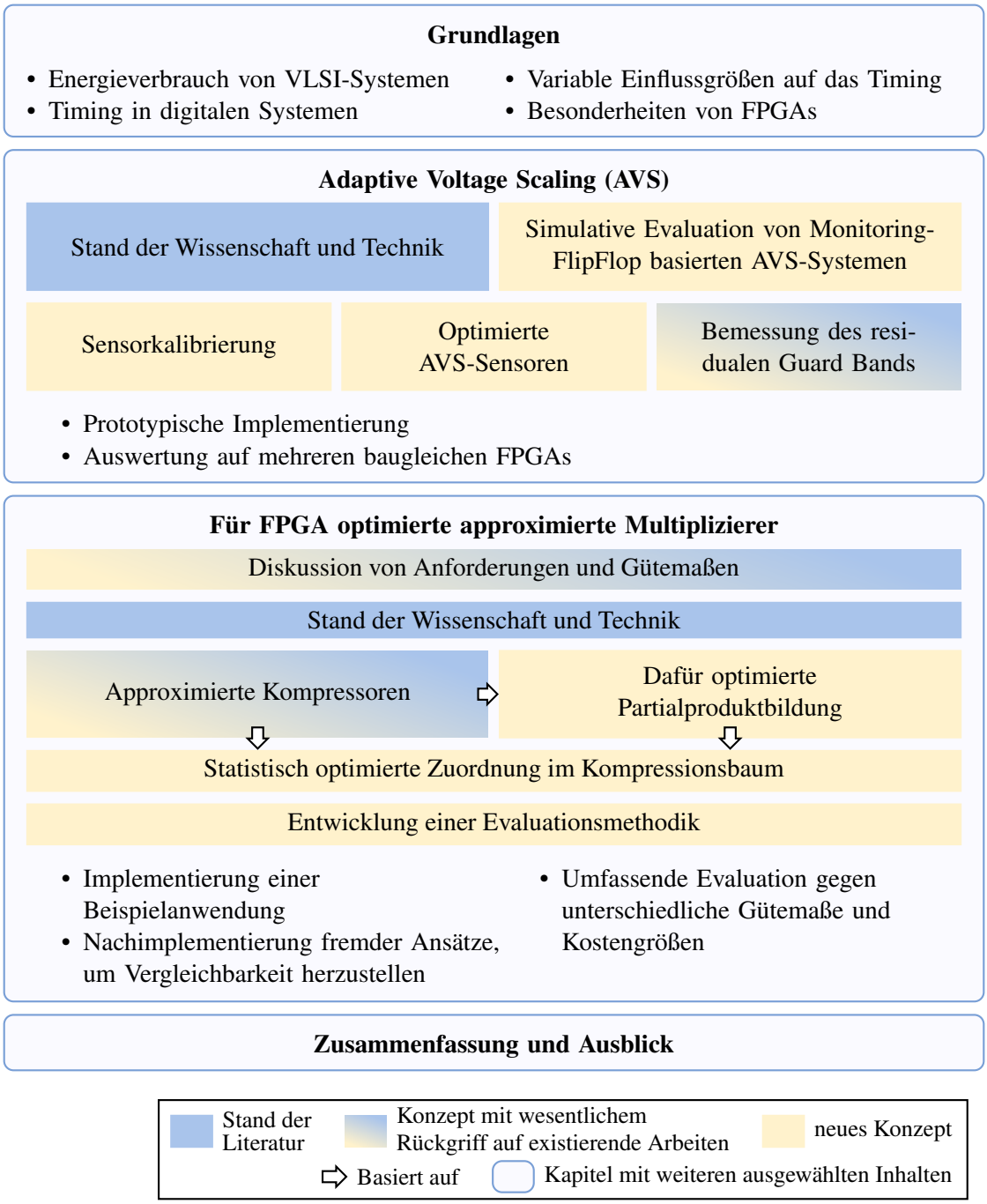


Abbildung 1.1: Struktur der vorliegenden Arbeit

1.3 Konventionen und Fachtermini

Industrie und Forschung im Bereich von Halbleitern im Allgemeinen und VLSI-Systemen im Speziellen sind hochgradig internationalisiert. So ist beispielsweise die Entwicklung und Produktion eines aktuellen System-on-a-Chip (SoC) ohne Technologie aus mehreren Ländern quasi nicht mehr vorstellbar. Als gemeinsame Fachsprache hat sich in diesem Zusammenhang die englische Sprache etabliert. Praktisch der gesamte Korpus relevanter Literatur für diese Arbeit ist in englischer Sprache erschienen. Folglich wurden und werden Zusammenhänge, Bauteile und Konzepte zunächst auf Englisch benannt. Zwar existieren bisweilen Übertragungen ins Deutsche. Diese sind jedoch häufig ungebräuchlich. Das größere Problem ist jedoch, dass die Rückübersetzung ins Englische häufig mehrdeutig ist. Um dem/der Leser:in [1.3] die Zuordnung zu bekannten und die Recherche von eventuell unbekannten Konzepten und Begriffen zu erleichtern, sowie um Mehrdeutigkeiten zu vermeiden werden in dieser Arbeit häufig englischsprachige Fachbegriffe verwendet.

Für die grafische Darstellung von Mess- und Simulationsergebnissen wurden die einzelnen gemessenen oder simulierten Punkte häufig zusätzlich durch eine Linie zu einem durchgehenden Graphen ergänzt. Dies dient der leichteren Erfassung von Trends und Entwicklungen. Die Daten sind deshalb nicht als kontinuierlich zu werten, vielmehr handelt es sich um diskrete oder durch diskrete Abtastung ermittelte Werte.

In der vorliegenden Arbeit werden mehrere Literaturverzeichnisse verwendet. Das Literaturverzeichnis wird in der jeweiligen Zitationsangabe mit einem Großbuchstaben als Präfix gekennzeichnet. Dabei steht das Präfix „A“ für externe Literatur, an der der Autor dieser Arbeit nicht beteiligt ist. Veröffentlichungen unter Beteiligung des Autors dieser Arbeit werden nochmals in die Kategorien „B“ und „C“ unterschieden. Peer-reviewed Journalartikel und Konferenzbeiträge werden mit dem Präfix „B“ gekennzeichnet. Weitere Veröffentlichungen, die eher dem Austausch dienen werden gesondert in Kategorie „C“ aufgeführt. Dabei handelt es sich beispielsweise um Beiträge zu Invited Sessions und projektinternen Tagungen. Zwar durchliefen auch diese Arbeiten einen Review-Prozess, dieser wird vom Autor der vorliegenden Arbeit jedoch nicht mit dem üblichen Prozess bei Journalen und internationalen Konferenzen gleichgesetzt. Studentische Arbeiten, die vom Autor dieser Arbeit betreut wurden, werden mit dem Präfix „D“ gekennzeichnet. Die entsprechenden Literaturverzeichnisse sind dieser Arbeit als Anhänge A bis D beigefügt.

[1.3] Im Sinne der Genderneutralität wird in dieser Arbeit, sofern keine konkrete Person bezeichnet wird, die genderneutrale Schreibweise mit Doppelpunkt und sowohl männlichem als auch weiblichem Artikel genutzt.

2 Grundlagen

Thema dieser Arbeit sind Möglichkeiten zur Steigerung der Energieeffizienz Field Programmable Gate Array (FPGA)-basierter Anwendungen. Um die Nachvollziehbarkeit zu gewährleisten und Klarheit über die referenzierten Begriffe und Konzepte herzustellen, werden in diesem Kapitel die technologischen und konzeptuellen Grundlagen dargestellt. Zu diesen zählen die Grundlagen des Energieverbrauchs in Very Large Scale Integration (VLSI)-Systemen, welche in Abschnitt 2.1 behandelt werden. Weiterhin werden in Abschnitt 2.2 VLSI-Systeme bezüglich der angewendeten Mechanismen zur Synchronisation kategorisiert und anschließend das Timing synchroner, digitaler Schaltungen näher analysiert. Das Timing unterliegt dabei vielfältigen Einflüssen, die in Abschnitt 2.3 dargestellt werden, da das in Kapitel 3 vorstellte Konzept zu Adaptive Voltage Scaling (AVS) darauf abzielt, den Umgang mit der Variabilität des Timings möglichst energieeffizient zu gestalten. Abschnitt 2.4 behandelt ausgewählte, bedeutende Alterungseffekte von VLSI-Systemen. Anschließend wird in Abschnitt 2.5 auf die für diese Arbeiten wesentlichen Eigenschaften des Clock Tree eingegangen. In Abschnitt 2.6 wird Dynamic Voltage and Frequency Scaling (DVFS) als ein Konzept, zu dem sich die vorliegende Arbeit abgrenzt, beschrieben. Die bis zu diesem Punkt behandelten Grundlagen betreffen prinzipiell sowohl FPGAs als auch Application-specific Integrated Circuits (ASICs). Abschnitt 2.7 beschreibt dagegen die für diese Arbeit wesentlichen Besonderheiten von FPGAs.

2.1 Energieverbrauch von VLSI-Systemen

Die Leistungsaufnahme eines digitalen, Complementary Metal–Oxide–Semiconductor (CMOS)-basierten VLSI-Systems setzt sich aus mehreren Komponenten zusammen. Diese werden in den folgenden Unterkapiteln dargestellt und diskutiert. Eine erste grundlegende Einteilung erfolgt in die dynamische Verlustleistung P_{dyn} und die statische Verlustleistung P_{stat}. Dabei bezeichnet die dynamische Verlustleistung alle Komponenten des Energieverbrauches, die nur im Zusammenhang mit der Schaltaktivität eines Transistors auftreten, während die statische Verlustleistung alle Komponenten erfasst, die von dieser unabhängig sind [A 19].

2.1.1 Dynamische Verlustleistung

Die dynamische Verlustleistung setzt sich aus zwei Komponenten zusammen: der Leistungsaufnahme durch kapazitives Umladen und der Leistungsaufnahme durch Kurzschlussströme während der Umschaltvorgänge [A 19], [A 20].

Leistungsaufnahme durch Umladen von Kapazitäten
Abbildung 2.1 zeigt beispielhaft die Stromflüsse zum Umladen der Lastkapazität C_L über den parasitären Widerstand R durch einen CMOS-Inverter. Wird der Eingang des Inverters von `'1'` auf `'0'` geschaltet, wird der P-type Metall-Oxid-Semiconductor (PMOS)-Transistor leitend und die Lastkapazität durch den Stromfluss $I_{E(1\rightarrow0)}$ aufgeladen. Wird der Eingang zu einem beliebigem späteren Zeitpunkt wieder auf `'1'` geschaltet, wird die Kapazität über den N-type Metall-Oxid-Semiconductor (NMOS)-Transistor durch den Strom $I_{E(0\rightarrow1)}$ gegen Masse entladen. Somit ist die Energieaufnahme für den Gesamtvorgang bestimmt durch die Ladung $Q_C = C_L \cdot V_{DD}$, die in die Lastkapazität fließt, da genau diese Ladung einmal von der positiven Versorgungsspannung

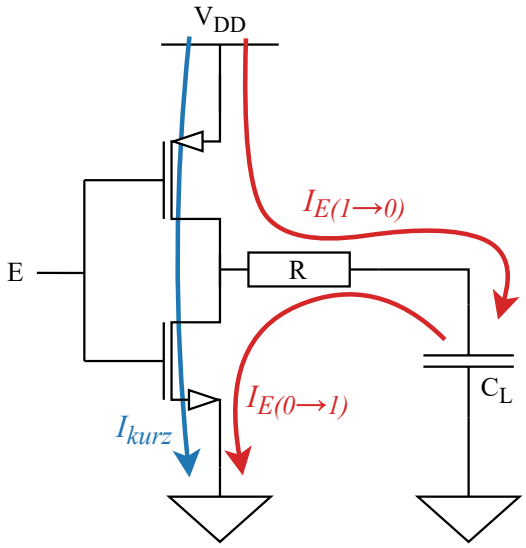


Abbildung 2.1: Durch Schaltvorgänge hervorgerufene Stromflüsse in einem CMOS-Inverter

V_{DD} nach Masse fließt. Der Energiebedarf für jedes Hin- und Herschalten eines Logikgatters ist also:

$$E = Q_C \cdot V_{DD} = C_L \cdot {V_{DD}}^2. \tag{1}$$

Dies ist insbesondere unabhängig vom parasitären Widerstand R und der Zeitdauer, innerhalb derer sich die Schaltvorgänge abspielen.

Die Leistungsaufnahme ergibt sich dann durch die Häufigkeit, mit der diese Energie in einer bestimmten Zeit T aufgenommen wird. Für synchrone Schaltungen berechnet sich diese als Produkt aus der Taktfrequenz f und der Wahrscheinlichkeit, dass innerhalb eines Taktzyklus eine Transition stattfindet: $\frac{1}{T}\sum_{t=0}^{T} Transitionen_{0\to1} = f \cdot p_{0\to1}$. Da Schaltvorgänge in beide Richtungen zwangsläufig immer abwechselnd stattfinden, genügt es dabei, die Schaltvorgänge in eine Richtung zu betrachten [A 20], [A 19]. Die Wahrscheinlichkeit für solche wird als $p_{0\to1}$ bezeichnet. Die Leistungsaufnahme durch Umladen der Kapazitäten P_{kap} ergibt sich somit als:

$$P_{kap} = C_L \cdot {V_{DD}}^2 \cdot f \cdot p_{0\to1} \tag{2}$$

In realen Umsetzungen der CMOS-Technologie werden nicht nur die Lastkapazität, sondern auch verschiedene interne, parasitäre Kapazitäten umgeladen. Diese werden hier vernachlässigt, da die prinzipielle Aussage und die Implikationen aus Formel 2 dennoch bestand haben [A 20].

Verlustleistung durch Kurzschlussströme

Im Vergleich zur Leistungsaufnahme durch das Umladen der Kapazitäten spielt die zweite Art der dynamischen Verlustleistung eine untergeordnete Rolle. Es handelt sich dabei um Kurzschlussströme, die während der Umschaltvorgänge auftreten, weil das Umschalten nicht beliebig schnell, sondern in einer endlichen Zeit abläuft. Dadurch sind unter bestimmten Randbedingungen kurzzeitig sowohl NMOS- als auch PMOS-Transistoren leitfähig und es fließt ein

Kurzschlussstrom I_{kurz} von V_{DD} nach Masse [A 19]. Die Verlustleistung P_{kurz} durch diesen Kurzschlussstrom ergibt sich als:

$$P_{kurz} = k \cdot \left(a \cdot \frac{\tau_{in}}{\tau_{out}} + b \right) \cdot {V_{DD}}^2 \cdot f \cdot p_{0\to1} \tag{3}$$

Dabei sind a und b Technologieparameter. k ist eine Funktion von V_{DD}, Schwellenspannung (V_{TH}) und der Dimensionierung des Transistors. Die Anstiegs- bzw. Abfallzeiten von Eingang und Ausgang werden mit τ_{in} und τ_{out} bezeichnet [A 19].

Die gesamte dynamische Verlustleistung ergibt sich als Summe der Teilverlustleistungen P_{kap} und P_{kurz}. Wie Gleichungen 2 und 3 zeigen, sind beide Komponenten der dynamischen Verlustleistung proportional zu $V_{DD}^2 \cdot f$. Somit gilt für die dynamische Verlustleistung insgesamt:

$$P_{dyn} \propto {V_{DD}}^2 \cdot f \tag{4}$$

2.1.2 Statische Verlustleistung

Als statische Verlustleistung werden alle Komponenten der Leistungsaufnahme bezeichnet, die unabhängig von der Schaltaktivität sind. Ein entscheidender Vorteil und Grund für die weit verbreitete Nutzung der CMOS-Technologie ist die geringe statische Verlustleistung, die im theoretischen Idealfall Null ist. Praktisch treten aber durch verschiedene Mechanismen Leckströme auf, die eine statische Verlustleistung verursachen. Diese gewinnen in neueren Prozessgenerationen gegenüber der dynamischen Verlustleistung an Bedeutung. Abbildung 2.2 illustriert die drei wesentlichen Bestandteile der statischen Verlustleistung: *Junction Leakage*, *Drain Source Leakage* und *Gate Leakage*. Die Junction Leakage ist in aktuellen Prozessgenerationen nicht mehr von Bedeutung, da sie von den anderen Leakage-Effekten um mehrere Größenordnungen übertroffen wird [A 19].

Drain Source Leakage
Das (zumindest im digitalen Bereich) wünschenswerte Verhalten eines Transistors ist, bei Unterschreiten der Schwellenspannung keinerlei Stromfluss zwischen Drain und Source zuzulassen. In realen Transistoren sinkt dieser Strom bei Unterschreiten der Schwellenspannung zwar

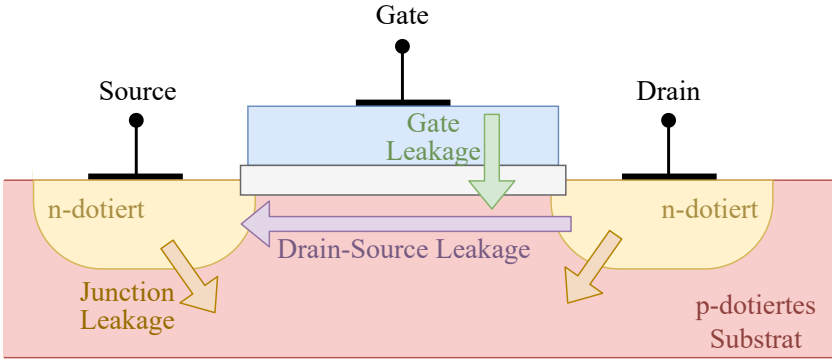


Abbildung 2.2: Komponenten der statischen Verlustleistung (Abbildung nach [A 19])

exponentiell mit der Gate-Spannung, erreicht aber niemals den Wert Null. Da die Schwellenspannung parallel zur Versorgungsspannung über die Prozessgenerationen immer weiter gesenkt wurde, unterschreitet die Gate-Spannung, die bestenfalls gleich der Source-Spannung ist[2.1], die Schwellenspannung weniger stark. Somit kommt es zu einer zunehmenden Leakage zwischen Drain und Source, die deshalb auch Subthreshold Leakage genannt wird [A 19]. Der Drain-Source-Leakage-Strom I_{DS} ist von einer Vielzahl von Einflussgrößen abhängig:

$$I_{DS} = 2\eta\mu C_{OX}\frac{W}{L}V_t{}^2 e^{\frac{V_{GS}-V_{TH}}{\eta V_t}}\left(1-e^{\frac{-V_{DS}}{V_t}}\right) \tag{5}$$

Dabei ist η der Unterschwell-Koeffizient, μ die Ladungsträgermobilität und C_{OX} die Kapazität des Gateoxids (beziehungsweise des High-k-Dielektrikums). W und L sind die Kanalbreite beziehungsweise -länge des Transistors. $V_t = \frac{kT_\vartheta}{q}$ ist die Temperaturspannung, die sich aus der Boltzmann-Konstante k, der absoluten Temperatur[2.2] T_ϑ und der Elementarladung q ergibt [A 19]. Bei Raumtemperatur nimmt sie einen Wert von etwa 25mV an [A 19]. Die bisher genannten Faktoren werden häufig zu einem Faktor $I_0 = 2\eta\mu C_{OX}\frac{W}{L}V_t{}^2$ zusammengefasst. Für den relevanten Wertebereich von V_{DS} gilt außerdem $\left(1-e^{\frac{-V_{DS}}{V_t}}\right) \approx 1$ [A 19]. V_{GS} ist die Spannung zwischen Gate und Source, V_{TH} die Schwellenspannung und V_{DS} die Darin-Source-Spannung [A 21], [A 19]. In modernen Technologien ist die Schwellenspannung V_{TH} jedoch nicht unabhängig von den anliegenden Spannungen. Neben dem Back Biasing, also dem Anlegen einer Vorspannung an den Bulk-Anschluss, wird sie auch durch weitere Effekte beeinflusst, insbesondere durch Drain-induced Barrier Lowering (DIBL). Beide Effekte beeinflussen die Schwellenspannung nichtlinear, lassen sich jedoch durch eine abgebrochene Taylor-Reihenentwicklung mit guter Genauigkeit linear approximieren [A 22], [A 19]. Dafür werden die Koeffizienten γ_d und λ_d für Back Biasing respektive DIBL eingeführt. Die tatsächliche Schwellenspannung kann dann mit der unbeeinflussten Schwellenspannung $V_{TH,0}$ durch Aufsummieren gebildet werden. Damit ergibt sich:

$$I_{DS} = I_0 \cdot e^{\frac{V_{GS}-V_{TH,0}+\lambda_d V_{DS}+\gamma_d V_{BS}}{\eta V_t}} \tag{6}$$

Für die Betrachtung der Subthreshold Leakage in Abhängigkeit von der Versorgungsspannung wird nun V_{GS} für einen NMOS im nichtleitenden Zustand als näherungsweise Null angenommen. Gleiches gilt für V_{BS}, da in der vorliegenden Arbeit kein Back Biasing angewendet wird. Damit ergibt sich:

$$P_{I_{DS}} \propto e^{\frac{\lambda_d V_{DS}-V_{TH,0}}{\eta V_t}} \tag{7}$$

Somit wächst die Verlustleistung durch die Subthreshold Leakage asymptotisch exponentiell mit der Versorgungsspannung [A 19].

Gate Leakage

Ein idealer Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET) besitzt einen (im stationären Zustand) unendlich hohen Eingangswiderstand. Mit der fortschreitenden Skalierung

[2.1] Dies gilt nur für die üblicherweise genutzte Konfiguration, in der das Gate von einem anderen Logikgatter getrieben wird und beide Gatter über eine gemeinsame Versorgungsspannung (V_{DD}) und ein gemeinsames Bezugspotential GND versorgt werden. Für spezielle Anwendungen kann die Gate-Spannung durchaus anders gewählt werden.

[2.2] Da das Symbol T bereits das Delay bezeichnet, wird die Temperatur in dieser Arbeit zur besseren Unterscheidbarkeit als T_ϑ bezeichnet. Es handelt sich dabei stets um die absolute Temperatur in Kelvin.

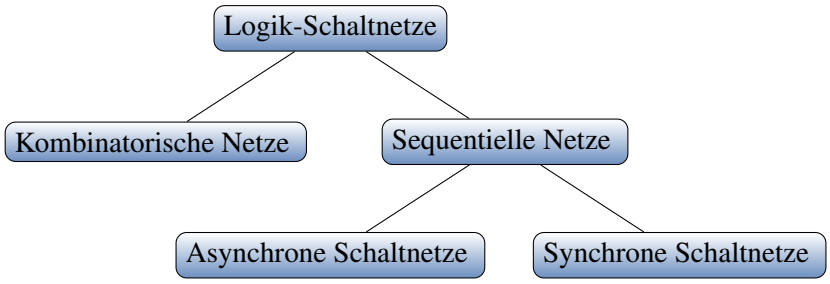


Abbildung 2.3: Einteilung von Logik-Schaltnetzen [A 23]

wurde jedoch auch die Schichtdicke des Gateoxids immer weiter verringert. Dadurch gewinnen quantenmechanische Effekte, die zu einem Leckstrom durch das Gateoxid führen, einen immer größeren Einfluss. Die beiden dafür wesentlichen Effekte sind das Fowler–Nordheim Tunneling (FN Tunneling) und das Direct Oxide Tunneling, wobei Letzteres das Verhalten dominiert. Auch die Gate Leakage ist exponentiell von der Versorgungsspannung abhängig [A 19].

2.1.3 Zusammenhang zwischen Versorgungsspannung und Verlustleistung

Zusammenfassend kann festgehalten werden, dass sich der Leistungsverbrauch aus einer von der Schaltaktivität unabhängigen statischen und einer von ihr proportional abhängigen dynamischen Komponente zusammensetzt. Beide Komponenten der Verlustleistung zeigen eine ausgeprägte Abhängigkeit von der Versorgungsspannung V_{DD}. Diese ist für die dynamische Verlustleistung quadratisch und für die statische sogar exponentiell.

2.2 Timing in digitalen VLSI-Systemen

2.2.1 Synchronität digitaler VLSI-Systeme

Die gewünschte Funktionalität digitaler Systeme wird auf Netzwerke von Logikgattern abgebildet. Diese Logiknetzwerke werden nach verschiedenen Kriterien eingeteilt. Im Rahmen der vorliegenden Arbeit wird eine aus [A 23] abgeleitete Einteilung verwendet. Diese Einteilung wird in Abbildung 2.3 dargestellt. Zunächst werden Schaltnetze grundlegend danach unterschieden, ob ihre Ausgänge lediglich von den aktuellen Eingangswerten, oder auch vom Verlauf der vorherigen Eingaben abhängig sind. Sind die Ausgangswerte nur von den aktuellen Eingangswerten abhängig, bezeichnet man das Schaltnetz als kombinatorisch, andernfalls als sequentiell. In sequentiellen Schaltnetzen hängt der Ausgabewert auch von früheren Eingangswerten ab – folglich muss ein solches Schaltnetz über Speicherelemente verfügen. Diese Speicherelemente werden häufig über ein gemeinsames Taktsignal synchronisiert. Ist dies der Fall, handelt es sich um ein synchrones Schaltnetz. Werden die Speicherelemente nicht über ein gemeinsames Taktsignal synchronisiert, handelt es sich um ein asynchrones Schaltnetz. In diesem Fall muss durch geeignete Maßnahmen sichergestellt werden, dass nur valide Daten in die Speicherelemente übernommen werden [A 24]. Der hierfür benötigte Aufwand sowie die fehlende Verfügbarkeit von geeigneten Electronic Design Automation (EDA)-Werkzeugen zum Entwurf asynchroner Schaltnetze verhindern bisher eine weitere Verbreitung asynchroner Schaltnetze im praktischen Einsatz [A 24], [A 25]. Die vorliegende Arbeit, insbesondere Kapitel 3, behandelt Problemstel-

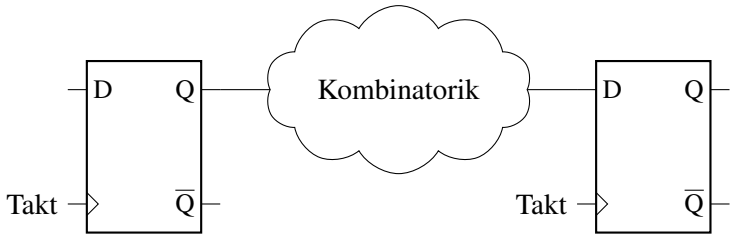


Abbildung 2.4: Aufbau eines synchronen Schaltnetzes aus kombinatorischer Logik und Speicherelementen

lungen in synchronen Schaltungen. Soweit nicht anders gekennzeichnet, ist deshalb stets von einer synchronen, digitalen Schaltung auszugehen.

Die Speicherelemente in synchronen Schaltnetzen werden von einem gemeinsamen Taktsignal synchronisiert. Dies kann durch den Pegel oder die Signalflanken des Taktsignals geschehen. Bestimmt der Signalpegel, ob das Speicherelement transparent ist oder den aktuellen Zustand speichert und am Ausgang weiterhin bereitstellt, spricht man von Latches. Wird mit jeder (typischerweise entweder steigenden oder fallenden) Signalflanke der gerade aktuelle Eingangswert gespeichert und am Ausgang bereitgestellt, spricht man von Edge-Triggered-Flipflops [A 26]. Im Rahmen dieser Arbeit bezeichnen „Flipflop“ und „Register“ auch ohne nähere Spezifikation stets ein flankengesteuertes Speicherelement. Aufgrund verschiedener Vorteile, insbesondere der weniger komplexen Timing-Analyse, sind flankengesteuerte synchrone Schaltkreise weitestgehend dominant [A 26] und stehen daher im weiteren Verlauf der Arbeit im Mittelpunkt der Betrachtung.

2.2.2 Timing in synchronen Designs

Synchrone Schaltnetze werden üblicherweise aus kombinatorischen Schaltnetzen und Speicherelementen aufgebaut. Somit ist es möglich statt eines einzigen, sehr komplexen Designs jeweils nur die zwischen zwei Registern liegende Kombinatorik zu betrachten [A 26]. Man erhält dadurch einen Aufbau, wie er in Abbildung 2.4 dargestellt ist.

In realen Schaltnetzen benötigen die Bauelemente stets eine endliche Zeit, um eine Logikoperation durchzuführen. Auch die Signallaufzeit in Leitungen ist in modernen VLSI-Designs erheblich. In synchronen Schaltnetzen muss folglich sichergestellt sein, dass die logischen Operationen der Kombinatorik zwischen zwei Registern abgeschlossen sind, bevor durch die nächste Taktflanke ein neuer Signalwert in das Speicherelement übernommen wird. Als reale Bauteile benötigen die Speicherelemente zudem eine endliche Zeit, um in Abhängigkeit vom Taktsignal ein neues Datensignal zu übernehmen und am Ausgang stabil bereitzustellen. Deshalb ist es notwendig, dass am Eingang D eines Flipflops für eine gewisse Zeit vor und nach der Taktflanke ein konstanter Signalwert anliegt [A 26]. Dies wird durch die *Setup* beziehungsweise *Hold Time Constraint* beschrieben, wie sie in Abbildung 2.5 dargestellt sind.

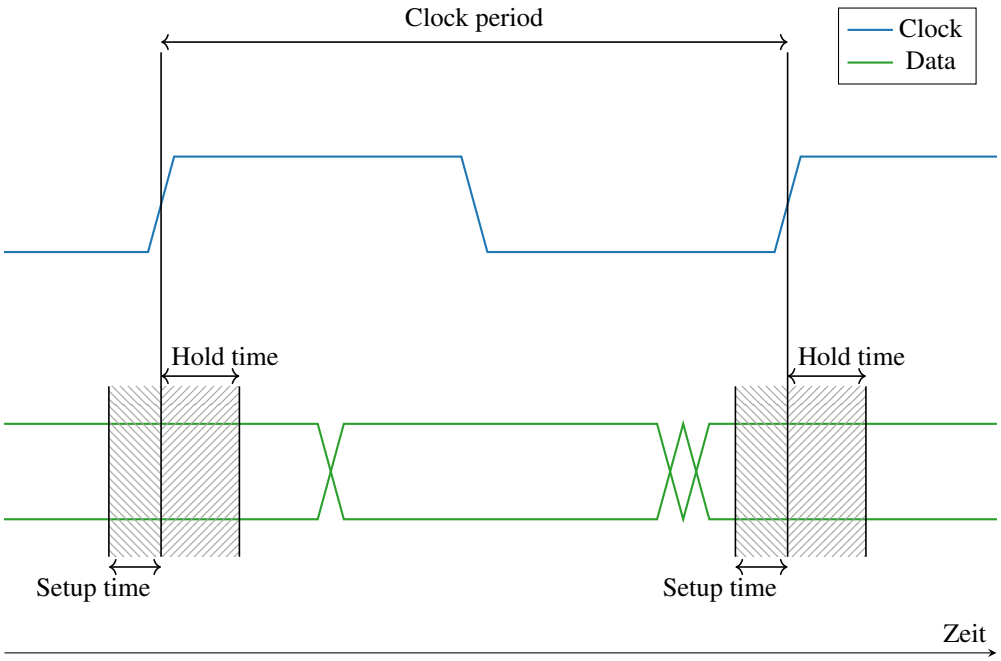


Abbildung 2.5: Setup und Hold Time (Abbildung nach [A 26])

2.2.3 Timing-Analyse

Um synchrone digitale Logik zuverlässig betreiben zu können, muss sichergestellt werden, dass die Setup und Hold Time Constraints stets eingehalten werden. In komplexen Designs ist die Untersuchung des Designs bezüglich ebendieser Fragestellung ein wichtiger Teilschritt im EDA-gestützten Systementwurf und wird als Timing-Analyse bezeichnet. Die Vielzahl bekannter Methodiken zur Timing-Analyse lässt sich in zwei Gruppen einteilen: die Dynamic Timing Analysis und die Static Timing Analysis (STA) [A 27].

Dynamic Timing Analysis
Die Dynamic Timing Analysis beruht auf einer Timing-Simulation der Netzliste des Designs. Dafür werden die Elemente der Netzliste mit dem dazugehörigen Delay annotiert. Das Design wird dann mit einer Abfolge von Stimuli simuliert, wobei für jeden Übergang zwischen zwei Stimuli geprüft wird, ob die Timing-Anforderungen erfüllt werden.

Der wichtigste Vorteil der Dynamic Timing Analysis ist, dass sie im Gegensatz zur STA nicht übermäßig pessimistisch ist. Dies liegt daran, dass eine Verletzung der Timing Constraints, die in einer Timing-Simulation festgestellt wird, stets auch tatsächlich auftreten kann. Im Gegensatz dazu werden in der STA auch sogenannte *False Paths* berücksichtigt, also Pfade, die niemals tatsächlich von einer Signalflanke durchlaufen werden können [A 26].

Der entscheidende Nachteil der Dynamic Timing Analysis ist, dass Timing-Simulationen vergleichsweise langsam sind. Daher ist die Dynamic Timing Analysis nur für sehr kleine Designs oder Teildesigns praktisch anwendbar. Für bestimmte Teilbereiche der Timing-Analyse, wie et-

wa die Analyse der Reset-Sequenz, ist die Dynamic Timing Analysis aber notwendig, da diese typischerweise nicht per STA untersucht werden können [A 27].

Static Timing Analysis
Im Gegensatz zur Dynamic Timing Analysis beruht die STA nicht auf Simulationen, sondern auf einer Darstellung des Designs als Menge gerichteter azyklischer Graphen. Dabei werden die Eingänge und Ausgänge der Logikgatter als Knoten dargestellt. Die Kanten des Graphen repräsentieren mit ihren Gewichten das jeweilige Delay. Das ist für Kanten von einem Eingangs- zu einem Ausgangsknoten das Gate Delay. Kanten, die einen Ausgang mit einem Eingang verbinden, stellen hingegen das Delay von Verbindungsleitungen dar. Zwischen zwei getakteten Elementen, also Registern oder Latches, wird nun das längste beziehungsweise kürzeste Delay durch Aufsummieren der Gewichte entlang der Pfade zwischen ihnen ermittelt [A 26]. Dabei werden jedoch auch Pfade betrachtet, die aufgrund der logischen Funktion der Gatter niemals von einer Signalkante komplett durchlaufen werden können. Dadurch ist die STA prinzipiell zu pessimistisch [A 26]. Da jedoch die Dynamic Timing Analysis für heute übliche Designgrößen zu langsam ist, wird, von Sonderfällen abgesehen, die STA genutzt, um das korrekte Timing sicherzustellen.

Als **kritischen Pfad** eines Designs bezeichnet man denjenigen Pfad zwischen zwei Speicherelementen, welcher laut STA das größte Delay aufweist [A 26]. Dieser Pfad bestimmt, mit welcher Taktfrequenz das Gesamtdesign betrieben werden kann. Dieses von der STA ermittelte Delay tritt jedoch nur real auf, wenn der Pfad tatsächlich in voller Länge von einer Signalflanke durchlaufen wird. Ist dies tatsächlich der Fall, wird es in dieser Arbeit in Anlehnung an Cortadella et al. [A 26] als „der Pfad wurde sensitiviert“ [2.3] bezeichnet. Wie im vorigen Absatz beschrieben, ist es jedoch sogar möglich, dass dieser Fall niemals eintreten kann.

2.3 Variable Einflussgrößen auf das Timing

Die in Unterabschnitt 2.2.2 beschriebenen Methoden zur Timing-Analyse basieren auf Timing-Modellen, die es ermöglichen, die Laufzeit einer Signalflanke durch ein Gatter oder eine Verbindungsleiterbahn zu berechnen. Das tatsächliche Delay ist jedoch von einer Vielzahl von Einflussfaktoren abhängig, die zur Entwurfszeit lediglich geschätzt werden können.

Wie zuvor beschrieben muss in synchronen VLSI-Systemen sichergestellt werden, dass alle Schaltvorgänge der Kombinatorik zwischen zwei Registern vor Beginn der Setup Time abgeschlossen sind. Zudem darf eine Signalflanke das Zielregister auch nicht zu früh nach der Taktflanke erreichen, da sonst die Hold Time verletzt wird. Wird keine der beiden Forderungen verletzt, ist das konkrete Delay jedoch von nachrangiger Bedeutung. Aus diesem Grund wird im Allgemeinen eine Best-Case- und eine Worst-Case-Abschätzung der zur Entwurfszeit unbekannten Einflussfaktoren auf das Delay vorgenommen. Die Worst-Case-Betrachtung wird dabei für die Setup-Time-Analyse genutzt. Da mit dieser die erreichbare Taktfrequenz des Systems bestimmt wird, ist sie für die Performance entscheidend. Die Systemauslegung anhand des Worst-Case wird auch als das Hinzufügen von *Guard Bands* beschrieben, da ausgehend vom

[2.3] Bisweilen wird in der Literatur zwischen der Sensitivierung eines Pfades und dem Erzeugen einer Signalflanke am Beginn des Pfades (die dann durch den gesamten Pfad läuft) unterschieden. Im Rahmen dieser Arbeit ist mit Sensitivierung jedoch stets gemeint, dass tatsächlich eine Signalflanke den gesamten Pfad durchläuft.

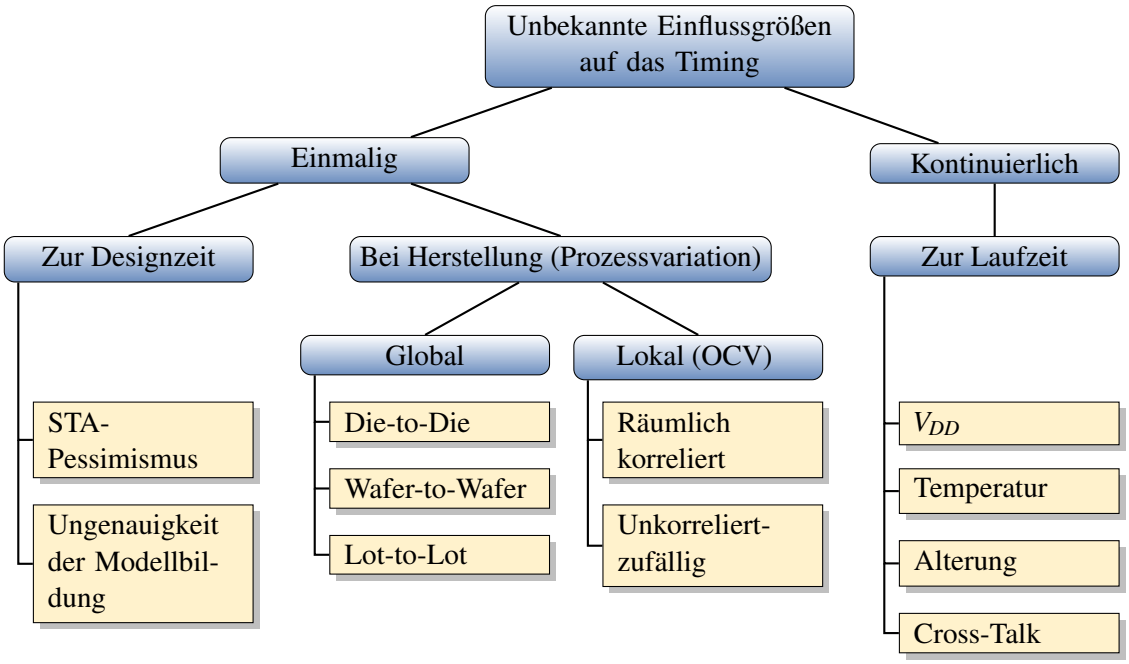


Abbildung 2.6: Zur Entwurfszeit unbekannte Einflussgrößen auf das reale Delay; „OCV“ steht für „On-Chip Variation“.

typischen Fall für verschiedene Einflüsse eben solche Guard Bands vorgesehen werden, um ein einwandfreies Funktionieren des Systems auch im ungünstigen Fall sicherzustellen. In aktuellen Prozesstechnologien ist die Variabilität der Back End of Line (BEOL)-Prozesse, also insbesondere der Leiterbahnen, für das Routing stark gestiegen und kann nicht mehr vernachlässigt werden. Dies führt dazu, dass nicht mehr trivial bestimmt werden kann, unter welchen Process, Voltage and Temperature (PVT)-Variationen tatsächlich das Worst-Case beziehungsweise Best-Case Delay auftritt. Folglich muss eine immer größere Zahl an *Design Corners* analysiert werden. Dies wird auch als *Corner Explosion* bezeichnet [A 28], [A 29].

Häufig wird von PVT-Variabilität gesprochen, da Prozessvariationen (P), Versorgungsspannung (V) und Temperatur (T) typischerweise die dominanten Einflussgrößen sind. Es gibt jedoch einerseits weitere Einflussgrößen und andererseits ist für diese Arbeit eine feinere Einteilung insbesondere der Prozessvariationen nötig. In Abbildung 2.6 wird die für diese Arbeit gewählte Kategorisierung der zur Design-Zeit unbekannten Einflussgrößen auf das Delay dargestellt. Diese werden im Folgenden näher erläutert.

2.3.1 Einfluss von Modellbildung und EDA-Tools

Bereits beim Designprozess werden bestimmte Abweichungen des Verhaltens des realen Integrated Circuits (ICs) vom angenommenen Verhalten verursacht. Dies liegt typischerweise daran, dass der Aufwand für die entsprechenden Berechnungen in einem praktikablen Rahmen gehalten werden muss. Ein Beispiel dafür ist die in Unterabschnitt 2.2.3 beschriebene mögliche Berücksichtigung von *False Paths* durch die STA. Diese wird in Kauf genommen, da eine alternative

Dynamic Timing Analysis durch eine geeignete Timing-Simulation für relevante Designgrößen nicht in annehmbarer Zeit berechenbar ist.

Ein weiteres Beispiel sind die angenommenen Delay-Modelle. Diese sind häufig stark vereinfacht. So werden häufig die realen nicht-konvexen Sachverhalte so vereinfacht, dass konvexe Optimierung angewendet werden kann [A 30]. Daraus ergibt sich zwangsläufig eine Abweichung des Verhaltens der produzierten ICs von den im Designprozess angenommenen Eigenschaften.

2.3.2 Einfluss der Prozessvariation

Mehrere Instanzen desselben, mit der gleichen CMOS-Prozesstechnologie gefertigten, VLSI-Designs sind untereinander keineswegs identisch. Vielmehr unterliegen sie in ihren Parametern einer Streuung, die durch unvermeidliche Imperfektionen des Herstellungsprozesses verursacht werden – den sogenannten Prozessvariationen.

Bei der Produktion von ICs durchlaufen die Chips eine ganze Reihe von chemischen und physikalischen Prozessschritten. Unter anderem werden die gewünschten Strukturen photolithographisch erzeugt, also auf den Wafer belichtet und anschließend chemisch durch Ätzen herausgearbeitet. Weitere wichtige Prozessschritte sind Chemical Vapour Deposition (CVD), Chemical Mechanical Polishing (CMP) und Ionenimplantation [A 31]. Jeder dieser Teilprozesse unterliegt in der Praxis einer gewissen Variabilität, die sich auf die Parameter der produzierten Strukturen auswirkt. Diese wiederum beeinflussen wichtige elektrische Eigenschaften des ICs wie beispielsweise Schaltgeschwindigkeit und Leckströme.

Um die vielfältigen Ursachen von Prozessvariationen näher zu charakterisieren, werden diese zunächst nach dem Ort ihres Auftretens unterteilt. So können Prozessvariationen beispielsweise durch bestimmte Bedingungen in einer Semiconductor Fabrication Plant (Fab) oder durch die Kalibrierung einer Fertigungsstraße entstehen. In diesem Fall beeinflussen die geänderten Bedingungen eine ganze Charge von Wafern. Man spricht deshalb von *Lot-to-Lot-Variationen*. Analog bezeichnen *Wafer-to-Wafer-Variationen* solche, die den ganzen Wafer in gleichem Maße betreffen – beispielsweise durch Ungenauigkeiten der Wafer-Platzierung unter den Belichtungsmasken. *Die-to-Die-Variationen* werden beispielsweise durch die Position des Dies auf dem Wafer verursacht. Alle bisher genannten Prozessvariationen werden als *global*, oder *Inter-Die-Variationen* bezeichnet, da sie Variationen zwischen einzelnen Dies verursachen, jedoch nicht zu Parametervariationen innerhalb eines Dies führen [A 31].

Variationen innerhalb eines Dies werden als *lokal*, *Intra-Die* oder *On-Chip Variations (OCVs)* genannt. Sie gewinnen in aktuellen Technologieknoten immer mehr an Bedeutung [A 32], [A 33]. Hauptursachen von OCVs sind Random Dopant Fluctuation, Oxide Thickness Variation und Line Edge Roughness (LER) [A 34].

Für diese Arbeit ist die Einteilung von OCVs nach ihrer räumlichen Korrelation von Bedeutung. Sie unterscheidet die OCVs in unkorreliert-zufällige Variationen und räumlich korrelierte Variationen. Hauptursache für **unkorreliert-zufällige Variationen** sind Line Edge Roughness (LER), Random Dopant Fluctuation und Metal Grain Granularity [A 35]. Beispielsweise schwanken die elektrischen Eigenschaften eines Transistors mit der Konzentration der Dotieratome. Durch die in kleineren Geometrien geringere absolute Anzahl an Dotieratomen nimmt die relative Schwan-

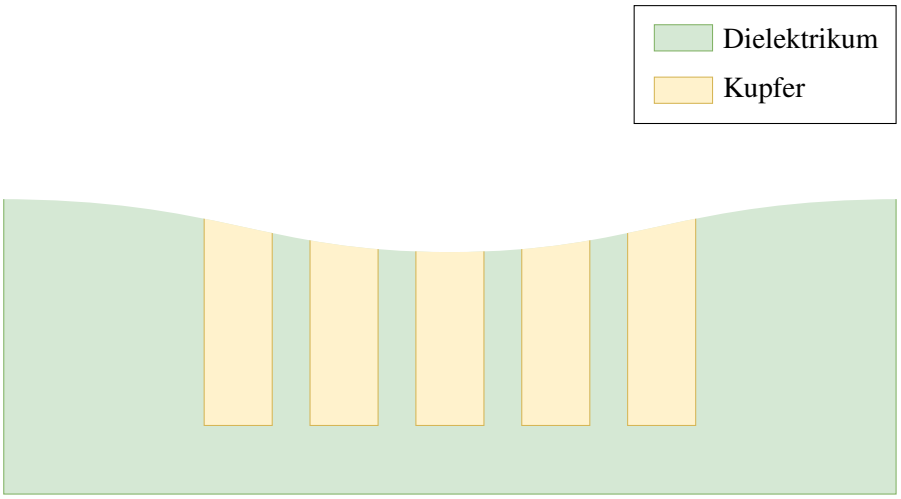


Abbildung 2.7: Schnitt durch einen Die; Durch die unterschiedlichen mechanischen Eigenschaften von Kupfer und Dielektrikum sind beim CMP räumlich korrelierte OCV entstanden. (Abbildung nach [A 31])

kung ihrer Konzentration zu. Dies führt zu einer größeren Variabilität der Schwellenspannung V_{TH} in neueren Prozesstechnologien [A 36].

Die räumlich korrelierte Variabilität, auch **systematische Variabilität** genannt, wird dagegen von solchen Einflussgrößen verursacht, die räumlich nahe Bauteile in ähnlichem Ausmaß beeinflussen. Wichtige Ursachen von systematischen OCVs sind der Einfluss der Photolithographie, des Ätzprozesses und des Chemical Mechanical Polishing (CMP).

Bezüglich der Photolithographie ist insbesondere die *Focus Position*, also der genaue Abstand zwischen der letzten Linse oder dem letzten Spiegel der Optik einerseits und des Fotolacks andererseits entscheidend. Eine Fehlpositionierung von wenigen Nanometern hat hier eine signifikante Veränderung der kritischen Dimensionen zur Folge [A 35]. Auch die genaue Intensität der Belichtung ist hier als ein wichtiger Einflussfaktor zu nennen [A 35].

CMP ist ein Prozessschritt, in dem Unebenheiten auf der Wafer-Oberfläche aus vorhergehenden Prozessschritten eingeebnet werden. Da jedoch verschiedene Materialien eine unterschiedliche Beständigkeit gegenüber dem CMP-Prozess aufweisen, kommt es in Regionen mit weicherem Material zu einer stärkeren Materialabtragung. Dies betrifft insbesondere Regionen mit einem großen Anteil an Kupfer, das im Vergleich zu den verwendeten Dielektrika deutlich weniger resistent gegenüber dem CMP-Prozess ist [A 31]. Die dabei entstehende Prozessvariabilität ist örtlich korreliert, da beispielsweise benachbarte Leiterbahnen in ähnlichem Ausmaß betroffen sind, wie in Abbildung 2.7 illustriert. Neben den hier genannten Beispielen für Ursachen von OCVs gibt es viele weitere, deren Behandlung jedoch den Rahmen dieser Arbeit übersteigen würde.

2.3.3 Einfluss der Bedingungen zur Systemlaufzeit

Nachfolgend werden die während der Systemlaufzeit dauerhaft wirkenden Einflussgrößen auf das Timing dargestellt, wobei die Alterungseffekte aufgrund des notwendigen Umfangs gesondert in Abschnitt 2.4 behandelt werden.

2.3.3.1 Versorgungsspannung

Der Einfluss der Versorgungsspannung V_{DD} auf das Timing lässt sich gut anhand des Alpha-Power-Law-Modells aufzeigen. Mit diesem lässt sich das *Propagation Delay* T_p eines kombinatorischen Pfades in Abhängigkeit verschiedener Einflussgrößen berechnen [A 19], [A 36], [A 37], [A 38].

$$T_p = k \cdot \frac{L}{W} \frac{C_{Last} V_{DD}}{(V_{DD} - V_{TH})^{\alpha}} \tag{8}$$

Dabei sind W und L die Weite und Länge des Transistor-Kanals, C_{Last} die Lastkapazität und V_{TH} die Schwellenspannung. Die Parameter α und k sind technologieabhängig. Typische Werte für planare Kurzkanal-Bulk-Transistoren sind $\alpha = 1...1,5$ [A 36]. Für einen 7 nm Fin Field-Effect Transistor (FinFET)-Prozess wird in der Literatur $\alpha = 1,25$ angegeben [A 38]. Eine Betrachtung von Zähler und Nenner der Gleichung 8 zeigt, dass die Versorgungsspannung nicht nur direkt eingeht, sondern auch die Differenz von Schwellenspannung und Versorgungsspannung entscheidend ist.

2.3.3.2 Temperatur

Die elektrischen Eigenschaften von Halbleitern und insbesondere von p-n-Übergängen sind stark temperaturabhängig [A 19]. Folglich muss eine Analyse des Delays die Temperatur berücksichtigen. Der Einfluss der Temperatur auf das Delay wird vor allem durch zwei Effekte bestimmt: zum einen der Einfluss der Temperatur auf die Ladungsträgermobilität und zum anderen ihr Einfluss auf die Schwellenspannung. Die Temperaturabhängigkeit der Ladungsträgermobilität μ berechnet sich nach:

$$\mu(T_\vartheta) = \mu_0 \cdot \left(\frac{T_\vartheta}{T_{\vartheta,0}} \right)^{\alpha_\mu} \tag{9}$$

Dabei ist T_ϑ die Temperatur [2.4] des Halbleiters, $T_{\vartheta,0}$ die nominelle Temperatur, μ_0 die Ladungsträgermobilität bei der Temperatur $T_{\vartheta,0}$ und α_μ der Mobilitäts-Temperatur-Exponent. Letzterer ist von der Prozesstechnologie und weiteren Bedingungen abhängig und wird experimentell bestimmt. Die Temperaturabhängigkeit der Schwellenspannung V_{TH} ist als

$$V_{TH}(T_\vartheta) = V_{TH,0} + \alpha_{V_{TH}} \cdot (T_\vartheta - T_{\vartheta,0}) \tag{10}$$

gegeben. Dabei ist $V_{TH,0}$ die Schwellenspannung bei Temperatur $T_{\vartheta,0}$ und $\alpha_{V_{TH}} = \frac{\partial V_{TH,0}}{\partial T_\vartheta}$ der Temperaturkoeffizient der Schwellenspannung. Dieser ist negativ. Somit gilt sowohl für die Ladungsträgermobilität als auch für die Schwellenspannung, dass sie mit steigender Temperatur ein monoton fallendes Verhalten zeigen. Dabei senkt eine sinkende Ladungsträgermobilität die Schaltgeschwindigkeit, während eine sinkende Schwellenspannung die Schaltgeschwindigkeit erhöht. Es liegen also zwei gegenläufige Einflüsse vor. Je nachdem, welcher Einfluss dominiert, steigt oder fällt das Delay in Abhängigkeit von der Temperatur.

[2.4] T_ϑ bezeichnet in dieser Arbeit stets die absolute Temperatur in Kelvin.

Tabelle 2.1: Entwicklung von Versorgungsspannung und V_{INS} über mehrere Technologieknoten nach [A 39]

Prozessknoten	Nominelle Versorgungsspannung	V_{INS}	V_{INS}/V_{DD}
90 nm	1,2 V	0,37 V	0,31
65 nm	1,1 V	0,4 V	0,36
45 nm	1,0 V	0,61 V	0,61
32 nm	0,9 V	0,69 V	0,77
22 nm	0,8 V	0,73 V	0,91

Ist die Versorgungsspannung deutlich größer als die Schwellenspannung, dominiert die Ladungsträgermobilität das Verhalten. In diesem Fall steigt das Delay mit steigender Temperatur. Für vergleichsweise große Technologieknoten sind das typische Betriebsbedingungen, weshalb dieser Zusammenhang als *Normal Temperature Dependence* bezeichnet wird. In aktuellen Technologieknoten ist die Versorgungsspannung jedoch nur noch wenig größer als die Schwellenspannung. In diesem Fall dominiert die Änderung der Schwellenspannung das Verhalten und das Delay sinkt mit steigender Temperatur. Dies wird als *Inverse Temperature Dependence* bezeichnet. Dazwischen liegt ein Bereich, in dem sich beide Effekte in etwa ausgleichen und nur eine geringe Temperaturabhängigkeit des Delay vorliegt. Die Versorgungsspannung, bei der sich die Effekte gerade ausgleichen, also $\frac{\partial Delay}{\partial T_\vartheta} = 0$ gilt, wird V_{INS} genannt [A 39]. In Tabelle 2.1 sind entsprechende Werte für mehrere Technologiegenerationen aufgeführt, die den beschriebenen Trend verdeutlichen.

2.3.3.3 Cross Talk

Signale in benachbarten Leiterbahnen eines IC sind nicht unabhängig voneinander. Vielmehr beeinflussen sich die Signale durch kapazitive Kopplung gegenseitig. Dieser Effekt wird *Cross Talk* genannt und kann unter anderem das Delay beeinflussen. Dies ist dann der Fall, wenn in benachbarten Strukturen gleichzeitig Signaltransitionen stattfinden. Verlaufen diese in entgegengesetzte Richtung, also je eine Transition von GND nach V_{DD} und eine Transition von V_{DD} nach GND, verlängert dies das Delay. Bei gleichgerichteten Transitionen wird die Signallaufzeit dagegen kürzer [A 40]. Da sich zur Design-Zeit typischerweise nicht mit der nötigen Präzision vorhersagen lässt, welche Signalflanken gleichzeitig durch welche Strukturen laufen, stellt dieser Effekt eine weitere zur Design-Zeit nicht bekannte Einflussgröße auf das Timing dar. Sie wird deshalb während der Timing-Analyse durch eine Abschätzung berücksichtigt [A 40].

2.4 Alterungseffekte

ICs unterliegen im Betrieb einer erheblichen Alterung, die in modernen Prozesstechnologien kritisch für die Zuverlässigkeit geworden ist [A 41], [A 42]. Die Alterung betrifft verschiedene Strukturen wie etwa die Transistoren, aber auch die Leiterbahnen und wird durch verschiedene Effekte hervorgerufen. *Bias Temperature Instability (BTI)* und *Hot Carrier Injection (HCI)* werden dabei als die bedeutendsten Alterungseffekte für Transistoren angesehen [A 43]. Für

Leiterzüge ist *Elektromigration* von großer Bedeutung [A 44], [A 45]. Die grundlegende Wirkungsweise und die Auswirkungen dieser Effekte werden nachfolgend dargestellt.

2.4.1 Bias Temperature Instability (BTI)

Im Gate-Dielektrikum eines MOSFET fällt die gesamte Versorgungsspannung über einer nur wenige Nanometer dicken Struktur ab. Dies führt zu sehr großen Feldstärken im Dielektrikum, durch die chemische Bindungen aufbrechen können. Durch das Aufbrechen von Si-H-Atombindungen bleiben Störstellen, sogenannte *Interface Traps*, und positive Ladungen zurück [A 46]. Diese *Trapped Charges* vermindern die Steuerwirkung des Gates auf den Kanal und erhöhen somit die Schwellenspannung [A 47]. Dadurch wiederum wird die Schaltgeschwindigkeit beeinflusst (vergleiche dazu Gleichung 8). Zusätzlich wird auch die Ladungsträgermobilität signifikant beeinträchtigt, was ebenfalls die Schaltgeschwindigkeit herabsetzt [A 48]. Vor der Einführung der High-k/Metal Gate (HKMG)-Technologie in Intels 45 nm Prozess im Jahr 2007 [A 49] betraf BTI vor allem PMOS-Transistoren, wenn diese einer negativen Gate-Source-Spannung (V_{GS}) ausgesetzt waren, weshalb der Effekt auch als Negative-Bias Temperature Instability (NBTI) bekannt ist. Im Folgenden wird zunächst das Entstehen von NBTI erklärt. Dabei lassen sich mehrere Phasen unterscheiden [A 43]:

- In der **Stressphase** verursacht die beschriebene große Feldstärke das Aufbrechen der Bindungen. Diese wird hervorgerufen, wenn eine negative Gate-Vorspannung anliegt [A 43]. Für einen CMOS-Inverter trifft dies für den PMOS-Transistor zu, wenn der Eingang auf Masse liegt [A 50]. Im, für digitale Schaltungen typischen, Extremfall gilt dann $V_{GS} = -V_{DD}$. Wie die Namensgebung bereits andeutet, beschleunigen hohe Temperaturen des Transistor-Kanals diesen Prozess [A 51].
- Zur **Ausheilungsphase** kommt es, wenn $V_{GS} = 0$ gilt. Für einen PMOS-Transistor in digitaler Logik ist dies typischerweise dann der Fall, wenn der Eingang des Gates auf V_{DD} liegt [A 46], [A 52]. Unter diesen Bedingungen wird ein Teil der Störstellen wieder durch Wasserstoff abgebunden. Dadurch verringert sich die Ladung und damit auch die Auswirkungen auf die Schwellenspannung des Transistors [A 46].

Allerdings können in der Ausheilungsphase nicht alle aufgebrochenen chemischen Bindungen wiederhergestellt werden, da ein Teil der Wasserstoff-Atome zu H_2 rekombiniert und durch Diffusion verloren geht. Dadurch bleiben die Auswirkungen der Stressphase teilweise bestehen. Diese Residuen akkumulieren sich über viele Stress-Ausheilungs-Zyklen und führen zu einer dauerhaften Verschiebung der Schwellenspannung [A 52], [A 53]. Das Ausmaß dieser langfristigen Verschiebung der Schwellenspannung ist unter anderem vom *Duty Cycle*, also dem Verhältnis der Dauer von Stress- und Ausheilungsphasen, abhängig [A 43]. Darüber hinaus spielt auch beim Prozess der Ausheilung die Temperatur eine Rolle, da der Ausheilungsprozess bei höheren Temperaturen langsamer abläuft [A 54].

Wie bereits erwähnt ist der Effekt seit Einführung des HKMG-Stacks mit umgekehrten Vorzeichen der Spannungen auch für NMOS-Transistoren relevant und wird dann als Positive-Bias Temperature Instability (PBTI) bezeichnet [A 43], [A 52]. Zusammengefasst werden beide Teileffekte als Bias Temperature Instability (BTI) bezeichnet. Bezüglich langfristiger Performanceeinbußen ist BTI einer der kritischsten Alterungseffekte [A 41], [A 42], [A 52], [A 53].

2.4.2 Hot Carrier Injection (HCI)

Hot Carrier Injection (HCI) wird, wie der Name sagt, durch *heiße Ladungsträger* hervorgerufen. Diese dringen in das Gateoxid ein, wobei die meisten von ihnen die Gate-Elektrode erreichen und dort abfließen. Ein Teil verbleibt jedoch im Gateoxid und schwächt durch seine Ladung den Einfluss des Gates auf den Kanal. Folglich steigt die Schwellenspannung des Transistors [A 43]. Da Elektronen sich deutlich schneller bewegen als Löcher, betrifft der Effekt hauptsächlich NMOS-Transistoren [A 55].

2.4.3 Elektromigration

Elektromigration ist einer der bedeutendsten Alterungseffekt der BEOL-Strukturen, in diesem Fall der Leiterbahnen [A 44], [A 45]. Fließt Strom durch einen metallischen Leiter, kommt es zu Wechselwirkungen zwischen den Ladungsträgern, also den Elektronen und den Metallionen. Bei hohen Stromdichten und begünstigt durch hohe Temperaturen kommt es dadurch zu einem Materialtransport in Richtung des Stromflusses [A 51]. Wird bezogen auf einen Abschnitt des Leiters mehr Material abtransportiert als durch den Stromfluss neu hinein transportiert wird, kann es zu einer Unterbrechung des Leiters kommen. Eine solche wird als *Void* bezeichnet. Kommt es hingegen zu einer Netto-Materialzufuhr bilden sich große Spannungen im Material, die zu einem Durchbruch durch den Isolator und somit zu einem Kurzschluss führen können [A 51].

2.5 Clock Tree

Wie in Unterabschnitt 2.2.1 dargelegt, benötigen die Speicherelemente eines synchronen Designs ein gemeinsames Taktsignal. Dieses muss an allen Speicherelementen bereitgestellt werden. Dabei erfolgt die Bereitstellung über Leiterbahnen und Buffer, also reale Komponenten, die eine endliche Signallaufzeit aufweisen und wie alle anderen Elemente auf einem IC der PVT-Variabilität unterliegen. Deshalb muss die Verteilung über eine Struktur geschehen, bei der das Taktsignal zwischen Quelle und allen Flipflops möglichst gleichartige Pfade durchläuft. Häufig wird dafür eine symmetrische Baumstruktur gewählt, deren Blätter die Flipflops sind. Deshalb wird auch vom *Clock Tree* gesprochen. Zwar werden, vor allem für die globale Verteilung des Taktsignals, bisweilen auch andere Strukturen, etwa redundante Bäume, Meshes und Spines, verwendet [A 56]. Da die nachfolgenden Überlegungen jedoch prinzipiell auch für diese Architekturen zutreffen und für die lokale Verteilung des Taktsignals typischerweise symmetrische Bäume verwendet werden [A 56], wird nachfolgend von einem symmetrischen Baum zur Verteilung des Taktsignals ausgegangen.

Aufgrund des Aufbaus des Clock Trees aus realen Bauteilen kann ein Taktsignal in der Praxis nie perfekt sein. Die wichtigsten Charakteristika eines realen Taktsignals sind *Skew*, *Jitter* und *Slew Rate* [A 57], [A 56], wobei für diese Arbeit Skew und Jitter von Bedeutung sind. Die Abweichung des Taktsignals zu verschiedenen Zeiten an ein und demselben Ort bezeichnet man als **Clock Jitter**. Die Abweichung, die zur gleichen Zeit an zwei verschiedenen Orten auftritt, wird hingegen als **Clock Skew** bezeichnet [A 58]. Insbesondere in den frühen Phasen des Design-Prozesses wird das Taktsignal häufig als ideal angenommen oder es werden lediglich Skew und Jitter durch ein Guard Band berücksichtigt. Dies sinnvolle Vereinfachung ist möglich, da die

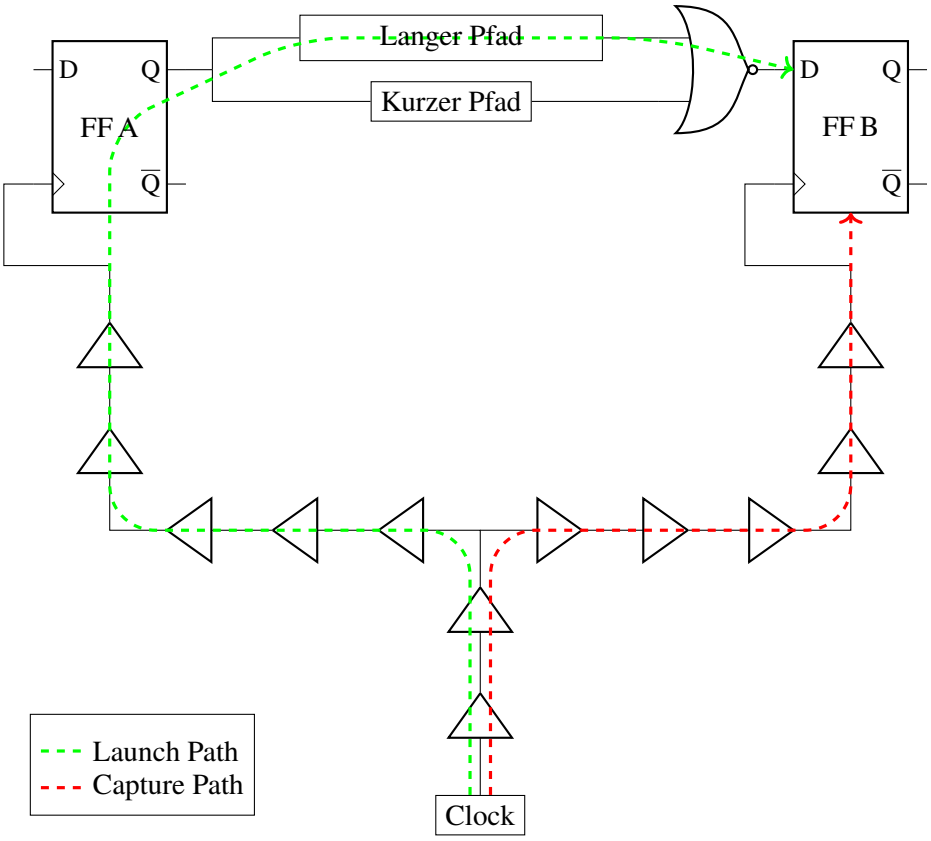


Abbildung 2.8: Launch und Capture Path (Abbildung nach [A 26])

zur Verteilung des Taktsignals notwendigen Komponenten in der Clock-Tree-Synthese durch EDA-Tools automatisiert generiert werden.

Da jedoch die lokale Variabilität durch OCVs, lokale Voltage Drops, Temperaturgradienten und verschieden starke Alterung mit fortschreitenden Prozessgenerationen immer schwerwiegender wird, muss auch deren Einfluss auf den Clock Tree berücksichtigt werden [A 26]. Zu diesem Zweck muss zunächst die Timing-Analyse, wie sie nach der Clock-Tree-Synthese durchgeführt wird, betrachtet werden. Dabei werden, wie in Abbildung 2.8 illustriert, ausgehend von der Phase-locked Loop (PLL) die kompletten Pfade zum Zielregister betrachtet, wobei das Delay des *Launch Path* ($T_{Launch\ Path}$) höchstens um eine Taktperiode (T_{Takt}) länger als das Delay des *Capture Path* ($T_{Capture\ Path}$) abzüglich der Setup Time (T_{Setup}) des Flipflops B sein darf [A 26]:

$$T_{Launch\ Path} < T_{Capture\ Path} + T_{Takt} - T_{Setup} \tag{11}$$

Der Clock Tree ist dabei so aufgebaut, dass die Pfade von der PLL zu den Registern stets das gleiche Delay aufweisen. Globale Variabilität wirkt also auf den Clock Tree in Launch und Capture Path in gleichem Maße und führt dadurch nicht zu einem Versatz der Taktflanken in Flipflop A beziehungsweise B. Im Gegensatz dazu kann lokale Variabilität das Delay der beiden

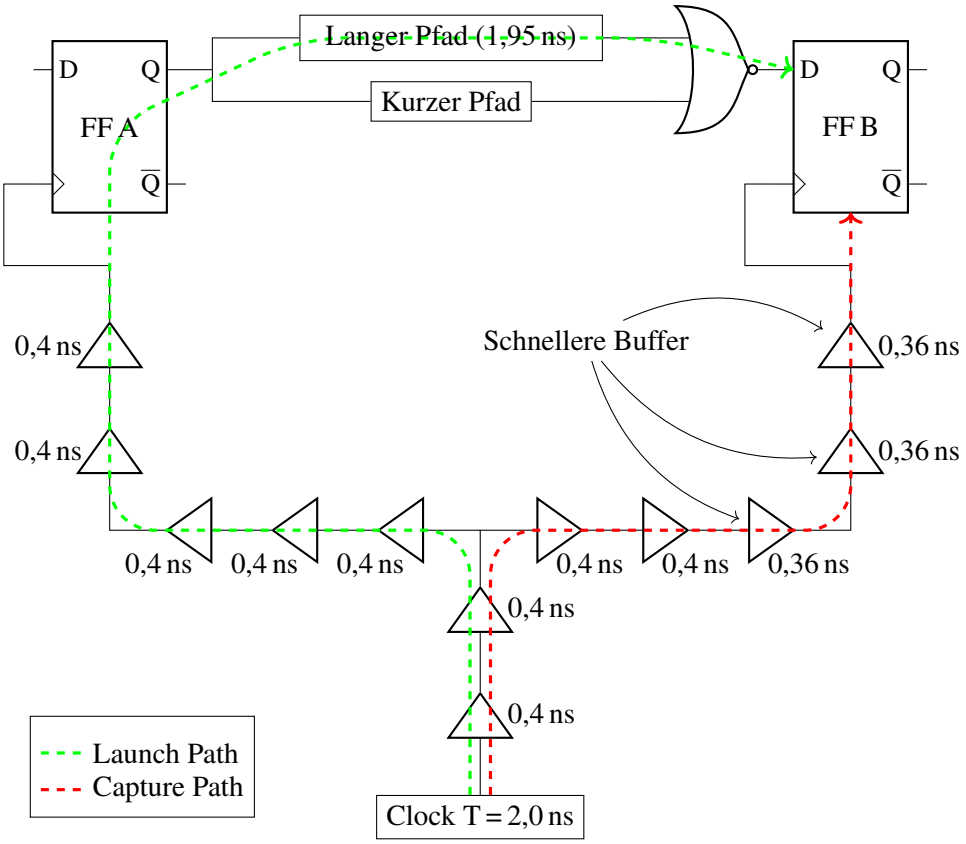


Abbildung 2.9: Launch und Capture Path unter dem Einfluss lokaler Variabilität

Pfade verschieden stark oder sogar in gegensätzlicher Richtung beeinflussen [A 56]. Dieser Fall wird in Abbildung 2.9 illustriert. Ohne lokale Variabilität gilt:

$$T_{Launch\ Path} = \underbrace{7 \cdot 0,4\,ns}_{Clock\ Tree} + \underbrace{1,95\,ns}_{Kombinatorik} = 4,75\,ns \tag{12}$$

$$T_{Capture\ Path} + T_{Takt} - T_{Setup} = \underbrace{7 \cdot 0,4\,ns}_{Clock\ Tree} + \underbrace{2,0\,ns}_{T_{Takt}} - \underbrace{0,04\,ns}_{T_{Setup}} = 4,76\,ns \tag{13}$$

Somit ist die Bedingung aus Ungleichung 11 erfüllt. In diesem Beispiel sind jedoch durch lokale Variabilität drei Buffer des Clock Tree etwas schneller als die anderen Buffer. Da die drei Buffer jedoch im Capture Path liegen, können sie zu einer Verletzung der Setup Time Constraint führen. Im Beispiel ergibt sich:

$$T_{Capture\ Path} + T_{Takt} - T_{Setup} = \underbrace{4 \cdot 0,4\,ns + 3 \cdot 0,36\,ns}_{Clock\ Tree} + \underbrace{2,0\,ns}_{T_{Takt}} - \underbrace{0,04\,ns}_{T_{Setup}} = 4,64\,ns \tag{14}$$

Damit ist in diesem Beispiel das Delay des Launch Path größer als das des Capture Path zuzüglich der Taktperiode abzüglich der Setup Time. Ungleichung 11 ist verletzt und eine korrekte Übernahme der Daten in Flipflop B kann nicht gewährleistet werden.

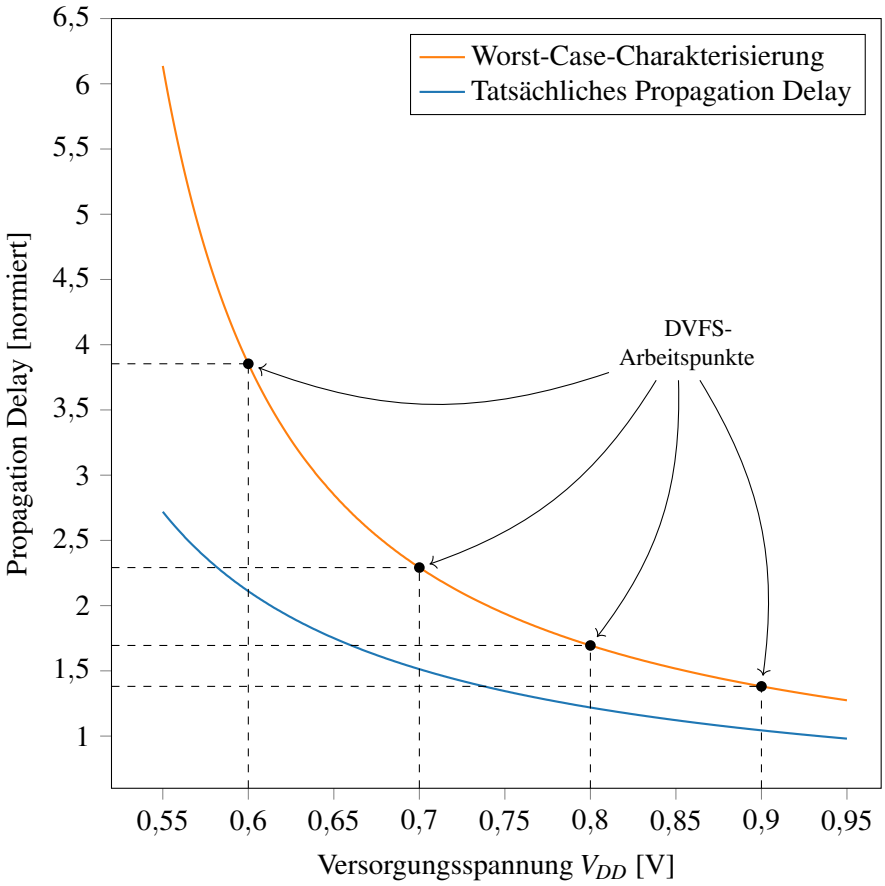


Abbildung 2.10: Spannungs-Taktfrequenz-Arbeitspunkte bei angewandtem Dynamic Voltage and Frequency Scaling (DVFS); 7nm FinFET Technologie (Prozessdaten aus [A 38], [A 59])

Für fortgeschrittene Technologieknoten werden die lokalen und zufälligen Komponenten von Clock Skew und Jitter immer bedeutender [A 36]. Deshalb muss die lokale Variabilität des Clock Tree in der Timing-Analyse berücksichtigt werden. Dies geschieht durch sogenannte *Derates*. Dabei wird beispielsweise für die Setup-Time-Analyse die Worst-Case Corner Library genutzt. Von den Ergebnissen ausgehend wird für den Capture Path ein geringeres Delay berechnet, indem das ursprüngliche Delay um einen von der Foundry vorgegebenen Prozentsatz gekürzt wird [A 26]. Es wird also ein zusätzliches Guard Band angewendet.

2.6 Dynamic Voltage and Frequency Scaling (DVFS)

Wie in Unterabschnitt 2.3.3.1 dargelegt, besteht ein monoton steigender Zusammenhang zwischen der Versorgungsspannung und der Schaltgeschwindigkeit von CMOS-basierten Schaltnetzen. Somit ist die höchstmögliche Taktfrequenz synchroner Designs direkt von der gewählten Versorgungsspannung abhängig. Die Taktfrequenz wiederum bestimmt maßgeblich den erreichbaren Durchsatz und somit die Leistungsfähigkeit des Systems. Allerdings steigt mit steigender Versorgungsspannung nicht nur die erreichbare Taktfrequenz, sondern auch die Leistungsauf-

nahme. Wie in Unterabschnitt 2.1.3 herausgearbeitet, steigt die Leistungsaufnahme dabei überproportional: für die dynamische Verlustleistung gilt ein quadratischer und für die Leakage sogar ein exponentieller Zusammenhang. Die Wahl des Spannungs-Taktfrequenz-Arbeitspunktes (VF-Arbeitspunktes) bestimmt also einerseits die Leistungsfähigkeit und andererseits den Energiebedarf des Systems. Allerdings ist das Anforderungsprofil vieler Designs nicht konstant. Vielmehr wechseln sich Spitzenlasten mit hohem Leistungsbedarf und Phasen im Teillastbereich ab. Ein System, dessen Arbeitspunkt für die geforderte Leistungsfähigkeit bei Spitzenlasten gewählt wurde, verbraucht somit im Teillastbereich überproportional viel Energie.

Dynamic Voltage and Frequency Scaling (DVFS) ist ein Ansatz, um diesen Gegensatz aufzuheben [A 60]. Dazu wird das System für mehrere Versorgungsspannungen charakterisiert. Die charakterisierten Versorgungsspannungen mit den zugehörigen erreichbaren Taktfrequenzen werden dann in einer Lookup Table (LUT) hinterlegt [A 58]. Je nach angeforderter Leistungsfähigkeit des Systems wird dann ein geeigneter VF-Arbeitspunkt gewählt. Abbildung 2.10 illustriert beispielhaft die Arbeitspunkte eines DVFS-Systems. Der Zusammenhang zwischen Spannung und Delay wurde dabei nach dem Alpha Power Law (vergleiche dazu Gleichung 8 in Unterabschnitt 2.3.3.1) für eine 7 nm Technologie modelliert. Man beachte, dass die VF-Arbeitspunkte mittels klassischer Worst-Case-Charakterisierung bestimmt werden. DVFS ändert also nichts an der pessimistischen Bestimmung der VF-Arbeitspunkte. Darüber hinaus zeigt die Grafik zwei Effekte, die bei Annäherung der Versorgungsspannung an die Schwellenspannung auftreten: Zum einen steigt das Delay bei weiterer Spannungssenkung überproportional. Zum anderen steigt mit der Annäherung von Versorgungs- und Schwellenspannung die Empfindlichkeit gegenüber PVT-Variationen, wodurch für die VF-Arbeitspunkte mit niedriger Versorgungsspannung größere PVT Guard Bands vorgesehen werden müssen [A 31].

2.7 Besonderheiten von Field Programmable Gate Arrays (FPGAs)

Die vorliegende Arbeit zielt auf die Steigerung der Energieeffizenz FPGA-basierter Anwendungen ab. In diesem Kapitel werden die grundlegende Funktionsweise sowie die Besonderheiten von FPGAs insbesondere im Vergleich zu ASICs dargestellt.

2.7.1 Architektur

Abbildung 2.11 illustriert die prinzipielle Architektur eines FPGAs als regelmäßige, sich wiederholende Anordnung dreier Basisbausteine: *Logic Blocks*, *Connection Blocks* und *Switch Blocks* [A 61]. In aktuellen FPGA-Architekturen werden je nach Hersteller und Modellreihe noch weitere Blöcke, etwa Input/Output (IO)-Blöcke, Digital Signal Processing (DSP)-Kerne und Analogkomponenten, verbaut. Diese sind jedoch nur beschränkt rekonfigurierbar und gleichen in ihren Eigenschaften weitestgehend ASIC-Designs. Im Rahmen dieser Arbeit steht jedoch die konfigurierbare Logik als Besonderheit von FPGAs im Vordergrund.

Die **Logic Blocks** sind gewissermaßen das Herzstück der FPGA-Logik. Sie bestehen aus mehreren Lookup Tables (LUTs) und weiterer Logik, wobei das Anwendungsdesign primär auf die LUTs abgebildet wird. Diese nehmen die Funktion ein, die in ASIC-Technologie den Logikgattern zufällt. Dabei kann eine n-Input-LUT eine beliebige boolesche Funktion von n Eingangssignalen abbilden. Die LUTs werden typischerweise aus 6-Transistor Static Random-Access

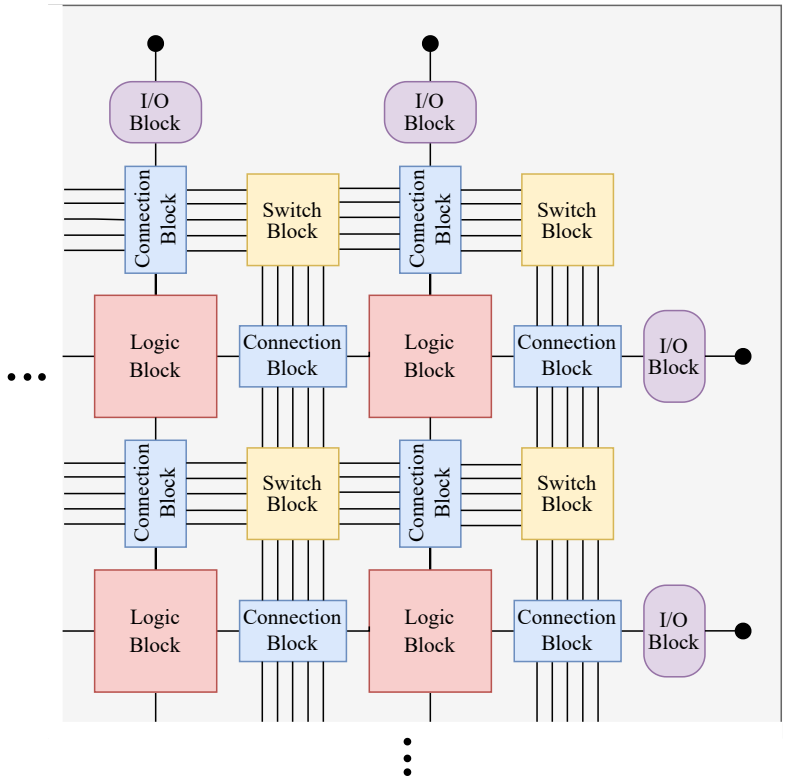


Abbildung 2.11: Prinzipieller Aufbau eines FPGAs aus Logic Blocks, Connection Blocks und Switch Blocks (Abbildung nach [A 61])

Memory (SRAM)-Zellen und Multiplexern aufgebaut. Dabei werden zur Realisierung einer n-Input-LUT 2^n SRAM-Zellen und ein entsprechender Multiplexer mit 2^n Eingängen benötigt [A 62], [A 63].

Connection Blocks dienen der Anbindung der Logic Blocks an die Routing-Infrastruktur. Dabei werden, je nach Konfiguration des FPGAs, die Ausgangssignale der Logic Blocks jeweils mit einer bestimmten Leiterbahn verbunden.

Die **Switch Blocks** wiederum verbinden, je nach Konfiguration, bestimmte Leiterbahnen miteinander. Sowohl Connection Blocks als auch Switch Blocks werden dabei durch Multiplexer realisiert, deren Steuersignale durch SRAM-Zellen konfiguriert werden können.

2.7.2 Technische Umsetzung auf Transistorlevel

Multiplexer und SRAM-Zellen sind also die dominierenden CMOS-Strukturen eines FPGAs [A 64]. Da in dieser Arbeit Leistungsaufnahme und Schaltgeschwindigkeit im Mittelpunkt der Betrachtung stehen, ist die Realisierung auf Transistorlevel von größtem Interesse. Die LUT-Multiplexer werden typischerweise mit Transmission-Gates oder NMOS-Pass-Transistoren realisiert [A 61], [A 64], [A 65]. Der Großteil der Multiplexer zur Realisierung des Routings wird herstellerunabhängig ebenfalls auf Basis von NMOS-Pass-Transistoren realisiert

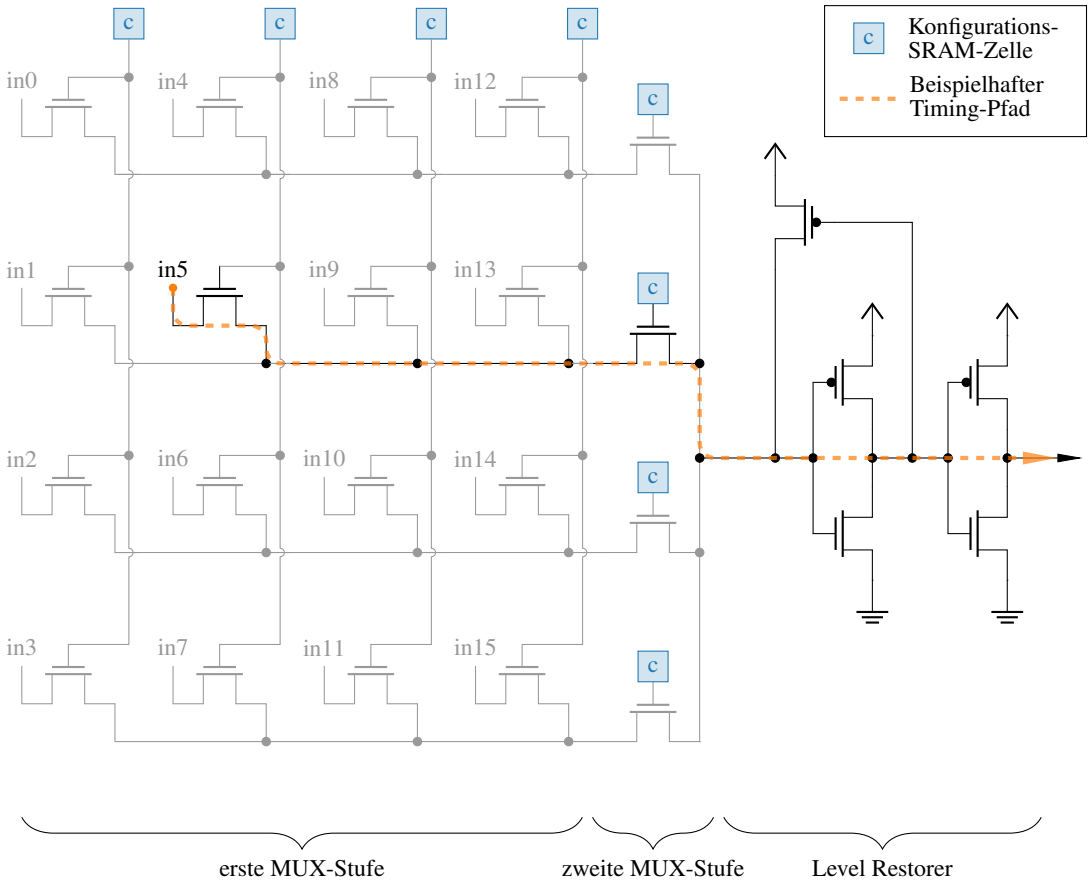


Abbildung 2.12: 16:1-Multiplexer (MUX) auf Basis von NMOS-Pass-Transistoren, wie er in den FPGAs aus Intels Stratix-10 Serie verwendet wird [A 67]

[A 64], [A 65], [A 66], [A 67]. Auch Architekturen, die für sich in Anspruch nehmen, nicht das klassische Pass-Transistor-Routing zu verwenden, beinhalten, soweit sie publiziert sind, weiterhin Pass-Transistoren an prominenter Stelle. Dies gilt beispielsweise für die von Altera publizierten *Direct Drive Multiplexer*, welche aus einem Pass-Transistor-basierten Multiplexer mit nachgeschaltetem Buffer bestehen [A 68]. Somit sind NMOS-Pass-Transistoren eine für die Eigenschaften des FPGAs entscheidende Schlüsselkomponente.

Das Verhalten eines NMOS-Pass-Transistors unterscheidet sich dabei erheblich von dem eines ansonsten gleichen Transistors, etwa in einem Inverter [A 69]. Pass-Transistoren haben je nach Richtung der Signal-Transition die Eigenschaft, das Signal deutlich abzuschwächen. Für den Fall eines NMOS-Pass-Transistors gilt, dass eine Transition des Signals in den Zustand `'0'`, also Ground, gut übertragen wird. Im Gegensatz dazu kann als maximale Ausgangsspannung $V_{OUT,max}$ jedoch nicht der Pegel der Versorgungsspannung erreicht werden, sondern maximal der Wert der Versorgungsspannung abzüglich der Schwellenspannung [A 64]:

$$V_{OUT,max} \leq V_{DD} - V_{TH} \tag{15}$$

Obwohl andere Multiplexer-Realisierungen, etwa auf Basis von Logikgattern oder Transmission Gates, diesen Nachteil nicht haben, werden die Pass-Transistoren weiterhin dominierend eingesetzt, da sie erheblich weniger Chipfläche benötigen. Zudem weisen sie eine geringere parasitäre Kapazität auf, was sich günstig auf Timing und Energieverbrauch auswirkt [A 22], [A 63], [A 64].

Häufig wird nach zwei NMOS-Pass-Transistoren ein *Level Restorer* eingefügt [A 70], da das Signal nach zwei Pass-Transistoren deutlich abgeschwächt ist [A 67] und das Delay quadratisch mit der Anzahl der verwendeten Pass-Transistoren ansteigt [A 64]. Abbildung 2.12 illustriert dies am Beispiel eines 16:1-Multiplexers, wie er in den FPGAs der Stratix-10-Serie genutzt wird [A 67]. Genauer handelt es sich um einen Driver-Input-MUX, welcher einerseits in Switch Blocks genutzt wird, um horizontale mit vertikalen Routingsträngen zu verbinden und anderseits auch in Connection Blocks die Anbindung der Logik-Block-Ausgänge an die Routing-Infrastruktur realisiert. Der Level Restorer wird von Intel als Kombination aus Buffer und Treiber bezeichnet, ist jedoch prinzipiell identisch mit Level Restorern anderer Hersteller. Level Restorer werden üblicherweise aus zwei Invertern aufgebaut, wobei ein zusätzlicher Pull-up-PMOS-Transistor im Rückkopplungspfad sicherstellt, dass der durch die Pass-Transistoren verminderte Signalpegel wieder auf die Versorgungsspannung angehoben wird.

Durch die beschriebenen Charakteristika von FPGAs ist das konfigurierbare Routing typischerweise für mehr als 50 % des Delays des kritischen Pfades sowie mehr als 50 % der Chipfläche verantwortlich [A 63], [A 64], [A 70], wobei sich die Angabe zur Chipfläche auf den Anteil der programmierbaren Logik bezieht. Der je nach Größe und Modell des FPGAs variierende Anteil sonstiger Logik, wie etwa IOs, Prozessorkerne, PLLs und weitere, wird hier nicht berücksichtigt, da der Fokus auf der programmierbaren Logik liegt und diese für die Performance des Gesamtsystems entscheidend ist [A 63].

3 Adaptive Voltage Scaling (AVS)

3.1 Motivation

Im Entwurfsprozess digitaler Systeme wird die zugrunde liegende Hardware üblicherweise als fehlerfrei arbeitend angenommen [3.1]. Allerdings basieren Very Large Scale Integration (VLSI)-Systeme auf analogen Schaltkreisen, deren Verhalten erheblichen Schwankungen unterliegt. Die Ursachen für diese Schwankungen sind die Parameterstreuung während der Produktion, der Einfluss der Temperatur, die Alterung, das Rauschen sowie die Parametertoleranz der Versorgungsspannung und weitere Einflüsse. Um dennoch ein wohldefiniertes digitales Schaltverhalten zu erzielen, werden großzügige Toleranzbereiche, sogenannte Guard Bands, vorgesehen [A 36], [A 71]. Ein VLSI-Design wird also nicht auf Basis seines tatsächlichen oder typischen Verhaltens erstellt, sondern ausgehend von einer extrem pessimistischen Abschätzung seiner Eigenschaften.

Ein wichtiger Anwendungsfall solcher Guard Bands ist die Wahl des Spannungs-Taktfrequenz-Arbeitspunktes (VF-Arbeitspunktes). Der theoretisch optimale VF-Arbeitspunkt ist zeitlich veränderlich, da er von einer Reihe von Einflussgrößen zur Laufzeit abhängig ist, die zur Designzeit prinzipbedingt unbekannt sind. Dazu zählen insbesondere die Temperatur, das Versorgungsspannungsrauschen und die Alterung. Für die genannten Einflussgrößen ist zudem nicht nur ein globaler Wert entscheidend. Auch die lokalen Schwankungen über die Fläche des Chips können nicht vernachlässigt werden [A 32], [A 33]. Die (lokale und globale) Prozessvariation während der Produktion ist zwar ein einmaliges Ereignis, hat jedoch dynamische Folgeeffekte, da sie beispielsweise die Alterung des Chips und die Empfindlichkeit gegenüber Versorgungsspannungsrauschen maßgeblich beeinflusst. Folglich können die Auswirkungen über einen einmaligen Ausmessungs- und Selektionsvorgang (dem sogenannten *Frequency Binning* [A 72], [A 73]) nur teilweise kompensiert werden. Aufgrund der genannten Unsicherheiten muss bei einer Wahl des VF-Arbeitspunktes zur Designzeit von extrem ungünstigen Annahmen ausgegangen werden, um die geforderte Zuverlässigkeit dennoch zu erreichen. Dadurch wird jedoch in nahezu jedem Fall die Leistungsfähigkeit des Systems nicht optimal genutzt [A 74], wobei eine verminderte Leistungsfähigkeit in diesem Zusammenhang sowohl einen geringeren Datendurchsatz, eine höhere Latenz, eine geringere Zuverlässigkeit als insbesondere auch einen höheren Energiebedarf bedeuten kann.

Wie in Abschnitt 2.6 ausgeführt, gilt diese Feststellung auch für Systeme, die Dynamic Voltage and Frequency Scaling (DVFS) anwenden, da hierbei zur Laufzeit lediglich eine Auswahl aus einer Reihe vordefinierter VF-Arbeitspunkte erfolgt, die zur Designzeit bestimmt wurden. Methoden, die Laufzeitwissen nutzen, um günstigere VF-Arbeitspunkte zu finden, werden dagegen als Adaptive Voltage Scaling (AVS) bezeichnet. Abbildung 3.1 illustriert diesen Unterschied zwischen Adaptive Voltage Scaling (AVS) und DVFS. Im Unterschied zu DVFS zielt AVS darauf ab, durch die Auswertung von *zur Laufzeit* ermittelten Daten die Guard Bands zu verkleinern. Die beiden Energiespartechnologien sind also orthogonal, lassen sich jedoch gut kombinieren, wobei durch die gemeinsame Nutzung von Infrastruktur zur Manipulation von Taktfrequenz und Versorgungsspannung Synergien erzielt werden können.

[3.1] Dies ist der Regelfall, es gibt jedoch Ausnahmen: etwa sicherheitskritische Systeme, Speicher, der mit Check-Summen geprüft wird und weitere

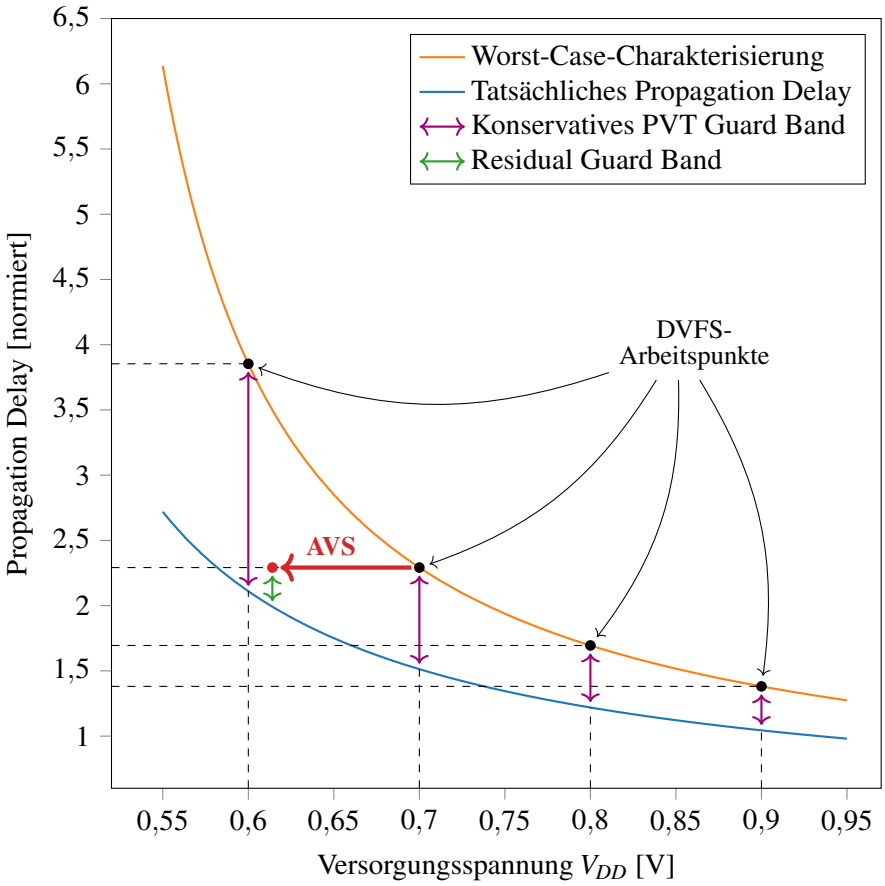


Abbildung 3.1: Vergleichende Darstellung von DVFS und AVS

Prinzipiell kann eine Anpassung des VF-Arbeitspunkts auch durch die Wahl einer höheren Frequenz bei konstanter Versorgungsspannung realisiert werden. Auch die dynamische Anpassung beider Parameter ist möglich. Dieser Fall wird gelegentlich als Adaptive Voltage and Frequency Scaling (AVFS) bezeichnet, kann aber auch als eine Kombination von DVFS und AVS aufgefasst werden. Da die Zielstellung der vorliegenden Arbeit eine Senkung der Verlustleistung ist, wird hier ausschließlich ein Absenken der Versorgungsspannung bei gleichbleibender Taktfrequenz angestrebt.

Die Bedingungen für eine Anwendung von AVS auf Field Programmable Gate Arrays (FPGAs) unterscheidet sich erheblich von denen bei Application-specific Integrated Circuits (ASICs). Die Gründe dafür sind:

- Sensoren, die nicht auf Standard-Logikgattern, sondern auf dedizierten Transistorlevel-Designs basieren, lassen sich im Allgemeinen nicht auf die Logikressourcen eines FPGA abbilden.
- Die Besonderheiten der Hardwarerealisierungen von FPGAs, wie sie in Abschnitt 2.7 beschrieben werden, müssen berücksichtigt werden. Beispielsweise muss bei einer Senkung

der Versorgungsspannung stets sichergestellt sein, dass der Inhalt der Static Random-Access Memory (SRAM)-Zellen, die die logische Funktion der Lookup Tables (LUTs) und das Routing definieren, nicht korrumpiert wird.

- Die kritischen Pfade künftiger Anwendungsdesigns sind dem Hersteller des FPGA prinzipbedingt nicht bekannt. Folglich können Techniken, die sowohl Wissen über das Anwendungsdesign, als auch über die eingesetzte Prozesstechnologie erfordern, nicht angewandt werden.

Eine triviale Übertragung von AVS-Ansätzen für ASICs auf FPGAs ist also nicht möglich. Vielmehr werden eigene Ansätze und Methoden benötigt, wie sie in diesem Kapitel konzipiert und diskutiert werden. Der Autor dieser Arbeit hat in diesem Themenfeld die Beiträge [B 1] und [B 2] veröffentlicht. In seiner Veröffentlichung [B 3] wird außerdem die Nutzung der in diesem Kapitel vorgestellten Sensorkalibrierung als Datenbasis für ein proaktives Alterungsmanagement behandelt. AVS für FPGAs war auch Themenbestandteil mehrerer studentischer Arbeiten am Institut für Angewandte Mikroelektronik und Datentechnik, die durch den Autor betreut wurden [C 1], [C 2], [C 3], [C 4], [C 5].

3.2 Stand der Wissenschaft und Technik

Um die für AVS charakteristische Guard-Band-Reduktion durch die Nutzung von Laufzeitwissen zu realisieren, muss ebendieses gewonnen werden. Dazu werden in der Literatur vielfältige Methoden beschrieben. Einzelne Ansätze ermitteln Störgrößen direkt, beispielsweise durch einen Temperatursensor oder sogenannte *Performance Counter*, die der Abschätzung der Alterung in Abhängigkeit vom Work Load dienen. Die große Mehrheit der Ansätze basiert jedoch auf einer Messung des Delays.

Solche Verfahren zur Messung des Delays werden in dieser Arbeit nach zwei Kriterien kategorisiert. Das Delay kann entweder direkt in den Datenpfaden des Anwendungsdesigns oder an anderer Stelle gemessen werden – beispielsweise durch zusätzliche Schaltungsblöcke, die nur zu diesem Zweck in räumlicher Nähe zum Anwendungsdesign platziert werden. Wird das Delay direkt im Datenpfad gemessen, soll dies als *In-Situ-Messung* bezeichnet werden. Ansätze, die das Verhalten an anderer Stelle überwachen, werden *Ex-Situ-Messungen* genannt. Das zweite Kriterium berücksichtigt den Zeitpunkt der Messung. Findet diese statt, während das Anwendungsdesign produktiv Daten verarbeitet, handelt es sich um eine *Online-Messung*, andernfalls um eine *Offline-Messung*. Letztere können einmalig oder auch periodisch, beispielsweise bei jedem Systemstart, durchgeführt werden. Die beiden Kriterien zur Einteilung von AVS-Ansätzen sind orthogonal. Somit ergeben sich vier Kategorien, die im Folgenden diskutiert werden.

3.2.1 Offline-Ex-Situ-Messung

Als *offline ex situ* werden solche Ansätze bezeichnet, die nicht direkt in den kritischen Pfaden des Anwendungsdesigns und auch nicht bei laufender Anwendung messen. In diese Kategorie fällt bei FPGAs auch das klassische Speed Binning, da dem Hersteller das Anwendungsdesign nicht bekannt ist. Ein Beispiel für Speed Binning ist die Vermessung und Einteilung von FPGAs in verschiedene Speed Grades durch Xilinx [A 75]. Eine weitere Anwendung des Speed Binnings durch Xilinx ist das Voltage Identification Bit [A 76]. Dieses ist ein in einem nichtflüchtigen

Speicher gesetztes Bit. Es kennzeichnet solche Instanzen eines FPGA-Modells, die aufgrund der Prozessvariation das nominelle Timing auch bei einer um 10 % niedrigeren Versorgungsspannung sicher erfüllen. Das Voltage Identification Bit kann von einem Anwendungsdesign auf dem FPGA ausgelesen werden. Falls es gesetzt ist, kann die Versorgungsspannung entsprechend abgesenkt werden.

Intel verwendet in den FPGAs der Stratix-10-Serie ein zweistufiges AVS-Verfahren. Zum einen wird einmalig nach der Produktion die Schaltgeschwindigkeit in Abhängigkeit von der Prozessvariation bestimmt. Diese wird dann dauerhaft in einen Read-only Memory (ROM) hinterlegt. Eine *Secure Device Manager* genannte Managementeinheit liest den ROM während des Systemstarts aus und senkt mittels Power Management Bus (PMBus) die Versorgungsspannung auf das benötigte Niveau ab. Zum anderen wird alle 100 ms die Temperatur gemessen. Fällt diese unter 10 °C, wird die Versorgungsspannung um 30 mV angehoben, um die Inverse Temperature Dependence zu kompensieren. Die Temperaturkompensation ist dabei mit einer Hysterese realisiert, sodass die Spannung erst bei Überschreiten von 20 °C wieder abgesenkt wird [A 77]. Diese kommerziell umgesetzte Anpassung der Versorgungsspannung verbindet ein ausgefeiltes Speed Binning, dessen Ergebnis nicht in der Typenspezifikation angegeben wird, sondern durch eine entsprechende Wahl der Versorgungsspannung genutzt wird, mit einer teilweisen Kompensation des Temperatureinflusses. Es handelt sich somit um eine Mischform zwischen AVS und Speed Binning, wobei die Temperatur nur unter vergleichsweise extremen Bedingungen durch eine einmalige Anhebung der Versorgungsspannung kompensiert wird. Alterungseffekte, Schwankungen der Versorgungsspannung und der Einfluss durch Temperaturänderungen bei Temperaturen von über 20 °C werden vernachlässigt.

Ein sehr interessanter Ansatz ist die Vermessung und Kartierung von FPGAs. Maragos et al. schlagen in [A 74] einen solchen vor. Das FPGA wird dazu in 408 Teilbereiche eingeteilt, die jeweils mittels eines Ringoszillators (ROs) bei verschiedenen Versorgungsspannungen und auch bei einem gezielt herbeigeführtem Voltage Drop charakterisiert werden. Somit wird eine 2-D-Karte der Schaltgeschwindigkeit erstellt. Diese wird dann genutzt, um das Anwendungsdesign so zu platzieren, dass möglichst die 'schnelleren' Regionen des FPGA genutzt werden. Auf diese Weise wird das Timing Guard Band sehr groß, was viel Spielraum für eine nachfolgende Anhebung der Taktfrequenz bei konstanter Versorgungsspannung über den in der Static Timing Analysis (STA) ermittelten Wert bietet. In dieser Arbeit werden explizit auch On-Chip Variations (OCVs) betrachtet. Da allerdings nur einige repräsentative Strukturen (3 LUTs pro Region) ausgemessen werden, können lediglich systematische OCVs erfasst werden. Unkorrelierte OCVs sowie räumlich unkorrelierte Alterungseffekte, deren Auswirkungen sich auch in direkt benachbarten Strukturen sehr stark unterscheiden können (beispielsweise durch ein früher auf dem FPGA ausgeführtes Design [A 52]), werden nicht erfasst.

Bei diesem Ansatz werden verschiedene Einflüsse auf das Timing nicht berücksichtigt. Dies betrifft zum einen die Temperatur und zum anderen vergleichsweise schnelle Störgrößen, wie das di/dt-Rauschen und den Teil des IR-Drops, den das Voltage Regulator Module (VRM) vergleichsweise schnell kompensiert, sodass der Einfluss nur für wenige Taktzyklen beobachtbar ist (vergleiche dazu Abschnitt 3.7 dieser Arbeit). Die vorstehend genannten Einflüsse auf das Timing können, wenn sie ignoriert werden, in der praktischen Anwendung zu Timing-Fehlern wie korrumpierten Daten oder Systemzuständen führen. Allerdings ließen sich diese Probleme

durch geeignete Erweiterungen des Ansatzes beseitigen – beispielsweise durch die Anwendung eines geeigneten residualen Guard Bands.

Die erreichbare Taktgeschwindigkeit eines Anwendungsdesigns wird in [A 74] mittels pseudozufälliger Eingangsvektoren ermittelt. Wie in Unterabschnitt 3.5.1 näher erläutert wird, ist das Delay eines kombinatorischen Schaltnetzes stark von den Eingangsvektoren abhängig. Somit wird auch bei einer vergleichsweise großen Zahl von Testvektoren möglicherweise der kritische Pfad nicht von einer Signalflanke durchlaufen. Dieses Problem betrifft aber lediglich die Auswertung und nicht das eigentliche Konzept.

3.2.2 Offline-In-Situ-Messung

Offline-In-Situ-Messungen des Delays ermöglichen es, direkt die kritischen Pfade des Anwendungsdesigns auszumessen, vermeiden dabei aber den Overhead von Online-In-Situ-Messungen, auf die im Unterabschnitt 3.2.4 näher eingegangen wird. Durch die in zeitlicher Hinsicht nur punktuell durchführbaren Offline-Messungen können zeitlich vergleichsweise schnell veränderliche Störgrößen, wie etwa die Temperatur oder Voltage Drops, nicht erfasst werden.

In [A 78] schlagen Li et al. eine innovative Möglichkeit zur Offline-Messung des Delays kritischer Pfade vor. Diese werden dazu in ROs umkonfiguriert. Um den Temperatureinfluss während der Messung zu minimieren, wird die Versorgungsspannung bis auf V_{INS} abgesenkt, also jenen Wert, bei dem sich Normal und Inverse Temperature Dependence gerade ausgleichen. Letzteres ist jedoch für Chips, die mit den aktuellsten Prozesstechnologien gefertigt sind, eher nicht mehr anwendbar, da hier typischerweise die Versorgungsspannung bereits niedriger als V_{INS} ist. Eine Erhöhung der Versorgungsspannung ist jedoch deutlich weniger trivial zu bewerkstelligen, da eine zu hohe Versorgungsspannung den Chip zerstören kann. Die Rekonfiguration des kritischen Pfades in einen RO ist jedoch unabhängig von der konkreten Prozesstechnologie ein sehr interessanter Ansatz. Die Arbeit zielt originär auf eine Messung der Alterung in ASICs ab, ließe sich aber mit geringen Anpassungen für AVS nutzen. Ein prinzipieller Nachteil dieses Ansatzes ist jedoch, dass er im Gegensatz zu den anderen hier besprochenen In-Situ-Messverfahren lediglich die Kombinatorik des kritischen Pfades berücksichtigt. Die Variabilität des Clock Tree, die, wie in Abschnitt 2.5 beschrieben, ebenfalls einen erheblichen Einfluss hat, wird nicht erfasst.

Eine weitere Möglichkeit zur Offline-In-Situ-Messung wird in Zhao et al. [A 79] vorgestellt. Der kritische Pfad der Anwendung wird repliziert und bei verschiedenen Temperaturen und Versorgungsspannungen vermessen, indem die maximale Frequenz ermittelt wird, bei der keine Timing-Fehler auftreten. Die so ermittelten VF-Arbeitspunkte werden in einer Tabelle gespeichert. Während des Anwendungseinsatzes werden die VF-Arbeitspunkte dann ähnlich wie bei konventionellem DVFS aus den Tabelleneinträgen gewählt. Eine Regelung der Versorgungsspannung über eine geschlossene Regelschleife findet nicht statt. Die VF-Arbeitspunkte sind aus dem Worst-Case-Temperatur-Szenario abgeleitet und gegenüber der Messung um ein residuales Guard Band von 5% pessimistischer. Dieses residuale Guard Band scheint jedoch willkürlich gewählt zu sein. Die Autor:innen geben eine Energieersparnis von 40% an.

Das in [A 79] vorgestellte Prinzip wird in verschiedenen Publikationen weiterentwickelt. In [A 80] wird näher auf das Problem der Replikation mehrerer kritischer Pfade eingegangen, wenn diese nicht unabhängig voneinander sind, sondern sich überlappen und dadurch Ressourcen in

mehreren kritischen Pfaden genutzt werden. Die triviale Lösung, für jeden dieser Pfade dedizierte Bitfiles zu erzeugen, ist in der Praxis nachteilig, da der Overhead bezüglich Energie, Zeit und Speicherplatz stark ansteigt. Mit der vorgeschlagenen Lösung kann die Zahl der benötigten Bitfiles stark reduziert werden. Für nicht berücksichtigte Variationen wird pauschal ein residuales Guard Band von 5 % vorgesehen. Zu diesem residualen Guard Band wird noch ein weiteres Guard Band eingeführt. Dieses kompensiert eine durch die Art der Replikation der kritischen Pfade verursachte Abweichung. Diese entsteht dadurch, dass die kritischen Pfade zwar im Mess-Bitfile mit denselben Logik- und Routing-Ressourcen repräsentiert sind, aber die Logikfunktion der LUTs in `XOR` geändert wird, um unkompliziert eine Signalflanke durch den gesamten Pfad zu erzeugen. Durch diese Änderung, insbesondere aber durch die isolierte Replikation nur des eigentlichen kritischen Pfades, ändert sich das Fan-Out der einzelnen Komponenten des kritischen Pfades. Dies wiederum verursacht eine Abweichung der Messergebnisse von den Verhältnissen in der eigentlichen Anwendungsschaltung. Die Autor:innen geben eine Reduktion der Verlustleistung um 33 % an. Insgesamt ist diese Publikation sehr innovativ und betrachtet das Gesamtproblem der Guard-Band-Reduktion in einer beeindruckenden Breite und Tiefe.

In [A 81] wird die Offline-In-Situ-Kalibrierung aus [A 79] um Online-Komponenten erweitert, um den Einfluss weiterer Einflussgrößen zu berücksichtigen. Konkret wird die Temperatur durch einen Sensor am Package des FPGA gemessen, um dann, basierend auf den Ergebnissen der Kalibrierung, den für die jeweilige Temperatur passenden VF-Arbeitspunkt auszuwählen. Zusätzlich wird basierend auf vom VRM ausgegebenen Werten zum Stromfluss der IR-Drop kompensiert. Als solchen bezeichnet man ein Einbrechen der Versorgungsspannung durch von der Stromstärke abhängige Spannungsabfälle über ohmsche Widerstände im Power Delivery Network (PDN). Die diesbezügliche Methodik wurde von den Autor:innen in [A 82] ausführlich beschrieben. Es wird eine Reduktion der Verlustleistung um 40 % angegeben. Bestimmte Störgrößen, wie beispielsweise schnelle Temperaturänderungen und lokale Hot Spots, können von der genutzten Temperaturmessung am Package des FPGA nicht erfasst und somit auch nicht ausgeregelt werden. Sowohl für die Messung der Temperatur, als auch für die dynamische Kompensation des Einbruchs der Versorgungsspannung durch ohmsche Widerstände (IR-Drop) werden externe, spezialisierte Komponenten verwendet. Zur Steuerung des Ablaufs der Kalibrierung ist sogar ein zusätzlicher programmierbarer Logikbaustein nötig. Die im Rahmen der vorliegenden Arbeit entwickelte Methodik realisiert hingegen ein AVS-System ohne zusätzliche, externe Hardwarekomponenten.

3.2.3 Online-Ex-Situ-Messung

Online-Ex-Situ-Ansätze zur Messung des Delays haben einige entscheidende Vorteile. So greifen sie nicht in das eigentliche Anwendungsdesign ein. Im Allgemeinen können sie wie ein beliebiger Intellectual Property Core (IP Core) in das Design eingebunden werden. Dadurch ist auch die Integration in die Electronic Design Automation (EDA)-Tools sehr einfach möglich. Da jedoch nicht unmittelbar im kritischen Pfad des Anwendungsdesigns gemessen wird, entsteht immer eine gewisse Abweichung durch lokale Variabilität. Solche lokal auftretenden Variationen können beispielsweise durch OCVs, Temperaturgradienten, örtlich begrenzte Voltage Drops oder lokal verschiedene Alterung entstehen. Durch die Platzierung mehrerer Sensoren, die entweder gleichmäßig über den Chip verteilt werden, oder gezielt in räumlicher Nähe zu kritischen

Pfaden eingebracht werden, kann diese Abweichung gemindert werden. Bestimmte Einflüsse, wie etwa unkorrelierte OCVs, können dennoch nur direkt im Datenpfad gemessen werden.

Chow et al. schlagen in [A 83] einen Online-Sensor auf Basis einer Inverterkette vor. Der Eingang dieser Inverterkette ist dauerhaft mit dem Taktsignal verbunden. Die dadurch verursachte atypisch hohe Schaltaktivität führt zu einer erhöhten Temperatur des Sensors durch Eigenerhitzung, die die Repräsentativität der Ergebnisse für die eigentliche Anwendungsschaltung verringert. Auch werden somit Effekte der tatsächlichen Arbeitslast beziehungsweise Schaltaktivität im Anwendungsdesign nicht berücksichtigt. Der Sensor wird mit Zufallsdaten kalibriert. Damit lässt sich jedoch keine zuverlässige Kalibrierung erreichen, wie in Unterabschnitt 3.5.1 erläutert wird. Die Autor:innen geben eine Energieersparnis von 54 % an.

Simevski et al. schlagen zwei verschiedene Online-Sensoren zur Abschätzung der Worst-Case-Alterung vor – je einen für Bias Temperature Instability (BTI) und Hot Carrier Injection (HCI) [A 50]. Die Arbeit zielt eigentlich auf eine Abschätzung der Alterung ab. Durch die Verwendung dieser Worst-Case-Sensoren könnte jedoch auch eine zuverlässige Operation der Anwendungsschaltung ohne zusätzliches Aging Guard Band gewährleistet werden. Beide Sensoren basieren auf Ketten von Invertern, die im Normalbetrieb eine Worst-Case-Alterung durch den entsprechenden Alterungsmechanismus erfahren. Dies wird im Falle von Negative-Bias Temperature Instability (NBTI) durch ein statisches Eingangssignal erreicht, das die Hälfte der Inverter dauerhaft in NBTI-Stress-Bedingungen hält. Für HCI dagegen wird ein Toggle-Flipflop (T-FlipFlop) verwendet, sodass mit jeder steigenden Taktflanke eine Signalflanke durch die Inverterkette läuft und somit eine Worst-Case-Alterung durch HCI hervorgerufen wird. Für die Messung der Alterung wird eine Signalflanke durch die jeweilige Inverterkette erzeugt. Mit der nächsten steigenden Taktflanke wird der Zustand der letzten n Inverter der Kette in Flipflops übernommen. Die Zahl der Inverter, die während eines Taktzyklus durchlaufen werden konnte, lässt sich nun aus den Registern auslesen.

Die Zielplattform des Ansatzes sind eigentlich ASICs. Eine Anpassung für FPGAs ist prinzipiell möglich, aber aufwändig. Dazu müsste zum einen das T-FlipFlop durch ein D-Flipflop mit negativer Rückführung des Ausgangssignals ersetzt werden. Zum anderen müssten die Inverterketten durch geeignete, auf dem FPGA vorhandene Strukturen ersetzt werden. Dafür bieten sich Ketten von LUTs oder die dedizierten Carry Chains an. Insbesondere für die Umsetzung des Worst-Case-NBTI-Sensors müssen dabei jedoch die realen Signalpegel beachtet werden, die nicht notwendigerweise denen des in der LUT abgebildeten äquivalenten ASIC-Gatters entsprechen [A 84].

In [A 85] schlagen Nunez-Yanez et al. vor, als Sensor eine Inverterkette zu nutzen, deren Eingangswert mit jedem Taktzyklus invertiert wird. Das Signal wird nach jedem Inverter abgegriffen und in einem Flipflop gespeichert. Um Probleme mit Metastabilität zu vermeiden, wird hinter jedes Flipflop ein weiteres Flipflop geschaltet. In Abhängigkeit von der Logiktiefe des kritischen Pfades wird per Bitmaske nur ein äquivalenter Teil der Inverterkette für die Auswertung berücksichtigt. Beinhalten die zu diesem Teil der Inverterkette zugehörigen Flipflops nicht abwechselnd die Werte `'0'` und `'1'`, besteht die Gefahr, dass das Timing des kritischen Pfades korrumpiert wird. Der VF-Arbeitspunkt wird dann entsprechend angepasst. In dem Paper werden einige Aspekte vernachlässigt. Obwohl das Delay einer LUT weitestgehend unabhängig von der implementierten logischen Funktion ist, ist das Timing einer Inverterkette gleicher

Logiktiefe nicht äquivalent zum kritischen Pfad. Die Ursachen hierfür sind unter anderem Unterschiede im Routing, ein abweichendes Fanout, OCVs sowie Gradienten von Temperatur und Versorgungsspannung.

Maragos et al. beschreiben in [A 86] ebenfalls einen auf einer Inverterkette basierenden Online-Sensor. Wie in [A 85] wird ein dem kritischen Pfad entsprechender Teil der Inverterkette berücksichtigt. Die Länge der berücksichtigten Inverterkette richtet sich jedoch im Gegensatz zu [A 85] nicht nach der Logiktiefe des kritischen Pfades. Stattdessen wird die Länge so gewählt, dass die per STA bestimmten Delays von Inverterkette und Sensor möglichst übereinstimmen. Es wird eine Kalibrierungsprozedur beschrieben, die aber lediglich das anfängliche Anpassen des VF-Arbeitspunkt darstellt. Dieses richtet sich allein nach den Messwerten des Sensors. Eine Kalibrierung des Sensors auf den kritischen Pfad des Anwendungsdesigns, wie sie in Abschnitt 3.5 dieser Arbeit beschrieben wird, findet nicht statt. Dadurch können OCVs und lokal verschiedene Alterung nicht hinreichend berücksichtigt werden. Die Autor:innen beschreiben eine entsprechende Abweichung des Sensor-Delays von dem des kritischen Pfades um ein bis zwei LUT-Delays. Im beschriebenen Fall führte dies zwar zu einem pessimistischeren Sensorergebnis, was unproblematisch ist. Nach Verständnis des Autors dieser Arbeit kann die Prozessvariation jedoch ebenso zu einem übermäßig optimistischen Sensor führen – was zu einem unzulässigen VF-Arbeitspunkt und somit potentiell zu Timing-Fehlern führen würde. Die Autor:innen geben eine Senkung der Leistungsaufnahme um 16-27 % bei gleichbleibender Performance an.

3.2.4 Online-In-Situ-Messung

Messungen direkt im Datenpfad ermöglichen die Berücksichtigung einiger Störgrößen, die von IP-Core-basierten Lösungen prinzipbedingt nicht erfasst werden können. Dazu zählen ganz offensichtlich die unkorrelierten OCVs. Aber auch die Alterung kann lokal sehr starke Unterschiede aufweisen, da sie abhängig vom Work Load und sogar von den statistischen Häufigkeiten bestimmter Signalzustände ist. Dies kann bei Ex-Situ-Messungen zu entsprechenden Ungenauigkeiten führen [A 87]. Andererseits muss für *Online-In-Situ-Messungen* offensichtlich sehr tief in das Anwendungsdesign eingegriffen werden.

Der wohl bekannteste Ansatz aus dieser Kategorie ist **Razor**. Dieser zuerst von Ernst et al. in [A 88] veröffentlichte Ansatz basiert auf speziellen Monitoring-Flipflops. Diese tasten das D-Signal nach der steigenden Taktflanke ein zweites Mal ab und speichern den Wert in einem zusätzlichen *Shadow-Register*. Verletzt eine späte Transition die Setup Time des Hauptregisters und korrumpiert dessen Inhalt, speichert das Shadow-Register dennoch den korrekten Wert. Mittels eines `XOR`-Gatters, das den Ausgang beider Register vergleicht, kann ein Fehler im Hauptregister festgestellt werden. In diesem Fall erlauben die in den Shadow-Registern gespeicherten, korrekten Werte einen Pipeline Rollback. Der detektierte Fehler wird also berichtigt. Dies ermöglicht es, einen VF-Arbeitspunkt zu wählen, der quasi frei von Guard Bands ist, da auftretende Timing-Fehler erkannt und berichtigt werden. Diesem sehr günstigen VF-Arbeitspunkt – er beinhaltet die niedrigsten Guard Bands aller hier betrachteten Ansätze – stehen allerdings erhebliche Nachteile gegenüber.

Neben dem zusätzlichen Ressourcenbedarf durch die Monitoring-Flipflops, der akzeptabel erscheint, entstehen an anderen Stellen deutlich größere Herausforderungen:

- Für die Ansteuerung der zusätzlichen Shadow-Register muss der Clock Tree angepasst werden. Dessen Fanout wird dadurch erheblich größer, was den Energiebedarf entsprechend steigen lässt [A 89].
- Die Ausgänge der `XOR`-Gatter aller Monitoring-Flipflops werden durch einen `OR`-Gatter-Baum zu einem einzelnen Fehlersignal zusammengefasst, welches dann den Pipeline Rollback auslöst. Dies muss offensichtlich innerhalb eines Taktzyklus erfolgen. Durch die resultierenden Timing-Anforderungen an diese große, globale Struktur ist sie schwer umzusetzen und verursacht einen großen Overhead [A 90].
- Die verzögerte Abtastung durch das Shadow-Register erfordert sehr strenge Hold Time Constraints, die verhindern, dass durch einen schnellen Pfad Daten aus dem nächsten Taktzyklus in das Shadow-Register übernommen werden. Diese zusätzlichen Hold Time Constraints erfordern das Einfügen einer großen Zahl von Buffern [A 89], [A 91]. Diese zusätzlichen Buffer verursachen auch einen erheblichen Energie-Overhead.
- Die Umsetzung des Pipeline Rollback erfordert einen komplexen, sehr tiefgehenden Eingriff in das Anwendungsdesign. Ein solcher macht den Designprozess aufwändiger und kostspieliger.

Razor wurde ursprünglich für ASICs entwickelt. Das Grundprinzip lässt sich aber gut auf FPGAs übertragen. Lediglich die in der ursprünglichen Razor-Veröffentlichung [A 88] für ASIC vorgesehenen Metastabilitäts-Detektoren basieren auf speziellen asymmetrischen Buffern, die sich nicht auf FPGA-Logikressourcen abbilden lassen. Dennoch gibt es erstaunlich wenig Literatur zur Umsetzung von Razor auf FPGA.

Minkovich et al. beschäftigen sich in [A 92] mit der Umsetzung der Fehlerkorrektur. Es werden Designtechniken zur Realisierung von dafür geeigneten Finite State Machines (FSMs) vorgeschlagen. Diese sind dediziert für LUT-basierte FPGAs konzipiert.

In [A 93] wird die Razor-Fehlerkorrektur für bidirektional kommunizierende Prozessorarrays untersucht. Es handelt sich zwar nicht um einen dedizierten FPGA-Ansatz, aber die prototypische Umsetzung auf einem FPGA lässt darauf schließen, dass der Ansatz für FPGA geeignet ist. Der Razor-Ansatz wird in dieser Arbeit nicht zur Senkung der Leistungsaufnahme, sondern zur Verbesserung des Durchsatzes genutzt. Wobei eine Anwendung zur Reduktion der Verlustleistung mit geringen Anpassungen möglich wäre. Der höchste Durchsatz wurde bei einer gegenüber dem STA-Ergebnis um 66 % höheren Taktfrequenz erreicht, wobei in 1-5 % der Takte Timing-Fehler auftraten. Durch die zur Korrektur nötigen Pipeline Stalls liegt die Steigerung des Durchsatzes etwas niedriger, bei immer noch beeindruckenden 63 %.

Ragavan et al. kombinieren die aus Razor bekannte Fehlererkennung mit einer Messung des verbleibenden Guard Bands durch ein zweites Shadow-Register, welches das Datensignal *vor* der Taktflanke abtastet [A 91]. Die Funktionsweise dieses zweiten Shadow-Registers entspricht weitestgehend den Monitoring-Flipflop-basierten AVS-Systemen ohne Fehlerkorrektur, die im nächsten Absatz behandelt werden und deren Funktionsweise in Abschnitt 3.3 ausführlich diskutiert wird. Als interessante Neuerung wird ein veränderliches *Speculative Window* genutzt. Das System wurde auf einem Virtex-7-FPGA umgesetzt. Die Autor:innen reklamieren, dass durch

die Shadow-Latches eine Fehlerkorrektur möglich sei. Eine nähere Beschreibung fehlt jedoch. Dem Autor dieser Arbeit war es nicht möglich, das diesbezügliche Verfahren nachzuvollziehen.

Eine weitere Gruppe von Online-In-Situ-Ansätzen basiert ebenfalls auf **Monitoring-Flipflops**, allerdings **ohne einen Mechanismus zur Fehlerkorrektur**. Im Gegensatz zu den Razor-Ansätzen wird der D-Eingang durch das Shadow-Register hier jedoch *vor* der Taktflanke abgetastet. Dadurch tritt bei Annäherung an die kritische Versorgungsspannung ein Timing-Fehler zuerst im Shadow-Register auf. Da dieses wiederum mittels `XOR`-Gatter mit dem Hauptregister verglichen wird, kann eine drohende Korrumpierung des Timings des Hauptregisters erkannt werden. Durch Gegenmaßnahmen, also die Anhebung der Versorgungsspannung oder das Absenken der Taktfrequenz, soll ein Timing-Fehler im Hauptregister verhindert werden.

Durch den Verzicht auf Fehlerkorrekturmaßnahmen entfällt ein erheblicher Teil des Overheads. Die Abtastung durch das Shadow-Register vor der eigentlichen Taktflanke (im Gegensatz zur Abtastung nach der Taktflanke bei Razor) vermeidet zudem die Hold-Time-Probleme von Razor-Flipflops. Auch dadurch sinkt der Overhead im Vergleich zu Razor erheblich. Allerdings ist das tatsächliche Delay einer kombinatorischen Schaltung von den Eingangsvektoren abhängig. Dadurch ist eine Messung des Guard Bands anhand des durch unbekannte Datenströme verursachten Delays mit einer gewissen Unsicherheit behaftet. Diese Problematik wird im nachfolgenden Abschnitt 3.3 näher untersucht.

Nunez-Yanez et al. haben eine ganze Reihe von Arbeiten veröffentlicht, in denen diese Grundidee auf FPGA-Systeme angewendet wird. In [A 94] wird ein für moderne FPGA-Architekturen angepasstes Monitoring-Flipflop vorgeschlagen. Dieses wird in Form eines *Soft Macros* bereitgestellt. Nach der Identifikation der kritischen Pfade des Anwendungsdesigns durch eine STA werden die Flipflops an den Enden der kritischen Pfade durch Monitoring-Flipflops ersetzt. Diese werden dann genutzt, um bei aktivem Anwendungsdesign die Versorgungsspannung zu regeln. Dazu wird diese so lange abgesenkt, bis die Monitoring-Flipflops Signaltransitionen kurz vor der Taktflanke feststellen. Darüber hinaus wird ein *Elongate* genannter EDA-Flow präsentiert. Dieser bewerkstelligt die Integration der Monitoring-Flipflops und der weiteren benötigten Komponenten in ein Anwendungsdesign.

In [A 95] wird dieser Ansatz auf ein Xilinx-Zynq-System angewendet, also ein System, das ein FPGA mit einem als ASIC implementierten Mikroprozessor kombiniert. Zudem wird er um ein Power Gating erweitert, das auf den FPGA-Teil des Systems angewendet werden kann. Für das Anwendungsszenario aus dem Bereich des Video-Encodings – es wurde ein Motion Estimation Core implementiert – konnte die Verlustleistung bei gleichbleibender Leistungsfähigkeit um bis zu 60 % gesenkt werden.

In [A 96] wird der Ansatz so erweitert, dass auch Pfade, die in einem Block Random-Access Memory (BRAM) enden, überwacht werden können. Zudem wird die Datenabhängigkeit des Monitorings diskutiert. Für die Zielanwendung, eine Video Fusion, wird als Lösung vorgeschlagen, das Delay für eine große Zahl an Taktzyklen zu überwachen bevor eine Entscheidung über eine Veränderung des VF-Arbeitspunktes getroffen wird. Auf diese Fragestellung wird im folgenden Abschnitt 3.3 der vorliegenden Arbeit näher eingegangen. Für die kombinierte Anwendung von AVS und Power Gating wird eine Reduktion des Energiebedarfs um 60-70 % genannt. In [A 97] wird dieser Ansatz auf die Inferenz eines Deep Neural Network (DNN) angewendet.

Zudem wird neben dem AVS, für das der Autor eine fehlerfreie Funktion angibt, ein weiterer Betriebsmodus eingeführt. Dieser nimmt bewusst Fehler in Kauf, um den Energiebedarf weiter zu senken. Bei diesem Betriebsmodus handelt es sich also um ein Voltage Underscaling im Sinne des Approximate Computing. Die Autor:innen gehen jedoch nicht auf die nötigen Constraints und Randbedingungen für einen solchen Einsatz ein, wie sie in Unterabschnitt 3.3.6 dieser Arbeit beschrieben werden.

Levine et al. entwerfen in [A 98], [A 99] und [A 100] ein AVS-System für FPGA auf Basis einer Online-In-Situ-Messung von Timing Slacks. Die Slack-Messung wird durch Shadow-Register realisiert, die von einem dedizierten Taktsignal gesteuert werden. Dieses eilt dem Taktsignal des Anwendungsdesigns um einen veränderlichen Phasenwinkel voraus. Die Ansteuerung der Shadow-Register über ein dediziertes Taktsignal bietet erhebliche Vorteile. Insbesondere ist der Phasenwinkel zum Taktsignal des Anwendungsdesigns frei wählbar und kann präzise gesteuert werden. Dies wird von den Autor:innen sehr elegant genutzt. So werden die gegenüber einem Standardregister benötigten zusätzlichen Komponenten erst nach dem eigentlichen Anwendungsdesign auf dem FPGA platziert. Dadurch erreicht eine Signalflanke Haupt- und Shadow-Register stets mit einem variierenden zeitlichen Versatz. Unter Nutzung des variablen Phasenwinkels der Shadow Clock wird dieser für jedes Shadow-Register bestimmt und die Messung des Timing Slacks entsprechend kalibriert.

Allerdings ist ein zusätzliches Taktsignal – und insbesondere dessen Verteilung mit einem eigenen Clock Tree – auch mit einem erheblichem Overhead verbunden. Dieser entsteht einerseits durch den Energiebedarf des zusätzlichen Clock Trees. So trägt die Bereitstellung des Taktsignals typischerweise 20-40 % zum Gesamtenergiebedarf bei [A 56]. Da nicht alle Register als Monitoring-Flipflops mit zwei Taktsignalen implementiert werden, ist der Energiebedarf des Shadow-Register-Clock-Trees kleiner. Es muss aber dennoch von einem signifikanten Energie-Overhead ausgegangen werden. Andererseits müssen auch die nötigen Ressourcen wie Clock-Manager und Clock-Buffer in hinreichender Menge auf dem FPGA vorhanden sein. Typischerweise sind moderne FPGAs mit mehreren Clock-Managern und diversen Ressourcen zur Verteilung von Taktsignalen ausgestattet. Sind solche ohnehin noch ungenutzt, entsteht kein Overhead. Für komplexe Designs, die die Clocking-Ressourcen des FPGA bereits auslasten, muss jedoch im Zweifel auf ein FPGA-Modell mit mehr Ressourcen gewechselt werden. In diesem Fall ist der Overhead erheblich.

Die Autor:innen benennen die Notwendigkeit eines residualen Guard Bands für schnelle Störgrößen. Dessen Dimensionierung bleibt jedoch weitestgehend unklar. Darüber hinaus werden verschiedene FSMs zur Steuerung des AVS diskutiert – insbesondere auch für den komplexen Fall eines kombinierten AVS und DVFS unter Energierestriktion. In der sehr gründlichen experimentellen Evaluation wird eine mittlere Energieersparnis von 33,5 % bei einer Temperatur des FPGAs von 27 °C erreicht.

Giesen et al. schlagen in [A 101] eine Änderung der FPGA-Architektur durch die Hersteller vor: jedes Register des FPGAs soll durch Monitoring-Flipflops ersetzt werden. Dabei wird ein Shadow-Register mit variablem Phasenwinkel verwendet, wie es auch von Levine et al. [A 98] vorgeschlagen wurde. Eine wesentliche Neuerung ist, dass kein kombinatorischer `OR`-Tree verwendet wird, um die erkannten Abweichungen zwischen Haupt- und Shadow-Register zu sammeln. Stattdessen wird ein Set-Reset Latch (SR Latch) in jedem Monitoring-Flipflop

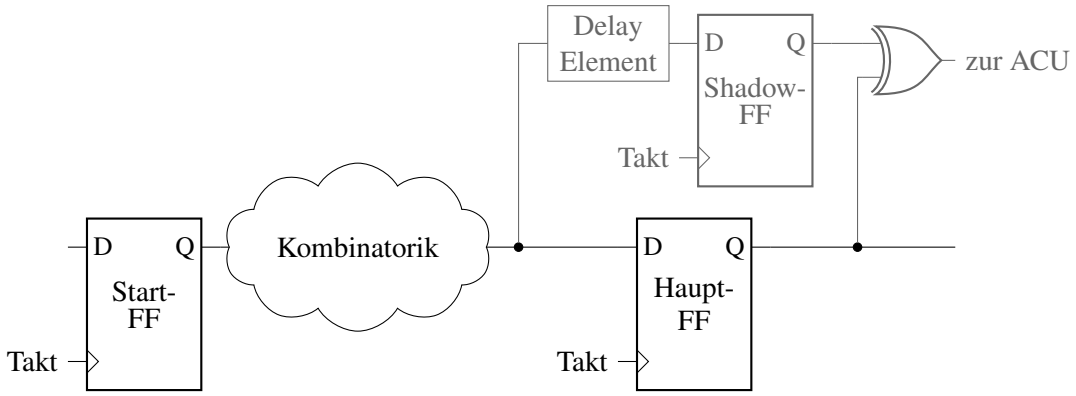


Abbildung 3.2: Aufbau eines In-Situ-Monitoring-Flipflops nach Nunez-Yanez [A 94]

vorgesehen, welches periodisch über eine Scan Chain ausgelesen wird. Dadurch kann für jedes Monitoring-Flipflop einzeln ausgewertet werden, ob im betrachteten Zeitraum mindestens eine Abweichung festgestellt wurde. Diese Information, die für jedes Flipflop im gesamten FPGA vorliegt, wird dann genutzt, um eine sehr fein aufgelöste Messung des Propagation Delays jeder Logikstufe des kritischen Pfades durchzuführen. Dadurch sollen besonders langsame Logikelemente erkannt und durch ein alternatives Placement und Routing umgangen werden.

3.3 Datenabhängigkeit von auf Monitoring-Flipflops basierenden AVS-Ansätzen ohne Fehlerkorrektur

Wie im vorherigen Kapitel dargelegt, schlagen vielfältige Publikationen die Überwachung des Timings mittels Monitoring-Flipflops an den Endpunkten kritischer Pfade vor. Im Gegensatz zu den 'Razor-artigen' Ansätzen wird jedoch nicht durch ein zweites Register festgestellt, ob eine späte Signaltransition fehlerhafte Daten verursacht, um diese zu berichtigen. Vielmehr wird auf Signaltransitionen kurz vor der Taktflanke abgestellt, um so eine drohende Verletzung des Timings rechtzeitig zu erkennen und Gegenmaßnahmen einzuleiten, bevor korrumpierte Daten in Registern gespeichert werden. Der Zeitraum vor der Taktflanke, der auf solche späten Transitionen überwacht wird, wird als *Speculative Window* bezeichnet [A 94]. In diesem Kapitel werden die Vor- und Nachteile dieser Klasse von AVS-Ansätzen diskutiert. Zur Veranschaulichung wird der Ansatz von Nunez-Yanez et al. genutzt, wie er in einer Reihe von hochwertigen Veröffentlichungen für verschiedene Zielplattformen und -anwendungen beschrieben ist [A 94], [A 95], [A 96], [A 97]. Die prinzipiellen Erkenntnisse lassen sich jedoch auf verwandte Ansätze übertragen.

3.3.1 Grundsätzliche Funktionsweise

Geeignete Monitoring-Flipflops können auf verschiedene Weise konstruiert werden. Sie enthalten neben dem eigentlichen Flipflop, das für die Funktion der Anwendungslogik nötig ist, ein weiteres sogenanntes *Shadow-Register*, welches den Zustand des D-Ports des Hauptregisters kurz vor der Taktflanke speichert. Nach der Taktflanke wird mittels eines XOR-Gatters eine eventuelle Abweichung der gespeicherten Daten in Haupt- und Shadow-Register erkannt und

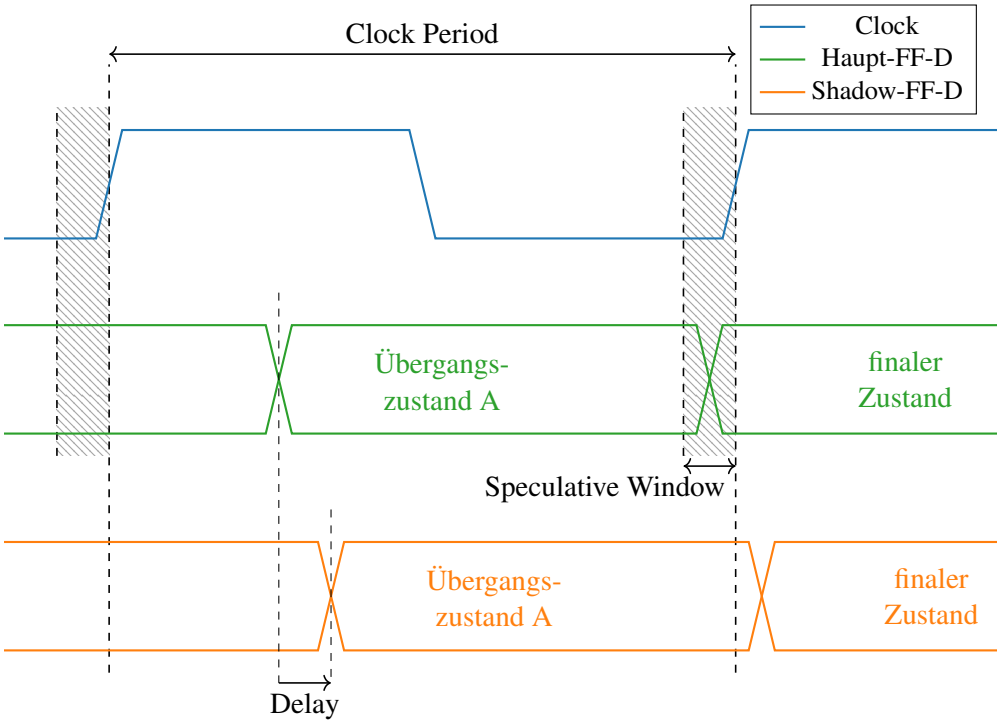


Abbildung 3.3: Wave Form für eine vom Monitoring-Flipflop erkannte späte Transition

von einer AVS Control Unit (ACU) entsprechend verarbeitet. Das Abgreifen des D-Signals zu einem anderen Zeitpunkt als durch das Hauptregister kann dabei einerseits über ein zusätzliches, phasenverschobenes Taktsignal oder andererseits über eine geeignete Verzögerung des D-Signals realisiert werden. Das von Nunez-Yanez et al. verwendete Monitoring-Flipflop ist in Abbildung 3.2 dargestellt. Eine sogenannte *späte Transition* des Datensignals innerhalb des Speculative Windows, also kurz vor der Taktflanke, führt dazu, dass Haupt- und Shadow-Flipflop abweichende Werte speichern. Abbildung 3.3 illustriert diesen Fall beispielhaft.

Die Regelung des VF-Arbeitspunkts erfolgt in den Publikationen [A 94], [A 95] und [A 96], indem die Taktfrequenz erhöht wird, wenn für eine definierte Zeitspanne keine späte Signaltransition detektiert wurde. Diese Zeitspanne wird nachfolgend Evaluationszeitraum genannt. Wurde die Taktfrequenz erhöht, beginnt jeweils wieder ein neuer Evaluationszeitraum und – falls keine späte Signaltransition detektiert wurde – wird die Taktfrequenz weiter erhöht. Wird jedoch eine Signaltransition innerhalb des Speculative Windows festgestellt, wird der aktuelle VF-Arbeitspunkt beibehalten und keine weitere Erhöhung der Taktfrequenz mehr vorgenommen. Ein offensichtlicher Nachteil dieses Regelalgorithmus ist, dass, wenn der finale Zustand erreicht ist, nicht mehr auf eine Herabsetzung der Schaltgeschwindigkeit reagiert werden kann. Eine solche kann jedoch leicht, beispielsweise durch eine Veränderung der Temperatur, auftreten. Dieser Fehler lässt sich beheben, indem bei Detektion einer späten Transition die Frequenz abgesenkt wird, wie dies auch in [A 97] vorgeschlagen wird.

3.3.2 Simulation zur quantitativen Bewertung der Datenabhängigkeit Monitoring-Flipflop-basierter AVS-Ansätze

Unabhängig von der konkreten Ausgestaltung des Regelungsalgorithmus ist jedoch allen Ansätzen, die Monitoring-Flipflops zur Bestimmung des VF-Arbeitspunktes nutzen, gemein, dass die gemessenen Delays von den Eingangsvektoren abhängig sind, die gerade von der Logik verarbeitet werden. Wie in Unterabschnitt 3.5.1 näher ausgeführt wird, ist das tatsächliche Delay eines Pfades stark von den Eingangsvektoren, genauer von der durchlaufenen Sequenz von Eingangsvektoren, abhängig. Dass tatsächlich der komplette kritische Pfad durchlaufen wird, ist je nach konkretem Logikdesign nur selten der Fall. Eine korrekte Festlegung des VF-Arbeitspunktes kann jedoch nur erfolgen, wenn der kritische Pfad durchlaufen wird, sodass das Monitoring-Flipflop an dessen Ende eine späte Transition feststellen kann. Alternativ genügt auch ein etwas kürzerer Pfad mit gleichem Endpunkt, wenn dieser um weniger als die Dauer des Speculative Windows schneller ist als der kritische Pfad. Dieser Sachverhalt soll im Weiteren als Datenabhängigkeit des Monitorings bezeichnet werden. Der Vollständigkeit halber sei angemerkt, dass es dabei um die Eingangsvektoren der kombinatorischen Logik geht – es kann sich also auch um Instruktionen handeln. Um trotz der Datenabhängigkeit des Monitorings eine zuverlässige Bestimmung eines geeigneten VF-Arbeitspunkt zu ermöglichen, wird in [A 96] vorgeschlagen, die Pfade für einen hinreichend langen Zeitraum zu überwachen. Offensichtlich erhöht sich die Wahrscheinlichkeit, dass ein hinreichend langer Pfad von einer Signalflanke durchlaufen wird, mit der Anzahl der betrachteten Übergänge zwischen Eingangsvektoren – vorausgesetzt die Eingangsvektoren weisen eine geeignete statistische Verteilung auf. Die quantitative Analyse dieses, für die betreffenden AVS-Techniken essentiellen, Zusammenhangs ist jedoch keineswegs trivial. Nach Wissensstand des Autors dieser Arbeit wurde eine entsprechende Analyse bisher nicht veröffentlicht.

Da auf Monitoring-Flipflops basierende AVS-Systeme jedoch eine ganze Reihe von Vorteilen aufweisen und vielversprechende Ergebnisse veröffentlicht wurden, wird im Folgenden eine **quantitative Analyse der Fehlerwahrscheinlichkeit durch die Datenabhängigkeit des Monitorings** vorgenommen. Da in diesem Fall der Einfluss der Eingangsvektoren von anderen Einflussfaktoren isoliert werden soll, wird als Untersuchungsform eine Gate-Level-Timing-Simulation gewählt. Dazu wird die Logik in Xilinx Vivado für einen Virtex-7 FPGA synthetisiert, gemappt und geroutet. Die Netzliste wird dann mit den Post Place and Route (P&R) Delays annotiert. Die in der Simulation verwendeten Delays sind also statisch und unterliegen weder der Prozessvariation, noch irgendeiner anderen dynamischen Einflussgröße. Somit können auftretende Fehler eindeutig der Datenabhängigkeit des Monitorings zugeordnet werden.

Die Monitoring-Flipflops sowie die Anpassung der Taktfrequenz durch das AVS-System werden in der Test Bench abgebildet. Dabei wurde das in [A 94] beschriebene System präzise nachgebildet. Um möglichst lange Zeiträume simulieren zu können, wurde eine kleine, aber als Komponente häufig verwendete Logikschaltung als Anwendungsdesign gewählt: ein 32-Bit-Ripple-Carry-Addierer. Bei dem von Nune-Yanez et al. vorgeschlagenen Regelalgorithmus ist vor allem die Phase, in der die Taktfrequenz gesteigert wird, von Interesse. In dieser wird der VF-Arbeitspunkt festgelegt – sie bestimmt damit das Verhalten des Systems. Deshalb werden vergleichsweise kurze Systemläufe von 4 ms Länge simuliert, wobei dennoch die Gesamtlaufzeit des Systems deutlich größer ist als die Zeit zum Einstellen der Taktfrequenz. Um verschiede-

Tabelle 3.1: Übersicht über die zur Bewertung der Datenabhängigkeit des Monitorings genutzten Constrained-Random-Eingangsvektoren

Bezeichnung	Eigenschaften der Eingangsvektoren		
Gleichverteilt	gleichverteilt		
WGN-A	normalverteilt	$\mu = 0$	$\sigma = 2^8$
WGN-B	normalverteilt	$\mu = 0$	$\sigma = 2^{10}$
WGN-C	normalverteilt	$\mu = 0$	$\sigma = 2^{16}$
WGN + Gleichanteil	normalverteilt	$\mu = 2^8$	$\sigma = 2^{16}$
Gauß, nichtnegativ	normalverteilt mit den angegebenen Parametern μ und σ, davon nur nichtnegative Werte	$\mu = 0$	$\sigma = 2^8$

ne Anwendungsszenarien abzubilden, werden Constrained-Random-Test-Eingangsvektoren mit verschiedenen statistischen Eigenschaften und anderen Beschränkungen genutzt. Konkret werden neben gleichverteilten Eingangswerten auch verschieden parametrisierte Normalverteilungen verwendet. Diese entsprechen, da sie zeitlich unkorreliert sind, der Definition des White Gaussian Noise (WGN). Einzelheiten können aus Tabelle 3.1 entnommen werden.

3.3.3 Ergebnisse und Diskussion

Abbildung 3.4 zeigt die wichtigsten Ergebnisse der Simulation. Im oberen Teil der Abbildung ist die durchschnittlich vom AVS-System gewählte stationäre Periode des Taktsignals (T_{clk}) dargestellt. Es handelt sich also um die Taktperiode, die nach Ende der Evaluation für den weiteren Betrieb gewählt wird. Das mittels Path Delay Test Pattern ermittelte Worst-Case-Delay ist gestrichelt eingezeichnet. Da in dieser Simulation die Register in der Test Bench implementiert sind und alle Unsicherheiten des Timings bis auf die Datenabhängigkeit ausgeschlossen wurden, entspricht dies der optimalen Taktperiode. Um einen fehlerfreien Betrieb des Systems auch bei ungünstigen Eingangsvektoren zu gewährleisten, muss die Taktperiode mindestens in dieser Höhe gewählt werden.

Kürzere Evaluationszeiträume führen offensichtlich dazu, dass kleinere Taktperioden gewählt werden. Dieser Effekt erklärt sich dadurch, dass es für kürzere Evaluationszeiträume weniger wahrscheinlich ist, dass während des Evaluationszeitraumes ein Übergang zwischen zwei Eingangsvektoren ein sehr langes Delay der Schaltung hervorruft. Dieser sehr offensichtliche Zusammenhang stellt interessanterweise jedoch keineswegs die dominierende Einflussgröße dar. Vielmehr überwiegt deutlich der Einfluss der statistischen Verteilung der Eingangsvektoren. Ist diese günstig (in dem Sinne, dass sie lange Delays der Logik mit hoher Wahrscheinlichkeit hervorruft), genügen auch kurze Evaluationszeiträume, um eine geeignete Taktfrequenz zu ermitteln. Im ungünstigen Fall genügen jedoch auch vergleichsweise lange Evaluationszeiträume nicht, um eine Taktfrequenz zu ermitteln, die den fehlerfreien Betrieb gewährleistet. Die durchschnittlich gewählte Taktperiode beträgt beispielsweise für die Verteilung 'Gauß, nichtnegativ' selbst beim längsten betrachteten Evaluationszeitraum nur ein Drittel des Delays des kritischen Pfades. Bei den Verteilungen 'WGN-A' und 'WGN-B' hingegen wird eine Taktperiode gewählt,

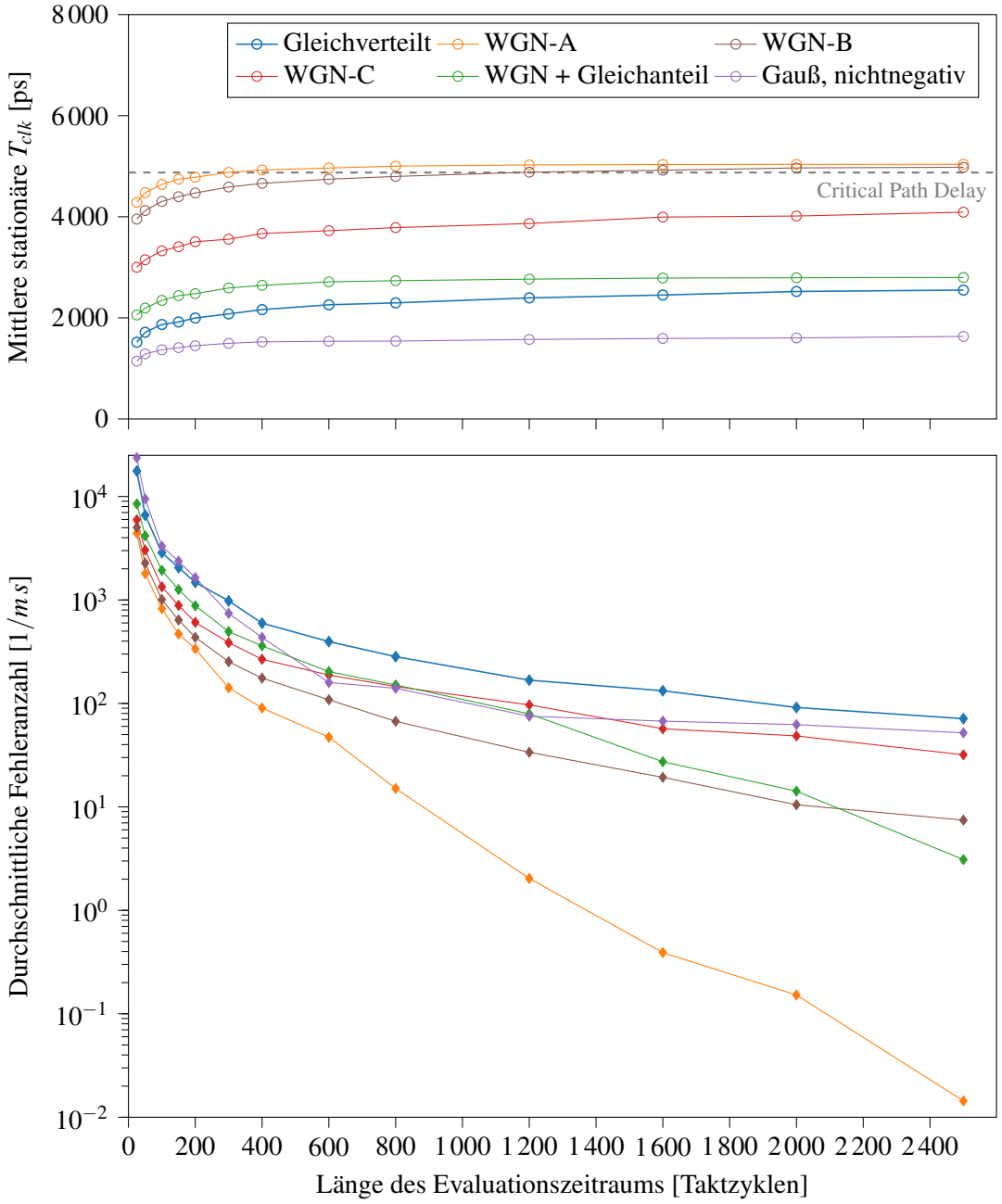


Abbildung 3.4: Gate-Level-Simulation zur quantitativen Bestimmung der durch datenabhängige Delays verursachten Fehler bei Monitoring-Flipflop-basierten AVS-Systemen

die im Mittel größer als das Worst-Case-Delay der Schaltung ist. Die Ursache hierfür ist, dass das System aufhört, die Taktperiode zu erhöhen, sobald die Summe aus dem größten innerhalb des Evaluationszeitraumes beobachteten Delay und dem Speculative Window größer als

die Taktperiode ist. Wird also das Worst-Case-Delay beobachtet, kann die gewählte Taktperiode größer als selbiges sein.

Im unteren Teil der Abbildung ist die durchschnittliche Fehleranzahl dargestellt. Ein Fehler wird hier definiert als das durch ein korrumpiertes Timing verursachte Speichern eines falschen Datenwertes im Haupt-Flipflop. Auch hier zeigt sich der naheliegende Zusammenhang von kürzeren Evaluationszeiträumen und höheren Fehlerraten, der sich offensichtlich aus der Wahl einer kürzeren Taktperiode ergibt. Die Eigenschaften der Eingangsvektoren spielen jedoch auch hier eine erhebliche Rolle. Insbesondere ist die Fehlerwahrscheinlichkeit keineswegs ausschließlich von der gewählten Taktperiode abhängig. Man beachte in diesem Zusammenhang, dass die Verteilung 'Gauß, nichtnegativ' die optimale Taktperiode am weitesten unterschreitet, und zwar unabhängig vom gewählten Evaluationszeitraum. Dennoch weist sie für längere Evaluationszeiträume nicht die höchste Fehlerrate auf.

In [A 100] gehen die Autor:innen davon aus, dass eine hinreichende Entropie der Daten dazu führt, dass die Monitoring-Flipflops zuverlässig eine drohende Korrumpierung des Timings erkennen. Um diese These zu evaluieren, wurden die Verteilungen 'WGN-C' und 'WGN + Gleichanteil' in die Untersuchung aufgenommen. Beide folgen einer Normalverteilung mit einer Standardabweichung von 2^{16} Bit und unterscheiden sich lediglich in einem geringfügig (um 2^8) verschobenen Erwartungswert. Dadurch ergibt sich de facto eine identische Entropie beider Datenströme [3.2]. Dennoch unterscheidet sich das hervorgerufene Verhalten des AVS-Systems erheblich. So führen Daten der Verteilung 'WGN-C' durchgängig zu wesentlich längeren Taktperioden. Das Delay des kritischen Pfades wird also weniger weit unterschritten als bei der Verteilung 'WGN + Gleichanteil'. Interessanterweise ist die mittlere Fehlerzahl für längere Evaluationszeiträume bei der Verteilung 'WGN-C' dennoch wesentlich größer. Diese Ergebnisse zeigen deutlich, dass die Entropie kein geeignetes Kriterium ist, um zu beurteilen, ob ein Datenstrom eine zuverlässige Messung des Delays durch Monitoring-Flipflops ermöglicht.

Schlussendlich zeigt der Vergleich der Verteilungen 'WGN-A' und 'WGN-B', die sich lediglich in der Standardabweichung unterscheiden, dass auch solche vermeintlich ähnlichen Verteilungen der Eingangsvektoren zu gravierenden Unterschieden in der Fehlerzahl führen können. In diesem Fall unterscheiden sich die Fehlerwahrscheinlichkeiten um nahezu zwei Größenordnungen. Es sei besonders darauf hingewiesen, dass die Fehlerwahrscheinlichkeit für die Verteilungen 'WGN-A' und 'WGN-B' bei großen Evaluationszeiträumen stark abnimmt. Sie verschwindet jedoch nicht, obwohl die durchschnittliche Taktperiode größer als das Worst-Case Delay ist. Die Ursache hierfür ist, dass auch in diesem Fall vereinzelt sehr ungünstige Sequenzen von Eingangsvektoren im Evaluationszeitraum ausgewertet werden und dann zu einer zu klein gewählten Taktperiode führen.

Die Simulationsergebnisse zeigen also deutlich, dass die Fehlerrate signifikant von der Verteilung der Eingangsdaten abhängt. Der quantitative Zusammenhang ist dabei offensichtlich kom-

[3.2] Es ist die Entropie im Sinne der Informationstheorie gemeint. Es sei angemerkt, dass die Entropien beider Verteilungen theoretisch nicht exakt gleich sind. Der Grund dafür ist das 'Abschneiden' der Gaußverteilungen durch den begrenzten Wertebereich von 32-Bit-Zahlen. Für die beiden hier betrachteten Verteilungen ist dieser Effekt jedoch vernachlässigbar. Konkret ist er so klein, dass er bei einer Berechnung der Entropien beider Datenströme mit 64-Bit-Fließkommagenauigkeit nicht mehr nachweisbar ist.

plex und lässt sich nach Einschätzung des Autors nicht mit angemessenem Aufwand durch Überwachung der Datenströme vorhersagen. Unbenommen dessen ist dies für bestimmte Fälle möglich und sicherlich auch sinnvoll. So lässt sich beispielsweise trivial ein pathologischer Fall konstruieren, in dem die Eingangsvektoren für eine hinreichend lange Zeit unverändert bleiben. In diesem Fall ist auch der Ausgang einer kombinatorischen Logik konstant. Somit kann das Monitoring-Flipflop keine späten Transitionen beobachten und es würde die Taktperiode Null gewählt. Wird die Schaltung anschließend mit veränderlichen Eingangsvektoren beaufschlagt führt dies sofort zur Verletzungen des Timings und korrumpierten Registerinhalten. Dieser durchaus praxisrelevante Fall lässt sich vergleichsweise leicht erkennen und durch geeignete Gegenmaßnahmen, wie das Einfrieren der Taktfrequenz auf den aktuellen Wert, beheben. Für die in dieser Arbeit aufgezeigte starke Abhängigkeit der Ergebnisse auch von kleineren Änderungen der statistischen Eigenschaften der verarbeiteten Daten ist dies jedoch, wenn überhaupt, nur mit großem Aufwand vorstellbar. Erschwerend kommt hinzu, dass der Zusammenhang zwischen sich einstellender Fehlerrate und Statistik der Eingangsvektoren von der konkreten Logikschaltung abhängig ist. Der Zusammenhang müsste also für jede Anwendungslogik und jede statistische Verteilung der Daten charakterisiert werden.

3.3.4 Nicht detektierte Fehler

Ein weiterer interessanter Effekt, der während der Untersuchung von Monitoring-Flipflops im Rahmen dieser Arbeit aufgefallen ist, sind **nicht detektierte Fehler**. Zur Erklärung dieses Effekts sei eine Verteilung der Eingangsvektoren angenommen, die fast ausschließlich kurze Delays verursacht. Die einzige Ausnahme sind seltene sehr lange Delays. Delays mittlerer Länge treten nicht auf. Ein auf Monitoring-Flipflops basierendes AVS-System wird nun häufig eine sehr hohe Taktfrequenz wählen, die jedoch bei kurzen Delays keine Timing-Fehler hervorruft. Bei den sehr selten auftretenden langen Delays wird dann offensichtlich das Timing korrumpiert. Da das lange Delay jedoch sehr viel größer als das kurze ist, ist es möglich, dass während des Speculative Windows das D-Signal stabil ist und sich erst nach der eigentlichen Taktflanke ändert. In diesem Fall werden unbemerkt falsche Daten im Register gespeichert und weiterverarbeitet. Dieser Fall wird in Abbildung 3.5 schematisch dargestellt. Ein nicht detektierbarer Fehler kann gemeinhin als besonders problematisch eingestuft werden, da keine Gegenmaßnahmen ergriffen werden können. Der Effekt der nicht-detektierbaren Fehler ist jedoch nicht auf die sehr konstruierte Situation beschränkt, die in diesem Beispiel beschreiben wurde. Vielmehr traten solche Fehler während der Simulationen immer wieder auf, wie Abbildung 3.6 zeigt.

Dabei wurde eine sehr strenge Definition für nicht detektierte Fehler angewendet. Gezählt wurden nur solche Fehler, bei denen die Monitoring-Flipflops keine späte Transition erkannten und die zudem auftraten, bevor im jeweiligen Evaluationslauf überhaupt eine späte Transition erkannt wurde. Es wird also explizit auf Fehler abgestellt, die auftreten, während das System-Management keine Anhaltspunkte für einen kritischen Systemzustand hat. Diese als besonders kritisch zu beurteilende Untermenge der nicht detektierten Fehler tritt erheblich seltener auf als Fehler im Allgemeinen, wie ein Vergleich mit Abbildung 3.4 zeigt. Dennoch liegen die Fehlerhäufigkeiten deutlich zu hoch, als dass dieser Effekt für beliebige Anwendungen vernachlässigt werden könnte.

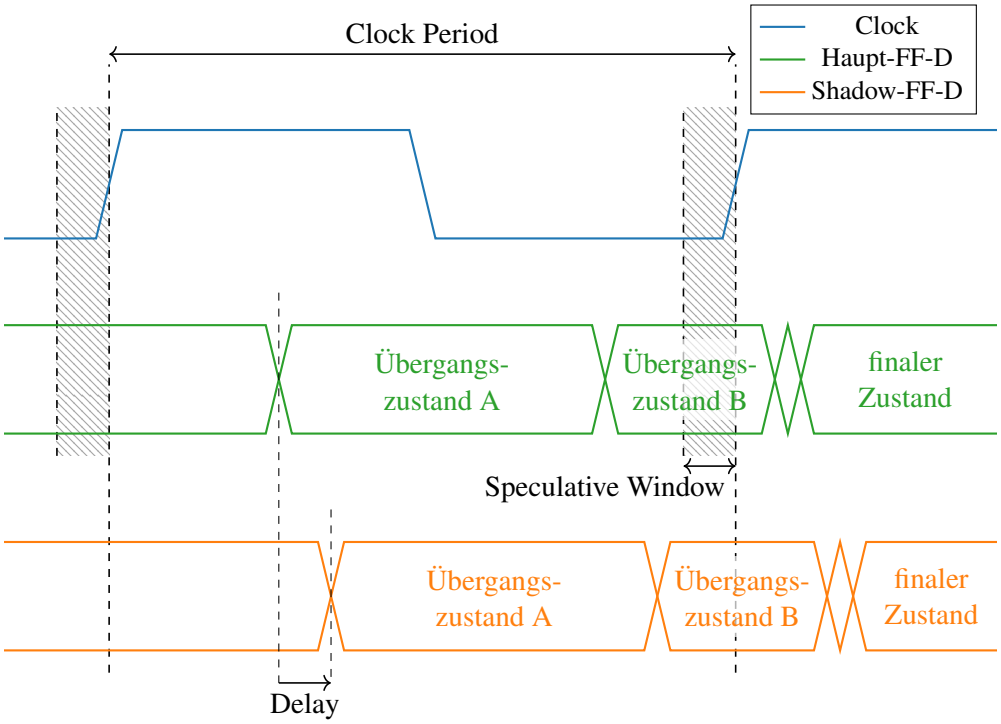


Abbildung 3.5: Beispielhafte Wave Form für einen nicht erkannten Timing-Fehler; In diesem Fall wird statt des korrekten Signalwertes der falsche Wert 'Übergangszustand B' in das Register am Ende des Pfades übernommen, ohne dass das Monitoring-Flipflop eine späte Signaltransition erkennt.

Aufgrund der relativ geringen Auftrittswahrscheinlichkeit und der daraus resultierenden niedrigen absoluten Zahl von nicht detektierten Fehlern in der Simulation ist davon auszugehen, dass die Daten in Abbildung 3.5 einer gewissen zufälligen Streuung unterliegen. Da die Simulation einer deutlich größeren Zahl an Evaluationsläufen aus praktischen Erwägungen – konkret der sehr langen Zeiträume, die für die Simulation benötigt werden – nicht möglich war, wurden die Daten dennoch so in diese Arbeit aufgenommen, da sich deutliche Trends aus den Daten ablesen lassen. Besonders interessant ist, dass die Zahl der nicht detektierten Fehler bei der Verteilung 'WGN-C' am höchsten ist, obwohl diese bezüglich der Gesamtfehleranzahl von anderen Verteilungen deutlich übertroffen wird und vergleichsweise große stationäre Taktperioden erreicht.

3.3.5 Übertragbarkeit der Simulationsergebnisse auf reale Systeme

Natürlich müssen bei einer simulativen Untersuchung immer die Grenzen ihrer Aussagekraft betrachtet werden. Zum einen wurde in dieser Simulation nur eine sehr kleine Schaltung betrachtet. Dies ist erforderlich, da sonst die simulierbare Zeit zu kurz wäre. Schon im aktuellen Setup wurde für alle Verteilungen kumulativ eine Simulationsdauer von über 1300 Stunden beziehungsweise 8 Wochen auf einem aktuellen Desktop-Rechner mit einem Intel-i7-9700 und

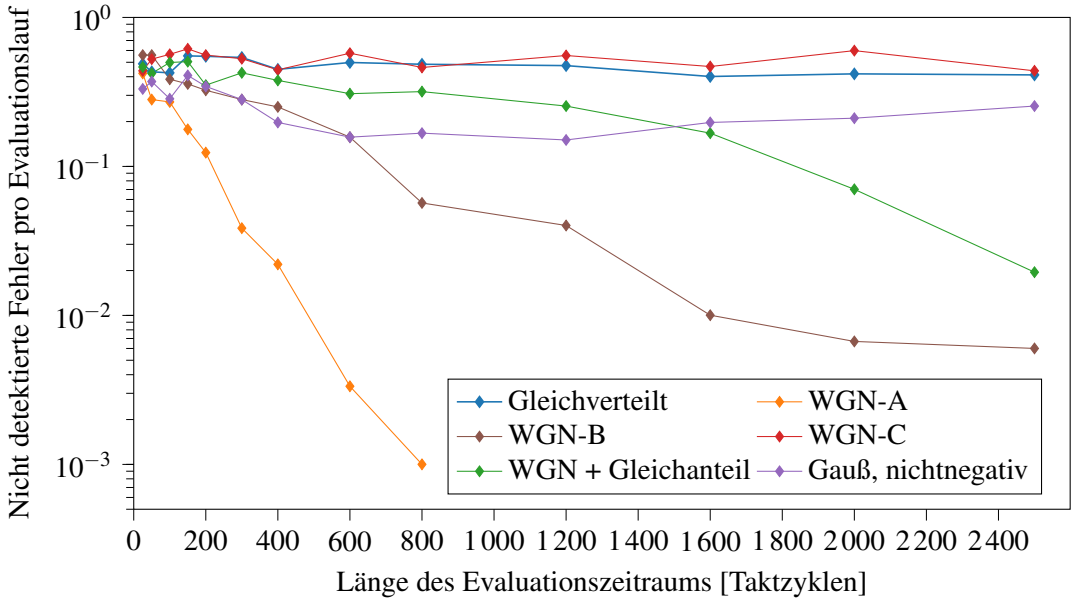


Abbildung 3.6: Nicht detektierte Fehler in Abhängigkeit von der Verteilung der Eingangsvektoren und des Evaluationszeitraums

32GB RAM benötigt. In einem größeren System werden typischerweise viele (nahezu) kritische Pfade überwacht, sodass die Wahrscheinlichkeit, dass in all diesen Pfaden während der Evaluationszeit keine späten Transitionen ausgelöst werden, geringer ist. Andererseits gibt es durchaus Designs, bei denen nur ein Pfad, oder einige wenige Pfade das Timing bestimmen. In diesem Fall ist die hier vorgestellte Simulation sehr gut übertragbar.

Eine zweite wichtige Einschränkung ist die Länge des Evaluationszeitraums. Natürlich könnte durch längere Evaluationszeiträume die Fehlerrate weiter gesenkt werden. Dafür gibt es jedoch zwei Einschränkungen: zum einen beeinträchtigen sehr lange Evaluationszeiträume das dynamische Verhalten einer AVS-Regelschleife. Dies wird in Abschnitt 3.7 ausführlich erörtert. Insbesondere wird die Fähigkeit beeinträchtigt, auf schnelle Änderungen der Schaltgeschwindigkeit zu reagieren. Zum anderen sind sehr lange Evaluationszeiträume nur bei konstanten statistischen Eigenschaften der Daten hilfreich. Verändern sich die Eigenschaften der Daten von einer Verteilung A zu einer Verteilung B, werden auch bei sehr langen Evaluationszeiträumen Fehler auftreten, wenn Verteilung A keine langen Delays verursacht, während solche bei Verteilung B mit einer sehr hohen Wahrscheinlichkeit auftreten.

3.3.6 Schlussfolgerung

Insgesamt ist AVS mittels Monitoring-Flipflops ein sehr interessanter und vielversprechender Ansatz, der unter bestimmten Voraussetzungen geeignet ist, den Energiebedarf eines Systems auf beeindruckende Weise zu senken. In diesem Kapitel wurde jedoch gezeigt, dass prinzipbedingt Fehler auftreten. Dabei besteht eine starke Abhängigkeit der Fehlerraten von schwer zu kontrollierenden Einflussgrößen, wie etwa den statistischen Eigenschaften der Datenströme.

Zudem ist die absolute Fehlerrate sehr groß. Für das untersuchte Setup traten selbst für den längsten untersuchten Evaluationszeitraum je nach Eigenschaften der Eingangsdaten zwischen 10 und 10^5 Fehler pro Sekunde auf. Als Vergleich sollen hier die SER, also die Anzahl der Soft Errors in 1 Milliarde Jahre, für FPGAs in Rechenzentren dienen. Für einen *Stratix V GX A7* wird diese mit 6 200 und für einen *Kintex 7 325T* mit 5 400 angegeben. Dies entspricht einer Mean Time between Failures von 18,3 Jahren beziehungsweise 21,2 Jahren [A 102]. Somit kann der in mehreren Veröffentlichungen erhobene Anspruch, einen sicheren VF-Arbeitspunkt zu ermitteln, der ein fehlerfreies Arbeiten des Systems ermöglicht, in dieser Simulation nicht bestätigt werden.

Es erscheint sogar angemessen, die Einordnung als AVS-Ansatz in Frage zu stellen. Aus Sicht des Autors dieser Arbeit liegt das große Potential dieses Ansatzes vielmehr im Bereich des Voltage Underscalings – also einer Form des Approximate Computings, da in diesem Zusammenhang eine gewisse Fehlerrate bewusst in Kauf genommen wird. Für diese Anwendung müssten dann alle an der Steuerung des Programmflusses beteiligten Komponenten durch strengere Timing Constraints gegen Timing-Fehler geschützt werden, da diese typischerweise nur in der Verarbeitung der Anwendungsdaten akzeptabel sind. Auch für die Anwendungsdaten müssten die Auswirkungen des Voltage Underscalings genauer untersucht werden. Gegebenenfalls müssten beispielsweise die Most Significant Bits von Zahlenwerten ebenfalls mit strengeren Timing-Constraints versehen werden. Die detaillierte Ausarbeitung der nötigen Constraints und Randbedingungen für einen Einsatz von Voltage Underscaling auf Basis von Monitoring-Flipflops erscheint vielversprechend, liegt jedoch außerhalb des Fokus dieser Arbeit und würde ihren Umfang sprengen. Details zum Thema Approximate Computing sowie eigene Konzepte dazu werden in Kapitel 4 dargestellt.

Eine weitere sinnvolle Anwendung findet sich im Bereich des Aging Monitorings durch Überwachung des Delays. In diesem Anwendungsbereich können die Evaluationszeiträume im Vergleich zum hier simulierten Setup im Speziellen, sowie AVS-Anwendungen im Allgemeinen, um mehrere Größenordnungen größer gewählt werden. Eine solche Nutzung von Monitoring-Flipflops für ein Aging Monitoring wird beispielsweise in den Veröffentlichungen [A 103], [A 87], [A 104] und [A 105] untersucht.

3.4 Analyse sensorbasierter AVS-Systeme

Im Folgenden werden auf Online-Ex-Situ-Sensoren basierende AVS-Systeme betrachtet. Nach einer Analyse in diesem Kapitel wird in den folgenden Kapiteln ein, im Rahmen dieser Arbeit entwickeltes, AVS-System vorgestellt. Dabei wird ein besonderes Augenmerk auf das Verhalten des AVS-Systems als Ganzes, also das Zusammenspiel der einzelnen Aspekte und Komponenten sowie seine Integration in den Design-Flow, gelegt.

Aus einer regelungstechnischen Perspektive setzt DVFS eine Steuerung in offener Kette um, wie sie in Abbildung 3.7 dargestellt ist. Im Gegensatz dazu realisiert AVS die, in Abbildung 3.8 dargestellte, geschlossene Regelschleife, um den VF-Arbeitspunkt dynamisch an die zeitlich veränderlichen Störgrößen anzupassen. Einige Ansätze verwenden Sensoren, um den aktuellen Zustand des Systems zu bestimmen, während andere ausschließlich Performance Counter und die üblichen zentralen Temperatursensoren nutzen. Letztere sind relativ unkompliziert und verur-

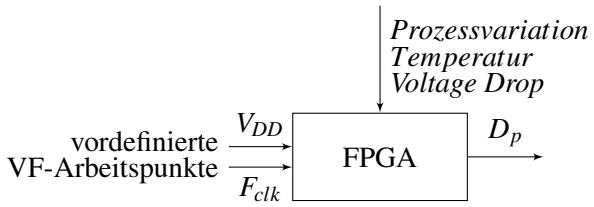


Abbildung 3.7: Steuerung von Taktfrequenz und Versorgungsspannung durch Auswahl aus vordefinierten VF-Arbeitspunkten in einem DVFS-System

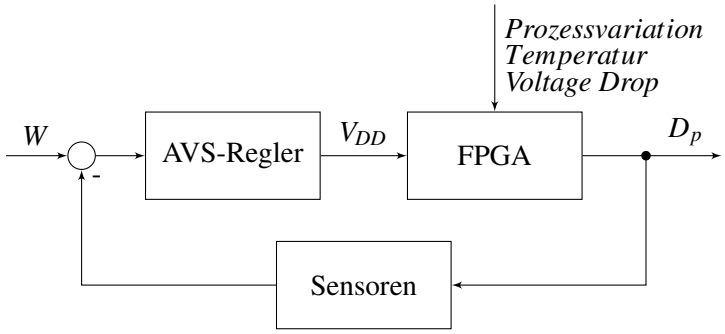


Abbildung 3.8: Blockdiagramm der charakteristischen geschlossenen Regelschleife von AVS-Systemen (abgeändert nach [B 2])

sachen lediglich einen vernachlässigbaren Mehraufwand bezüglich Energie und Flächenbedarf. Darüber hinaus lassen sie sich einfach in den bestehenden EDA-Prozess integrieren. Obwohl sich mit diesen Methoden beeindruckende Ergebnisse erzielen lassen (vgl. [A 106]), können die tatsächlichen Parameter eines VLSI-Systems damit nur in sehr grober Näherung bestimmt werden [A 107]. Zwar ist die Alterung als ein wichtiger Einflussfaktor stark von der Aktivitäts- und Temperaturhistorie abhängig. Die Funktion, die diese Abhängigkeit quantitativ beschreibt, wird jedoch entscheidend von der Prozessvariation während der Herstellung bestimmt. Anders ausgedrückt verursachen gleiche Temperatur- und Aktivitätsverläufe bei verschiedenen (baugleichen) Chips unterschiedliche Parameteränderungen durch Alterung [A 108].

Komplexere AVS-Ansätze berücksichtigen die Prozessvariation und weitere Einflussgrößen, indem sie Delay Lines, ROs oder ähnliche Messvorrichtungen auf den gleichen Die integrieren, auf dem sich auch die Anwendungslogik befindet. Dadurch erfahren die Sensoren ähnliche Prozessvariationen und Temperaturverläufe wie die eigentliche Anwendungslogik. Somit ermöglichen diese Sensoren günstigere VF-Arbeitspunkte als Performance-Counter-basierte Ansätze. Allerdings ist die Wahl eines VF-Arbeitspunkts aufgrund gemessener Sensorwerte keineswegs trivial. Darüber hinaus ist der Zusammenhang zwischen optimalen VF-Arbeitspunkten und Sensordaten nicht statisch, sondern unterliegt zeitlichen Änderungen durch ungleichmäßige Alterung.

Abbildung 3.8 zeigt die Darstellung eines AVS-Systems als regelungstechnisches Blockschaltbild. In dieser Darstellungsweise werden einige entscheidende Fragestellungen deutlich:

a) Wie wird der Sollwert bzw. die Führungsgröße bestimmt?

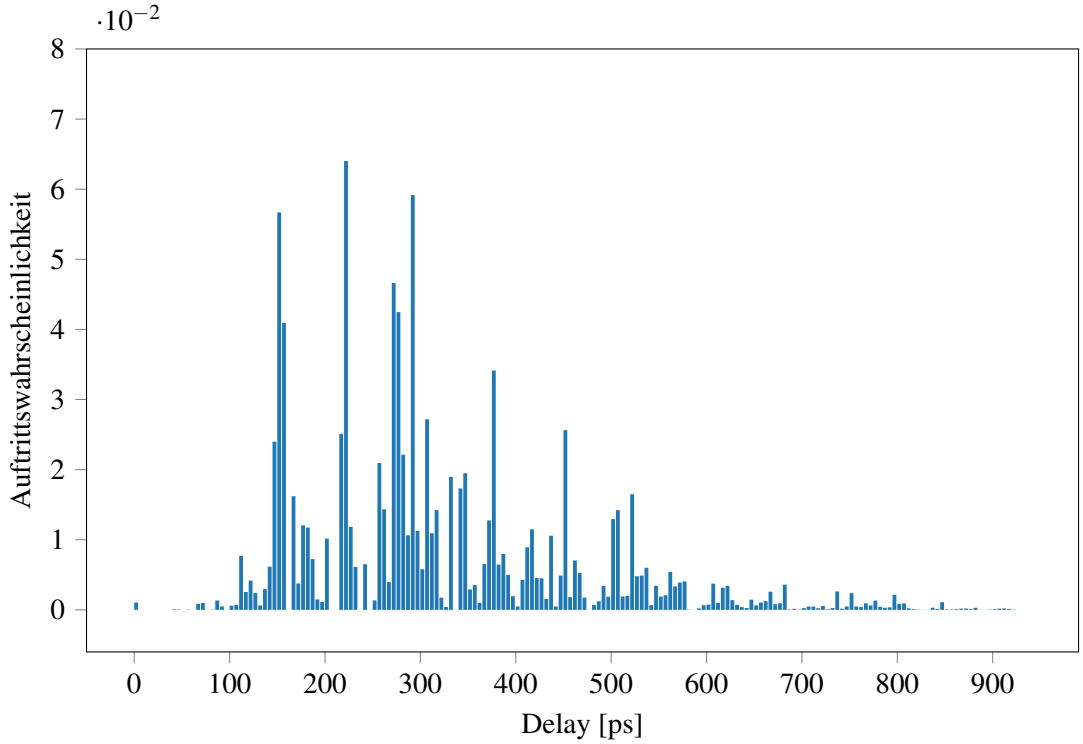


Abbildung 3.9: Verteilung des Delays eines 8-Bit-Ripple-Carry-Addierers bei gleichverteilten, unkorrelierten Eingangsvektoren (in Intervallen zu je 5 ps Breite)

b) Welche Sensortypen sind geeignet?

c) Worauf sollte der Sensor optimiert werden?

d) Welches Verhalten weist die Störgröße auf?

e) Welches dynamische Verhalten zeigt die geschlossene Regelschleife?

Die Frage a) kann mit einer geeigneten Kalibrierung beantwortet werden. Im Rahmen dieser Arbeit wurde eine solche entwickelt. Sie wird in Abschnitt 3.5 vorgestellt. Abschnitt 3.6 behandelt die Fragen b) und c) anhand eines Sensordesigns. Die Auswertung der vorgeschlagenen Sensorrealisierung erfolgt in Unterabschnitt 3.8.4. Die Fragen d) und e) bestimmen, welcher Anteil der Gesamt-Unsicherheit mit einem bestimmten AVS-Ansatz aufgelöst werden kann und welcher Anteil mit einem residualem Guard Band abgesichert werden muss. Sie werden in Abschnitt 3.7 behandelt.

3.5 Kalibrierung von AVS-Sensoren

3.5.1 Anforderungen an eine AVS-Sensorkalibrierung

Viele AVS-Publikationen legen das Hauptaugenmerk auf einen Delay-Sensor, während die Kalibrierung desselben lediglich durch trial-and-error-basierte Verfahren erfolgt. Typischerweise werden diese Trial-and-Error-Kalibrierungen während der normalen Anwendung des Systems

oder mit zufälligen Testvektoren durchgeführt [A 83], [A 109], [A 110]. Dieses Vorgehen ist jedoch problematisch, da das Delay einer kombinatorischen Schaltung erheblich von der Wahl der Eingangsvektoren abhängig ist. Abbildung 3.9 veranschaulicht diese Abhängigkeit des Delays von den Eingangsdaten. Sie zeigt die erschöpfende Simulation des Delays eines 8-Bit-Ripple-Carry-Addierers in Abhängigkeit von den Eingangsvektoren. Die Simulation basiert auf einer 65 nm Bibliothek von STMicroelectronics zur dynamischen Timing-Simulation mit statischen Delay-Werten für jedes Gatter. Somit beruht die gezeigte Schwankung des Delays tatsächlich ausschließlich auf den verschiedenen Eingangsvektoren, da alle weiteren Ursachen für Schwankungen des Delays in der Simulation durch konstante Werte repräsentiert werden.

Wie in Unterabschnitt 2.2.3 beschrieben wird derjenige Pfad zwischen zwei Registern, der in der STA das größte Delay aufweist, als kritischer Pfad bezeichnet. Das Worst-Case-Delay wird nur dann beobachtbar, wenn eine Sequenz von Eingangsvektoren eine Änderung des Signalwertes entlang des gesamten kritischen Pfades verursacht. Wir nennen dies eine Sequenz von Eingangsvektoren, die den kritischen Pfad sensitiviert. Wie in Abbildung 3.9 beispielhaft gezeigt, kann die Auftrittswahrscheinlichkeit von Eingangsvektorsequenzen, die lange Delays verursachen, sehr klein sein. Folglich sind unspezifische Eingangsvektoren im Allgemeinen nicht geeignet, um das Worst-Case-Verhalten bezüglich des Delays zu ermitteln. Dies gilt sowohl für Zufallsvektoren, als auch für Eingangsvektoren, die durch den normalen (Anwendungs-)Betrieb des Systems entstehen. Wenn also eine Kalibrierung darauf beruht, dass die kleinstmögliche Versorgungsspannung ermittelt wird, bei der das System noch korrekt arbeitet, dann müssen geeignete Sequenzen von Testvektoren genutzt werden.

Darüber hinaus gewinnen die Prozessvariationen innerhalb eines Dies, die sogenannten OCVs, in aktuellen Technologiegenerationen zunehmend an Bedeutung [A 32], [A 33]. Ein Online-Ex-Situ-Sensor unterliegt nicht den identischen OCVs wie die eigentliche Anwendungslogik. Dies muss bei seiner Kalibrierung berücksichtigt werden. Selbiges gilt für die Alterung, da diese den Einflüssen von Workload, Temperatur und OCVs unterliegt. Dadurch zeigt das Sensorverhalten bezogen auf das Verhalten der Anwendungslogik eine Drift. Folglich ist es erstrebenswert, eine Kalibrierung über die gesamte Lebenszeit des Systems regelmäßig anzuwenden. Dadurch ist die Verwendung aufwändiger Testausrüstung, wie beim herkömmlichen Frequency Binning, nicht möglich. Die Anwendbarkeit einer Kalibrierung 'vor Ort' ist insbesondere für FPGAs erstrebenswert, da eine Kalibrierung, die das konkrete Design berücksichtigt, häufig günstigere VF-Arbeitspunkte liefern kann. Für die praktische Anwendbarkeit ist es darüber hinaus entscheidend, dass die Methodik in bestehende EDA-Abläufe integriert werden kann.

Somit soll eine Kalibrierung von Online-Sensoren für AVS auf FPGAs folgende Anforderungen erfüllen:

- OCVs erfassen.
- Die Alterung berücksichtigen, insbesondere auch die Alterung in Abhängigkeit von OCVs und in der Vergangenheit auf dem FPGA betriebenen Designs.
- Die Abhängigkeit des Delays von den Eingangsdaten beachten.
- Die Besonderheiten des konkreten Anwendungsdesigns berücksichtigen.

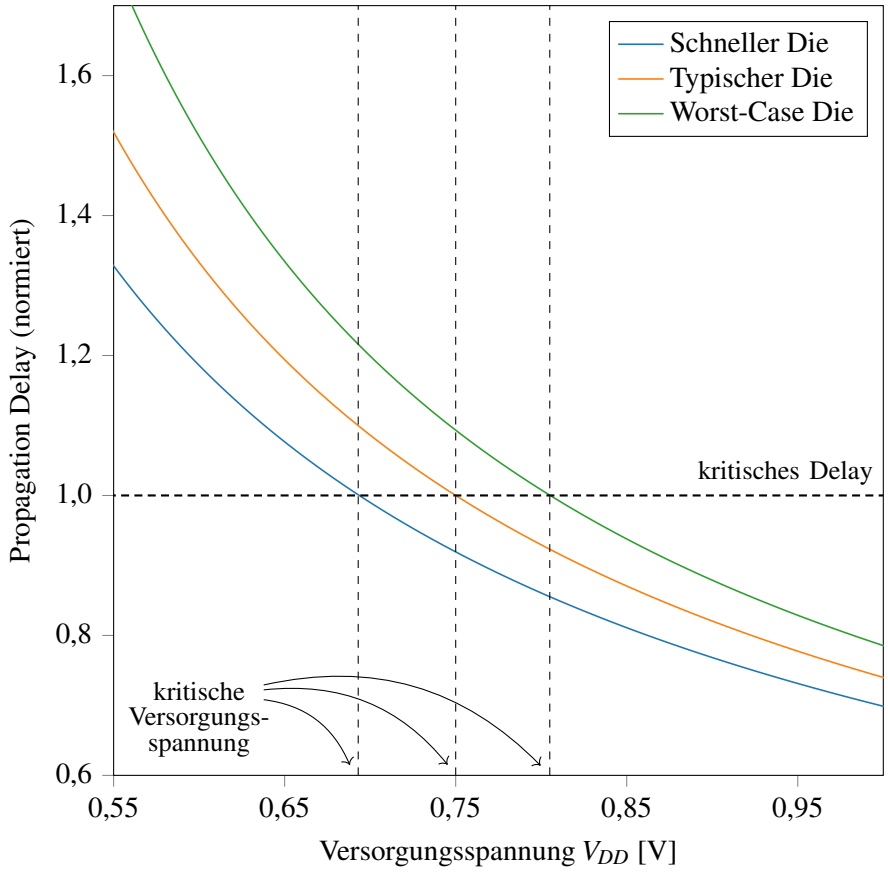


Abbildung 3.10: Kritische Versorgungsspannung verschiedener Dies (Abbildung nach [B 1]; Original ©2019 IEEE)

- Ein ausreichend dimensioniertes residuales Guard Band bestimmen, um alle Parameterschwankungen abzusichern, die nicht durch das konkrete AVS-System ausgeregelt werden können.
- Sie sollte mit der, im produktiven Einsatz vorhandenen, Peripherie umsetzbar sein. Dies betrifft insbesondere aufwändige externe Sensorik und die, in der Literatur häufig verwendete, Spannungsversorgung durch hochpräzise Laborgeräte. Kosten durch zusätzliche Anforderungen an die Peripherie müssen berücksichtigt werden.
- Sie sollte sich möglichst gut in bestehende EDA-Arbeitsabläufe integrieren lassen.

3.5.2 Konzeption einer Kalibrierungsmethode für AVS-Sensoren

Um die in Unterabschnitt 3.5.1 dargelegten Anforderungen an ein AVS-System zu erfüllen, wird im Rahmen dieser Arbeit eine geeignete Offline-Kalibrierungsprozedur vorgeschlagen. Diese basiert auf einer Messung des Delays der Anwendungslogik bei reduzierter Versorgungsspannung. Dazu wird die streng monotone Beziehung zwischen Versorgungsspannung und Delay der Complementary Metal–Oxide–Semiconductor (CMOS)-Logik genutzt. Wird die Versorgungs-

spannung unter Beibehaltung einer konstanten Taktfrequenz immer weiter gesenkt, verletzt das zunehmende Delay ab einem bestimmten Spannungsniveau die Setup Time des nachfolgenden Registers. Dies führt potentiell zu Fehlern – insbesondere falschen Dateninhalten von Registern und Metastabilität. Allerdings ist ein Absenken der Versorgungsspannung in einem gewissem Maß möglich, da zunächst lediglich die pessimistisch gewählten Guard Bands verringert werden. Das größtmögliche Delay, das gerade noch keine Fehler verursacht, soll *kritisches Delay* genannt werden. Die Versorgungsspannung, bei der der kritische Pfad einer Schaltung gerade das kritische Delay aufweist, ist theoretisch der optimale VF-Arbeitspunkt. Sie wird im Folgenden *kritische Versorgungsspannung* genannt. Verschiedene Instanzen ein und desselben Logikpfades zeigen aufgrund von Prozessvariabilität und Alterung bei gleicher Versorgungsspannung verschiedene Delays. Dadurch erreichen sie das kritische Delay bei verschiedenen Versorgungsspannungen. Dies gilt unabhängig davon, ob sich diese Instanzen des Pfades auf dem gleichen oder zwei verschiedenen Dies befinden. Abbildung 3.10 veranschaulicht diesen Sachverhalt für identische Pfade auf verschiedenen Dies.

Ein wichtiger Aspekt ist, dass es nicht genügt, nur den einen, in der STA ermittelten kritischen Pfad zu betrachten. Durch OCVs, Alterung sowie Versorgungsspannungs- und Temperaturgradienten kann in der Realität ein anderer Pfad tatsächlich kritisch sein [A 53]. Folglich müssen zur korrekten Kalibrierung mehrere Timing-Pfade berücksichtigt werden. Die Wahl der zu untersuchenden Pfade liegt außerhalb des Themenfeldes der vorliegenden Arbeit, wurde aber bereits ausführlich untersucht – beispielsweise in [A 111] und speziell für FPGAs in [A 112], [A 113] und [A 114]. In dieser Arbeit wird ein unkompliziert umsetzbares, sehr robustes Verfahren aus [A 104] genutzt: es werden alle Pfade berücksichtigt, deren Timing-Slack kleiner als ein bestimmter Anteil der Taktperiode ist. Prinzipiell lässt sich das Konzept dieser Arbeit aber mit beliebigen Verfahren zur Pfadauswahl kombinieren.

Das im Rahmen dieser Arbeit entwickelte Kalibrierungsverfahren für AVS-Sensoren bestimmt die kritische Versorgungsspannung $V_{DD,krit}$ mittels geeigneter Built-in Tests (BITs), die bei einer stufenweise verringerten Versorgungsspannung durchgeführt werden. Die kritische Versorgungsspannung liegt dann zwischen der Versorgungsspannung, bei der der letzte erfolgreiche BIT durchgeführt wurde und der ersten, bei der dieser ein fehlerhaftes Verhalten der Logik aufdeckt. Wie in Unterabschnitt 3.5.1 erläutert, sind Zufallsvektoren oder Anwendungsdaten nicht geeignet, um das Worst-Case-Delay eines Pfades anzuregen. Deshalb werden dedizierte *Path Delay Fault Test Patterns* verwendet. Mit ihnen können zielgerichtet die kritischen Pfade des Designs sensitiviert werden, was eine valide Aussage darüber zulässt, ob die kritische Versorgungsspannung für die untersuchten Bedingungen unterschritten wurde. Zu diesen Bedingungen gehören insbesondere das konkrete Exemplar des FPGA mit seiner individuellen Prozessvariation und Alterung, das Design beziehungsweise Bitfile, mit dem das FPGA konfiguriert wurde und die Temperatur. Die Path Delay Fault Test Patterns können für ASICs mit kommerziellen EDA-Tools erzeugt werden. Für FPGA sind dafür Anpassungen nötig, die in Unterabschnitt 3.8.5 beschrieben werden

Die kritische Versorgungsspannung, mithin der theoretisch optimale VF-Arbeitspunkt, ist kein statischer Wert. Vielmehr unterliegt er signifikanten Fluktuationen durch den Einfluss vielfältiger Störgrößen, wie Temperaturänderungen, reversiblen Änderungen der Transistorkennlinien durch Effekte wie BTI, sowie durch die Anwendung verursachten Voltage Drops. Die dynami-

sche Änderung des optimalen VF-Arbeitspunktes wird durch die Online-Sensoren erfasst und bei der Regelung der Versorgungsspannung entsprechend berücksichtigt.

Die kritische Versorgungsspannung wird mit der beschriebenen Offline-Messung ermittelt. Während dieser Prozedur werden, sobald die kritische Versorgungsspannung erreicht ist, aktuelle Werte der Online-Sensoren ausgelesen. Würden diese Werte nun als Stellgröße einer AVS-Regelschleife verwendet, würde das System theoretisch immer genau bei der (veränderlichen) optimalen Versorgungsspannung betrieben werden. In einem realen System ist dies jedoch nicht möglich, da sich die Änderung des Delays der Sensoren und der Anwendungslogik unter dem Einfluss der Störgrößen zwar gleichartig, jedoch niemals völlig identisch verhält. Zudem wirken bestimmte Störgrößen zu schnell, um ausgeregelt zu werden. Auch die Regelung selbst verursacht eine gewisse Abweichung vom Sollwert, beziehungsweise kann diese nicht völlig verhindern. Deshalb wird zu den bei der kritischen Versorgungsspannung ausgelesenen Sensorwerten noch ein Guard Band, das Fast Fluctuation Guard Band (FFGB), addiert. Die Summe aus ausgelesenen Sensorwerten und Fast Fluctuation Guard Band (FFGB) wird dann als Stellgröße der Regelschleife verwendet. Die Bestandteile und die Methodik zur quantitativen Bestimmung des FFGB werden in Abschnitt 3.7 erläutert.

Da Störgrößen nicht unabhängig von der Position auf der Chipoberfläche sind, sondern sich auch innerhalb eines Dies signifikant unterscheiden, wird ein Array von Sensoren verwendet um diese entsprechend zu erfassen. Jeder Sensor repräsentiert gewissermaßen einen bestimmten Flächenanteil des Dies. Jeder dieser Sensoren muss kalibriert werden. Somit ergibt sich als Sollgröße kein einzelner Wert, sondern ein Array von Werten. Die, je nach Position auf dem Die unterschiedlichen, Störgrößen bewirken im Betrieb des Systems, dass die verschiedenen Sensoren verschieden stark von den Sollwerten abweichende Messwerte zurückliefern. Eine harte Randbedingung für die Regelung des AVS-Systems ist jedoch, dass Anwendungslogik auf jeder Teilfläche des Dies korrekt arbeiten muss. Deshalb wird für alle Sensorwerte die Abweichung zum jeweils zugehörigen Sollwert berechnet. Aus diesem Array von Werten wird dann der kleinste ermittelt. Das ist der Wert desjenigen Sensors, dessen Kalibrierungs-Sensorwert am weitesten unterschritten bzw. am wenigsten weit überschritten wurde. Dieser repräsentiert die Region auf dem Chip, deren Delay bei einer Spannungsabsenkung mit ansonsten gleichbleibenden Bedingungen als erstes kritisch werden würde. Nur dieser Sensorwert wird dann als Regelabweichung an den Regler weitergeleitet.

Die lokalen Effekte und Gradienten werden jedoch nicht nur durch Process, Voltage and Temperature (PVT)-Variationen verursacht. Vielmehr weist, im Allgemeinen, auch die zu realisierende Schaltung ortsabhängige Besonderheiten auf. Es kann beispielsweise nicht davon ausgegangen werden, dass kritische Pfade gleichmäßig über die gesamte Chipfläche verteilt sind. Um dieses lokal spezifische Verhalten zu reflektieren, wird das FPGA in eine Anzahl von Regionen eingeteilt. In jeder dieser Regionen wird ein Online-Sensor platziert. Während der Kalibrierung wird eine Vielzahl von potentiell kritischen Pfaden getestet. Die einzelnen Pfade werden den Sensorregionen zugeordnet, wobei jeder Pfad allen Regionen zugeordnet wird, innerhalb derer Elemente des Pfades liegen. Wird während der Kalibrierung durch einen Path Delay Test ein Timing-Fehler eines Pfades festgestellt, wird die Versorgungsspannung für alle Regionen, denen der Pfad zugeordnet ist, als zu niedrig markiert. Für die Regionen, bei denen kein Pfad bei einer höheren Versorgungsspannung einen Timing-Fehler verursacht hat, liegt die kritische

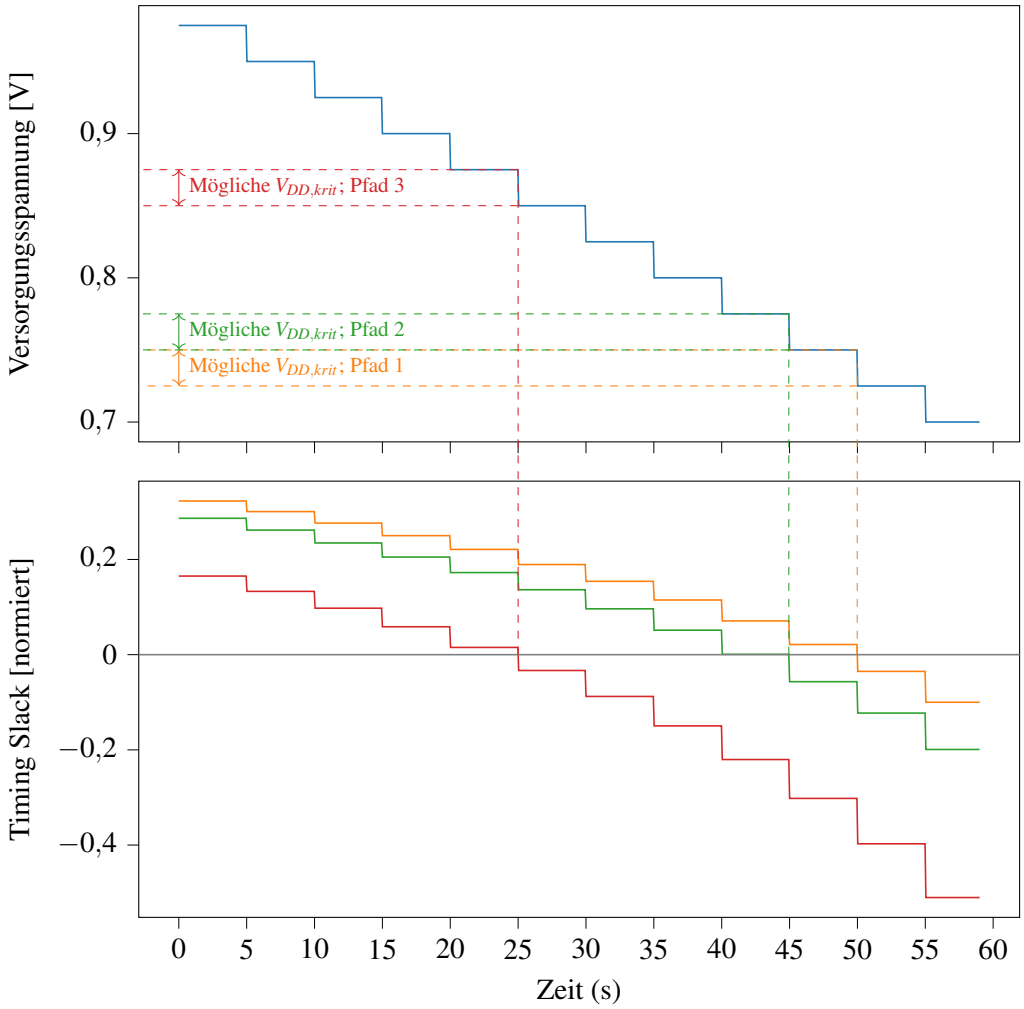


Abbildung 3.11: Begrenzte Auflösung der Kalibrierung durch Schrittweite der Spannungssenkung. Hier beispielhaft dargestellt für drei verschiedene Pfade eines Designs

Versorgungsspannung somit zwischen der als zu niedrig erkannten und der nächsthöheren Versorgungsspannung. Abbildung 3.11 veranschaulicht dies für drei verschiedene Pfade.

FPGAs bilden das Anwendungsdesign auf LUTs und ein konfigurierbares Routing ab. Sowohl die LUTs als auch die Konfiguration des Routings werden dabei in SRAM-Zellen gespeichert. SRAM benötigt eine gewisse Mindestversorgungsspannung, um zuverlässig zu funktionieren. Wird diese sogenannte *Data Retention Voltage* unterschritten, kann der Speicherinhalt verloren gehen. Daraus ergibt sich, dass die Berücksichtigung der Data Retention Voltage eine harte Randbedingung für die Anwendung von AVS auf FPGAs ist. Diese darf nicht unterschritten werden, da sonst nicht sichergestellt ist, dass die SRAM-Zellen die Konfiguration des FPGA zuverlässig halten. Eine mögliche Veränderung der Konfiguration des FPGA kann jedoch offensichtlich zu vollständig undefiniertem Verhalten des Designs führen und ist somit unbedingt zu vermeiden. Insbesondere gilt es, den Unterschied zu einer Verletzung des Timings heraus-

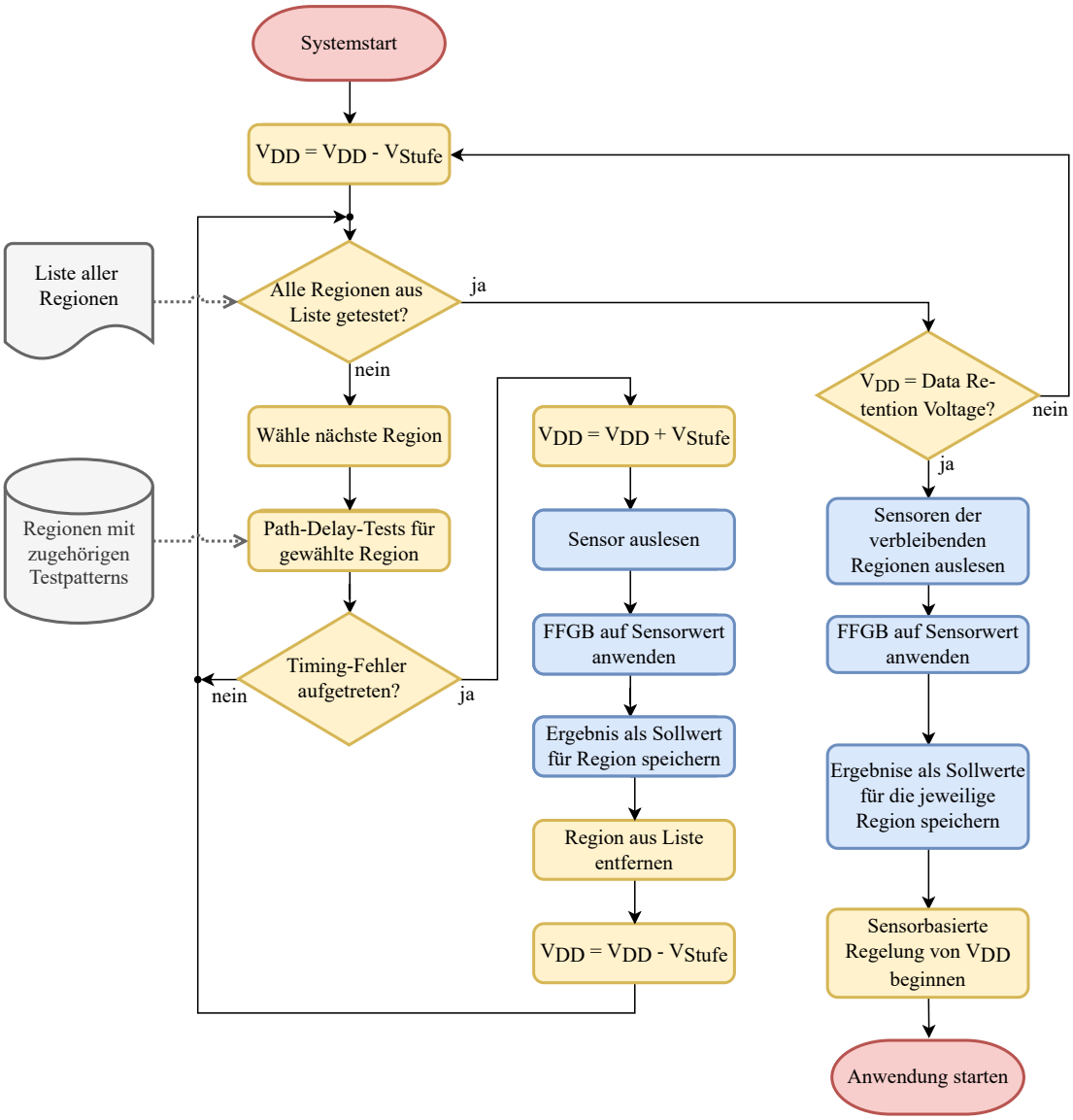


Abbildung 3.12: Ablauf der Sensorkalibrierung

zustellen. Diese wird in der Kalibrierungsphase bewusst in Kauf genommen und ist durch die Anhebung der Versorgungsspannung vor dem produktivem Einsatz des Designs vollständig reversibel. Eine potentielle Korrumpierung der Konfiguration des FPGAs würde hingegen auch nach einer Anhebung der Versorgungsspannung ein undefiniertes Verhalten des Designs verursachen. Deshalb wird die Versorgungsspannung während der Kalibrierungsprozedur höchstens bis zur Data Retention Voltage abgesenkt. Selbiges gilt natürlich auch für den eigentlichen Anwendungsbetrieb des AVS-Systems. Der Gesamtablauf der Kalibrierungsprozedur ist in Abbildung 3.12 dargestellt.

Wird für eine Region beim Absenken der Versorgungsspannung bis auf die Data Retention Voltage durch die Path Delay Tests kein Timing-Fehler erkannt, wird die Data Retention Voltage als kritische Versorgungsspannung angenommen. Es werden dann die bei dieser Spannung ausgelesenen Sensorwerte zuzüglich des FFGB als Sollwerte verwendet. Dies ist notwendig, da Pfade in dieser Region potentiell bei einer nur minimal kleineren Versorgungsspannung ein korrumpiertes Timing aufweisen könnten. Das Timing solcher Pfade könnte dann bei laufender Anwendung durch entsprechende Störgrößen tatsächlich korrumpiert werden, was es zu vermeiden gilt.

Wie bereits in Abschnitt 2.5 erläutert, ist das Timing eines Pfades nicht nur von der kombinatorischen Logik abhängig. Diese wurde lediglich aufgrund ihrer Anschaulichkeit in den Vordergrund gestellt. Abbildung 2.9 aus Abschnitt 2.5 veranschaulicht, dass das Timing eines Pfades immer auch vom Clock-Netzwerk abhängig ist. Aus Sicht des Registers am Ende eines Pfades ist für das korrekte Timing entscheidend, dass das Signal am D-Port seinen stabilen Endzustand für eine gewisse Zeit eingenommen hat, bevor es von der Clock-Flanke erreicht wird. Somit ist nicht nur die kombinatorische Logik im Launch Path entscheidend, sondern auch die relative Abweichung des Zeitverhaltens von Launch und Capture Path. Beispielsweise ist die Kombination aus einem 'langsamen' Launch Path und einem 'schnellen' Capture Path besonders ungünstig hinsichtlich der Setup Time. Dieser Sachverhalt gewinnt in modernen CMOS-Prozessen an Bedeutung und kann nicht mehr vernachlässigt werden. Da das Clock-Netzwerk aus den gleichen Strukturen aufgebaut ist (nämlich Transistoren, Leiterbahnen etc.), unterliegt es den gleichen Störgrößen wie auch die kombinatorische Logik. Allerdings haben diese eine andere Wirkung. Eine (relativ zum Systemtakt langsame) Störgröße wirkt auf alle Bestandteile des Clock-Netzwerkes gleich. Somit wird lediglich das Taktsignal an allen Registern zeitlich verschoben. Eine örtlich verschiedene oder zeitlich sehr schnelle Änderung kann jedoch dazu führen, dass die Clock-Flanken an den Registern an Beginn und Ende eines (kritischen) Pfades unterschiedlich beeinflusst werden und somit das Timing korrumpieren. Somit kann der Einfluss des Clock-Netzwerks nicht vernachlässigt werden und muss bei Ansätzen, die ihn nicht explizit behandeln, in Form eines größeren residualen Guard Bands berücksichtigt werden.

In der in diesem Kapitel vorgeschlagenen Kalibrierungsprozedur werden unverändert die identischen Register- und Clock-Tree-Bestandteile verwendet, die auch während des Anwendungseinsatzes des Designs genutzt werden. Somit werden Variationen des Clock Tree mit erfasst und in der Kalibrierung berücksichtigt. Für viele Offline-In-Situ-Messverfahren aus dem in Unterabschnitt 3.2.2 vorgestellten Stand der Technik trifft dies nicht zu, da sie lediglich auf das Verhalten der Kombinatorik im kritischen Pfad abstellen. Ein wesentlicher Vorteil der in diesem Kapitel vorgestellten Kalibrierungsmethodik ist somit, dass sie auch das Verhalten des Clock-Netzwerkes berücksichtigt.

3.6 Online-Sensoren

Die im vorhergehenden Unterabschnitt 3.5.2 vorgestellte Kalibrierungsprozedur kann prinzipiell für beliebige Online-Ex-Situ-Sensoren genutzt werden. Dies gilt insbesondere für die in Unterabschnitt 3.2.3 vorgestellten Sensoren aus der Literatur. Im Rahmen dieser Arbeit werden beispielhaft Ringoszillator (RO)-basierte Sensoren verwendet.

Unter einem RO versteht man eine ringförmig geschlossene Kette von Invertern. Durch die Wahl einer ungeraden Anzahl an Invertern wird diese immer wieder von einer Signalflanke durchlaufen. Die Anzahl der Signalflankendurchläufe innerhalb einer Referenzzeit wird mittels eines Counters festgestellt und steht für die beobachtete Schaltgeschwindigkeit. Durch ein zusätzliches `AND`-Gatter, das zwischen zwei Invertern eingefügt wird, kann der RO an- und abgeschaltet werden, um Eigenerhitzung und übermäßige Alterung zu vermeiden. ROs sind ein bekanntes, weit verbreitetes Sensorkonzept. Für die Verwendung auf FPGA sind sie besonders geeignet, da sie keine Analogkomponenten, spezielle Transistorschaltungen oder ähnliches benötigen. Vielmehr sind sie aus Standard-Logikelementen aufgebaut, wie sie auf FPGAs vorhanden sind. Folglich wurde ihre Verwendung als Sensor auf FPGAs in der Literatur vielfach vorgeschlagen – etwa zur Messung des Delays [A 115], [A 116], [A 117], [A 118], [A 74], der statischen Verlustleistung [A 116], der dynamischen Verlustleistung [A 116], der Temperatur [A 119], [A 116] und von Cross Talk [A 119].

Obwohl ROs ein lang bekanntes und weit verbreitetes Sensorkonzept sind, wurde ihre Verwendung auf FPGAs als Online-Ex-Situ-Sensoren für eine geschlossene AVS-Regelschleife bisher nach bestem Wissen des Autors dieser Arbeit nicht näher untersucht. ROs bieten für diesen Einsatz jedoch erhebliche Vorteile. So wurde in der Literatur gezeigt, dass sie sehr gut geeignet sind, um die Schaltgeschwindigkeit einer Anwendungsschaltung vorherzusagen [A 39], [A 74]. Zudem benötigen sie nur wenige Ressourcen. Sie basieren auf Standard-Logikzellen, wie sie auf FPGAs in großer Zahl verfügbar sind. Insbesondere werden keine zusätzlichen, phasenverschobenen Taktsignale benötigt, deren Erzeugung und Verteilung aufwändig bezüglich verfügbaren Ressourcen und Energie wäre (vergleiche dazu Unterabschnitt 3.2.4). In diesem Kapitel werden wesentliche Designentscheidungen für die Anwendung von ROs für ein FPGA-basiertes AVS-System mit der in Unterabschnitt 3.5.2 vorgestellten Kalibrierung diskutiert.

In der Literatur werden RO-basierte Sensoren typischerweise als *Hard Macro* oder per *Directed Routing* realisiert – also mit stets gleichem relativem Placement und Routing [A 74], [A 119], [A 118], [A 117], [A 115], [A 116]. Zudem wird auf eine möglichst hohe zeitliche und/oder räumliche Auflösung abgestellt [A 74], [A 118], [A 117]. Nachfolgend wird für das konkrete Anwendungsszenario eine andere Vorgehensweise vorgeschlagen und diskutiert.

Bezüglich der **räumlichen und zeitlichen Auflösung** gelten folgende Überlegungen: Wie in Unterabschnitt 3.5.2 beschrieben, wird für das hier vorgestellte AVS-System eine begrenzte Anzahl von Regionen durch je einen Sensor repräsentiert. Eine räumliche Auflösung, die feingliedriger ist als die für die Regelung der Versorgungsspannung genutzte Einteilung in Regionen, erbringt somit keinen zusätzlichen Nutzen. Vielmehr ist es wünschenswert, dass der Sensor die tatsächlichen Gegebenheiten in der Region möglichst repräsentativ wiedergibt. Auch eine sehr hohe zeitliche Auflösung ist im Rahmen des hier betrachteten Anwendungsszenarios nicht vorteilhaft. Wie im nachfolgenden Abschnitt 3.7 näher erläutert wird, ist die Fähigkeit des AVS-Systems, sehr schnelle Störgrößen auszuregeln, ohnehin begrenzt. Somit ist es wünschenswert, einen über einen gewissen Zeitraum gemittelten Messwert zu erhalten.

Die **Realisierung des Sensors als Hard Macro** wiederum dient der Reproduzierbarkeit. Wobei der Begriff Hard Macro hier auch stellvertretend für alle anderen Möglichkeiten eines exakt reproduzierbaren Placements und Routings verwendet wird. Ohne Hard Macro oder vergleichbare Constraints bilden die EDA-Tools die ROs bei jeder Instanziierung auf abweichende Logik-

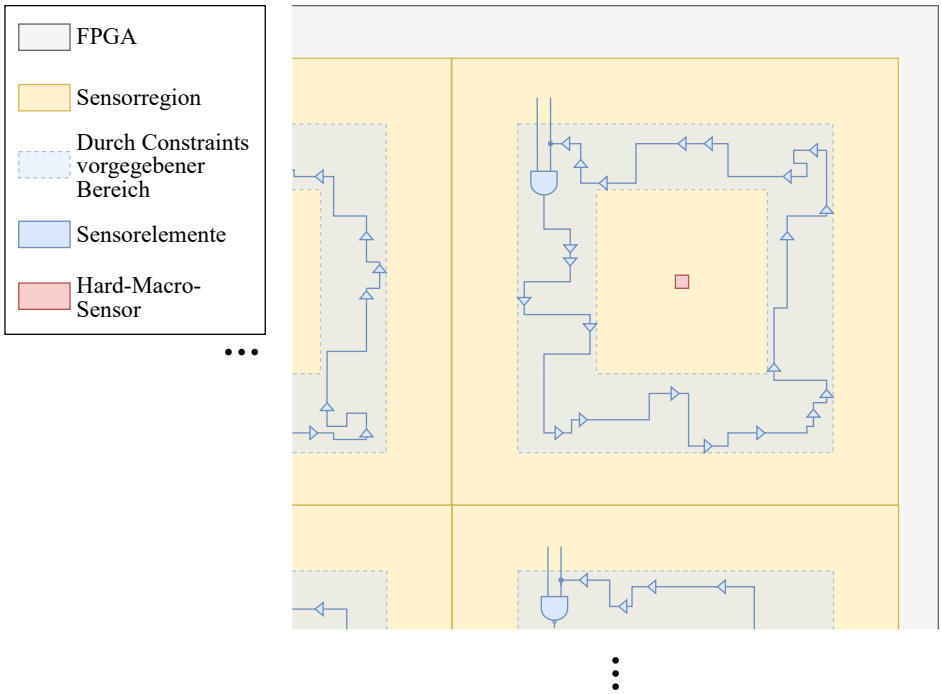


Abbildung 3.13: Schematischer Vergleich der Sensorplatzierung mittels Hard Macro und der in dieser Arbeit vorgeschlagenen freien Platzierung innerhalb eines Bereichs

und Routing-Ressourcen ab. Dies führt zu einer Streuung im Sensorverhalten. Die Realisierung als Hard Macro dagegen erzwingt ein immer gleiches Placement und Routing des ROs. Dies dient dem Zweck, dass mehrere Instanzen des Sensors ein möglichst gleiches Verhalten zeigen. Dadurch sollen in Voruntersuchungen ermittelte Sensorkennlinien möglichst auf alle Sensoren an unterschiedlichen Orten auf dem FPGA, beziehungsweise auf verschiedenen FPGAs gleichen Typs, angewendet werden können. Die Realisierung als Hard Macro bringt jedoch auch erhebliche Nachteile mit sich. So lassen sich Hard Macros nicht an beliebiger Stelle auf dem FPGA platzieren, da sie auf eine ganz bestimmte Kombination von Routing-Ressourcen angewiesen sind. FPGAs weisen zwar prinzipiell eine regelmäßige Struktur auf, diese wird jedoch durch funktionale Blöcke wie Digital Signal Processing (DSP)-Slices, BRAM und weitere unterbrochen und auch die verfügbaren Routing-Ressourcen weisen ortsabhängige Unterschiede auf. Zudem sind die vom Hard Macro belegten Ressourcen für das Anwendungsdesign gesperrt. Dies kann zu erheblichen Problemen führen – insbesondere wenn Teile des Anwendungsdesigns ebenfalls als Hard Macro IP Core vorliegen. Aber auch bei einem aus einer Register Transfer Level (RTL)-Beschreibung synthetisierten Design können im ungünstigen Fall Routing Congestions auftreten, die das Timing Closure erschweren oder sogar unmöglich machen. Um die genannten Nachteile zu minimieren, werden Hard-Macro-Sensoren typischerweise möglichst auf eine kleine Fläche und möglichst wenig Ressourcen konzentriert. Auf die Verwendung von Ressourcen des globalen Routings – also des Routings außerhalb von Configurable Logic Blocks (CLBs) – wird oft komplett verzichtet. Dies passt außerdem zu den häufig angewandten Optimierungszielen einer hohen räumlichen und zeitlichen Messauflösung.

In der vorliegenden Arbeit wurde ein anderer Ansatz gewählt. Die Aufgabe der Sensoren ist es nicht, eine absolute physikalische Größe, wie etwa eine Temperatur oder Spannung, zu messen. Vielmehr soll die Änderung des Delays des kritischen Pfades ermittelt werden, auf das die Sensoren ohnehin kalibriert werden. Deshalb wird auf eine Platzierung der Sensoren als Hard Macro verzichtet. Stattdessen wird lediglich per Constraint eine Platzierung der einzelnen Sensorelemente innerhalb bestimmter Bereiche vorgeben. Die Synthese des Sensors kann dann gemeinsam mit dem Anwendungsdesign erfolgen, sodass das EDA-Tool die Freiheit hat, bestimmte, vom Anwendungsdesign benötigte, Ressourcen für dieses zu verwenden. Eine vollständige Auslastung aller Ressourcen ist in der Praxis ohnehin kaum erreichbar. Der Sensor kann also in den ungenutzt bleibenden Ressourcen des FPGA realisiert werden. Da der Sensor nun nicht mehr ganz bestimmte Ressourcen hart blockiert, ist es möglich, einen großflächigeren Sensor mit einer größeren Kettenlänge des RO zu wählen. Dies ist von großem Vorteil, da unkorrelierte OCVs oder eine ortsspezifische Alterung (beispielsweise aufgrund eines in der Vergangenheit auf dem FPGA betriebenen Designs [A 52]) einiger weniger, lokal konzentrierter Elemente das Sensorverhalten nicht mehr so stark beeinflussen können. Über geeignete `LOC...RANGE`-Constraints wird der Sensor so platziert, dass er die Sensorregion möglichst gut abdeckt. Konkret wird dafür eine ringförmige Fläche vorgesehen. Da über `LOC...RANGE`-Constraints nur rechteckige Bereiche definiert werden können, wird dieser Ring aus acht Teilbereichen zusammengesetzt, wobei jedem dieser Teilbereiche bestimmte Sensorelemente zugeordnet sind. Dadurch wird auch sichergestellt, dass der RO tatsächlich ringförmig den gewünschten Teilbereich der Sensorregion umfasst. Abbildung 3.13 illustriert die hier vorgeschlagene Sensorplatzierung im Vergleich zu einem zentralen Hard-Macro-Sensor.

Diese Platzierung des ROs über ein größeres Gebiet führt zudem dazu, dass der Anteil des Routings am Delay anteilsmäßig zunimmt. Routing-Ressourcen werden jedoch anders von PVT-Variationen beeinflusst als Logikelemente [A 80], [A 58]. Somit ist es erstrebenswert, dass die Sensoren möglichst einen realistischen Anteil an Routing Delay enthalten. Da kritische Pfade typischerweise signifikante Anteile an Routing Delay beinhalten, ist die in dieser Arbeit gewählte Realisierung von ROs diesbezüglich repräsentativer. Auf eine exakte Replikation des Anteils des Routing-Delays wird jedoch verzichtet. Die Gründe dafür sind die gleichen, die auch ganz allgemein gegen Critical Path Replicas sprechen: Zum einen kann ein Critical Path Replica stets nur einen bestimmten Pfad replizieren. Wie bereits erläutert sind jedoch typischerweise viele Pfade kritisch, oder nahezu kritisch. Es ist folglich einerseits nicht möglich, zu bestimmen, welcher einzelne Pfad unter dem Einfluss von PVT-Variationen und Alterung unter Einsatzbedingungen tatsächlich kritisch ist. Andererseits weicht das Verhalten einer isolierten Replikation eines Pfades deutlich von dem des eigentlichen kritischen Pfades ab, da dieser durch die Einbindung in ein Logiknetzwerk andere Fanouts aufweist. Durch die Unterschiede im Fanout ergibt sich ein abweichendes Delay. Zhao et al. geben in [A 81] für die dort untersuchten Beispiele eine Abweichung von bis zu 6 % an. Es ist davon auszugehen, dass diese in ungünstigen Fällen noch größer ausfallen kann. Deshalb wird in dieser Arbeit der Sensor unabhängig vom Anwendungsdesign mit repräsentativen Elementen aufgebaut.

Die Möglichkeit, aufgrund der zyklisch durchgeführten Kalibrierung tatsächlich auf die Sensorplatzierung mittels Hard Macro zu verzichten, wird in Unterabschnitt 3.8.4 experimentell untersucht. Dies ist notwendig, da die reine Verwendung in einem AVS-System eine isolierte Bewertung des Einflusses der Sensorplatzierung nur sehr begrenzt zulässt.

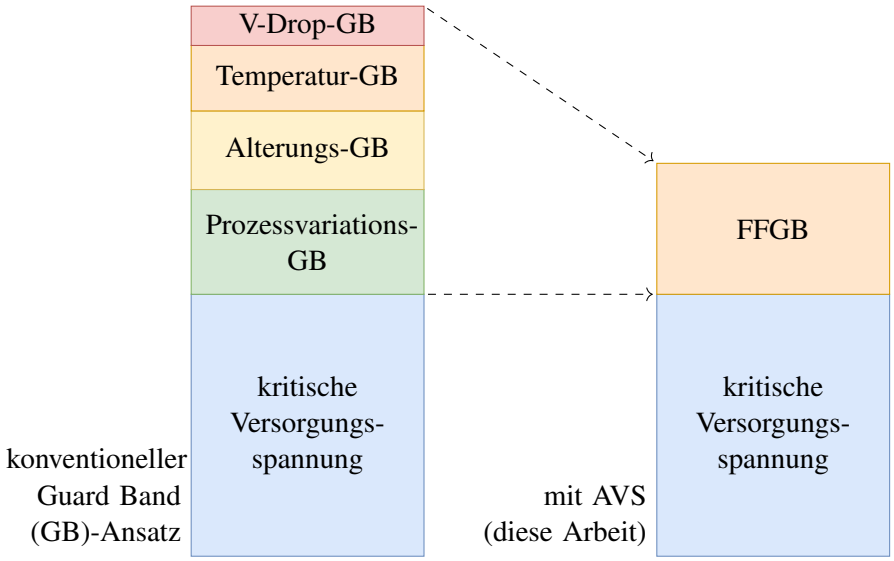


Abbildung 3.14: Partielle Guard-Bands mit und ohne AVS (Abbildung übersetzt aus [B 2])

3.7 Bemessung des residualen Guard Bands

Das im Standard-VLSI-Designprozess angewandte Guard Band ist eigentlich die Summe mehrerer partieller Guard Bands, die jeweils eine bestimmte Störgröße kompensieren. Abbildung 3.14 illustriert die partiellen Guard Bands in herkömmlichen Systemen im Vergleich zu AVS-basierten Systemen. Von besonderer Bedeutung sind die Guard Bands für Prozessvariation, Alterung, Temperatur und Voltage Drop. Je nach Umsetzung kann AVS nur einige davon, typischerweise auch nur teilweise, ersetzen. Wie bereits in Abschnitt 3.5.1 erläutert, können beispielsweise viele veröffentlichte AVS-Ansätze keine OCVs erfassen. Allgemein gilt, dass jede (Teil-)Störgröße, die nicht durch das AVS-System erfasst und ausgeregelt werden kann, durch ein entsprechendes residuales Guard Band ausgeglichen werden muss. Wird dies nicht vorgesehen, kann ein fehlerfreier Betrieb des Gesamtsystems nicht gewährleistet werden.

Die verschiedenen Störgrößen unterscheiden sich sehr stark in ihrem dynamischen Verhalten. Beispielsweise treten Prozessvariationen nur einmal während der Produktion auf und können für die Lebensdauer des Systems als weitestgehend statisch angesehen werden. Im Gegensatz dazu beeinflussen Voltage Drops das Systemverhalten sehr schnell. Einzelheiten können aus Abbildung 3.15 entnommen werden. Offensichtlich kann eine Störgröße nur dann durch AVS kompensiert werden, wenn das AVS-System diese schnell genug erfassen und ausregeln kann. Das bedeutet, dass die folgenden Schritte erfolgreich ausgeführt werden müssen, bevor eine Störgröße aufgrund ihres Zeitverhaltens das System korrumpieren könnte:

- Das Messen oder Abschätzen des aktuellen Delays beziehungsweise das Erkennen eines Störereignisses
- Die Verarbeitung des Regelalgorithmus und die Anforderung der neuen Versorgungsspannung vom VRM
- Das Ausregeln der Spannung auf den neuen Spannungspegel durch das VRM

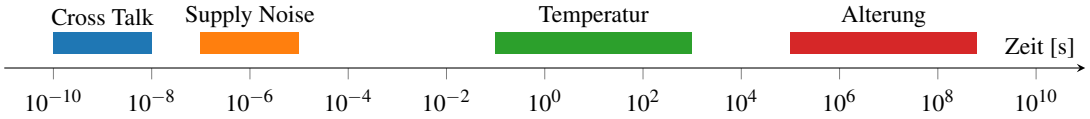


Abbildung 3.15: Einordnung verschiedener Störgrößen bezüglich der zeitlichen Größenordnung, innerhalb derer sie das Timing beeinflussen. Daten aus [A 19], [A 120] sowie eigenen Messungen

Schnelle Störgrößen wie beispielsweise Voltage Drops können deshalb nur mit sehr großem Aufwand ausgeregelt werden. Exemplarisch sei hier der Ansatz von [A 71] zur teilweisen Ausregelung von Voltage Drops genannt. Um die Spannung bei Auftreten eines Voltage Drops schnell genug auf den benötigten Wert zu stabilisieren, wird für die Verarbeitung und Übermittlung der Sensorwerte dynamische Logik verwendet. Dynamische Logik hat jedoch entscheidende Nachteile, insbesondere in Kombination mit abgesenkter Versorgungsspannung wie bei DVFS und AVS. Darüber hinaus ist sie auf keinem dem Autor dieser Arbeit bekannten FPGA verfügbar. Die Versorgungsspannung wird im Normalbetrieb durch einen zwischengeschalteten Widerstand abgesenkt, durch dessen Kurzschließung die Spannung dann sehr schnell erhöht werden kann. Da der gesamte Stromfluss zur Versorgung des Chips – im Normalbetrieb – durch diesen Widerstand fließt und dort einen entsprechenden Spannungsabfall verursacht, wird ein erheblicher Teil der durch die niedrigere Versorgungsspannung in der CMOS-Schaltung eingesparten Energie in diesem Vorwiderstand in Wärme umgesetzt. Aufgrund dieses sehr großen Aufwands vernachlässigen viele AVS-Ansätze sehr schnell veränderliche Störgrößen und sehen stattdessen ein geeignetes residuales Guard Band vor.

Eine untere Grenze für die Reaktion des AVS-Systems auf eine Störgröße ist die aufsummierte Verzögerung entlang der Regelschleife, wie sie in Abbildung 3.16 dargestellt ist. Sie soll

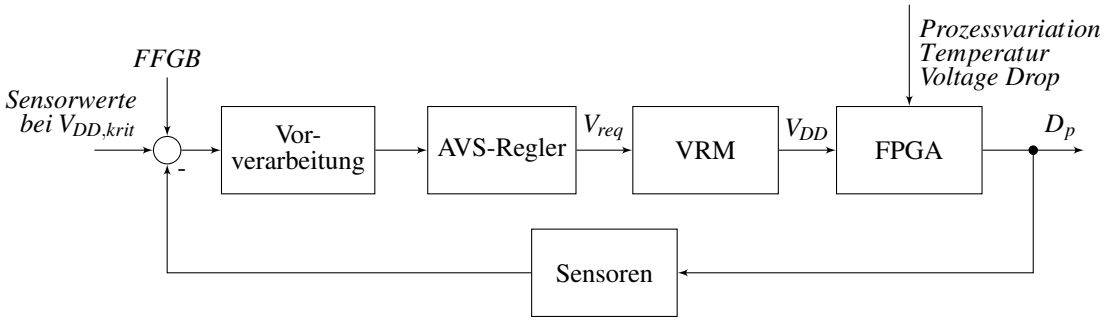


Abbildung 3.16: Geschlossene AVS-Regelschleife mit residualem Guard Band (abgeändert aus [B 2])

$T_{control\_loop}$ heißen. Gleichung 16 ist die Berechnungsvorschrift dieser Größe für das in dieser Arbeit entworfene AVS-System. Mit geringen Anpassungen beschreibt sie auch andere AVS-Systeme, sodass die abgeleiteten Überlegungen verallgemeinert werden können.

$$T_{control\_loop} = T_S + T_{S\_C} + T_{controller} + T_{VRM} + T_{FPGA}, \tag{16}$$

Dabei ist T_S die Verzögerung durch das Auslesen der Sensoren, welche vor allem durch die Wiederholrate der Sensorauslesung bestimmt wird. T_{S_C} ist die Verzögerung durch die Vorverarbeitung der Sensordaten. Hier ist als dominierender Prozessschritt die Ermittlung des Worst-Case-Sensorwertes unter den jeweils gleichzeitig gemessenen Werten zu nennen. Die Zeit, die die ACU benötigt, um den neuen Spannungspegel zu berechnen und anzufordern, wird als $T_{controller}$ bezeichnet. Das VRM benötigt die Zeit T_{VRM}, um die angeforderte Spannung tatsächlich bereitzustellen. Auch zwischen dem Ausgang des VRM und den Transistoren auf dem FPGA kommt es durch Parasitäten im PDN und auf dem Chip zu einer Verzögerung. Diese wird hier T_{FPGA} genannt.

Die Berechnungen T_{S_C} und $T_{controller}$ werden in dieser Arbeit durch FSMs in Hardware ausgeführt (vergleiche dazu Unterabschnitt 3.8.5) und benötigen lediglich wenige Taktzyklen. Dies entspricht einem Delay von unter einer μs. T_{FPGA} wird vom PDN dominiert und liegt typischerweise in der Größenordnung von Nanosekunden [A 121]. Im Gegensatz dazu dominiert T_{VRM} die Verzögerungszeit entlang der AVS-Regelschleife. Im Falle des im Rahmen dieser Arbeit verwendeten ML605-Boards ist die Reaktionszeit der VRM mit 300 μs angegeben [A 122].

VRM-Module mit kürzerer Reaktionszeit sind prinzipiell realisierbar und auch am Markt erhältlich. Allerdings ist eine kürzere Reaktionszeit nur zu Lasten einer schlechteren Energieeffizienz realisierbar [A 123]. Zudem sind entsprechende Module kostenintensiv. Man beachte, dass Linearregler trotz ihrer schnellen Reaktionszeit nicht geeignet sind, da sie durch ihre schlechte Energieeffizienz die Vorteile von AVS zunichte machen würden [A 123]. Eine vielversprechende Möglichkeit für künftige AVS-Anwendungen sind Fully Integrated Voltage Regulators (FIVRs) mit sehr schneller Reaktionszeit, wie von Zimmer et al. vorgeschlagen [A 124]. Die angegebene Reaktionszeit von 20 ns würde das schnelle Ausregeln einiger Störgrößenbestandteile ermöglichen, die bisher vom FFGB kompensiert werden müssen. Da heutige kommerzielle FPGAs, soweit dem Autor bekannt, ausschließlich externe VRMs verwenden, fokussiert diese Arbeit auf ebensolche Systeme.

T_S kann durch die Wiederholrate der Sensorauslesung in weiten Bereichen frei gewählt werden. Da T_{VRM} nicht frei beeinflussbar ist und das Verhalten der Regelschleife dominiert, ist es günstig, T_S in der gleichen Größenordnung zu wählen. Dadurch wird ohne signifikante Verschlechterung des dynamischen Verhaltens der AVS-Regelschleife ein möglichst infrequentes Auslesen der Sensoren gewährleistet. Dies ist günstig, da somit Probleme wie die Eigenerhitzung und Alterung der Sensoren vermieden werden.

Da Guard Bands die Leistungsfähigkeit, genauer den Energiebedarf für eine bestimmte Rechenleistung, ungünstig beeinflussen, ist es von entscheidender Bedeutung, das residuale Guard Band eines AVS-Systems so klein wie möglich zu wählen. Dabei muss jedoch als harte Randbedingung die Zuverlässigkeit des Systems auch bei seltenen, besonders ungünstigen Störgrößenkonstellationen gewährleistet sein. Das im Rahmen dieser Arbeit entwickelte AVS-System misst und

berücksichtigt im Kalibrierungsschritt einige Störgrößen, die bei vielen AVS-Ansätzen durch residuale Guard Bands kompensiert werden müssen. Dazu zählen insbesondere OCVs, lokal verschieden ausgeprägte Alterung und die Variabilität im Clock-Netzwerk. Folglich dient das residuale Guard Band primär dazu, schnell veränderliche Störgrößen zu kompensieren und soll deshalb Fast Fluctuation Guard Band (FFGB) genannt werden.

Die dominierende Störgröße bezüglich sehr schneller Parameteränderungen ist der Voltage Drop [A 125]. Für den ASIC-Entwurfsprozess stehen mächtige EDA-Tools zur Verfügung, um diesen abzuschätzen. Im Rahmen der vorliegenden Arbeit werden kommerzielle FPGAs als eine wichtige Zielplattform für AVS untersucht. Da die für eine toolgestützte Voltage-Drop-Analyse nötigen Informationen über den verwendeten CMOS-Prozess und den internen Aufbau des FPGA von den Herstellern nicht preisgegeben werden, wird auf experimentell ermittelte Werte zurückgegriffen.

Die wichtigsten Komponenten des Voltage Drop sind der IR-Drop und das $L \cdot di/dt$ Rauschen [A 126]. Bei schnellen Änderungen des Stromflusses durch das PDN verursacht dessen parasitäre Induktivität eine Änderung des Potentials zwischen der positiven Versorgungsspannung (V_{DD}) und Ground (GND) über den Transistoren bzw. Gattern einer CMOS-Schaltung. Dieser Effekt wird als $L \cdot di/dt$ Rauschen bezeichnet. In vielen Veröffentlichungen zu AVS wird dieser Effekt vernachlässigt. Dies wurde in Abschnitt 3.3 für die entsprechenden Arbeiten vermerkt. Allerdings wurden in anderem Zusammenhang Arbeiten zur Charakterisierung von FPGAs unter dem Einfluss von schnellen Änderungen des Stromflusses veröffentlicht. Die hervorragende Arbeit von Gnad et al. [A 125] zu dieser Thematik nutzt spezielle Delay Lines auf Basis von Carry Chain Primitives des FPGA. Es wird eine bemerkenswerte zeitliche Auflösung von 11,7 ps erreicht. Ursprünglich wurde diese Methodik nicht für AVS-Systeme entwickelt, sondern für Messsysteme für physikalische Experimente [A 127] sowie für die Erkennung von Seitenkanalattacken über das PDN eines FPGA [A 128]. Gnad et al. [A 129], [A 125] nutzen diese Methode, um die benötigten Guard Bands für $L \cdot di/dt$ Rauschen unter dem Einfluss des schwankenden Stromflusses durch das PDN zu bestimmen. Es werden Empfehlungen für FPGA-Designstrategien abgeleitet, die den Einfluss von Voltage Drops minimieren. Dabei wird allerdings mit statischen Guard Bands gearbeitet. Die Methodik wird also nicht für AVS genutzt. Die starke Schwankung des Stromflusses durch das PDN, die als Auslöser des zu messenden Versorgungsspannungsrauschens benötigt wird, wird durch die Schaltaktivität einer großen Zahl von Flipflops hervorgerufen. Durch das Aktivieren und Deaktivieren der Schaltaktivität wird der Stromfluss durch das PDN moduliert. Zudem werden verschiedene periodische Muster von Schaltaktivität erzeugt und ihr Einfluss auf das PDN bestimmt. Diese Methodik eignet sich ideal, um den Teil des FFGB zu bestimmen, der das $L \cdot di/dt$ Rauschen kompensiert. Für das Xilinx ML605 Evaluation Board wird bezüglich des Path Delays als Worst-Case eine Verlangsamung um 260 ps respektive 5,2 % angegeben. Im Rahmen dieser Arbeit wird für die experimentelle Evaluation die exakt gleiche Plattform (Xilinx ML605) verwendet. Somit sind FPGA, VRM und Printed Circuit Board (PCB)-Layout identisch und die veröffentlichten Messwerte können für die Bestimmung des FFGB in dieser Arbeit genutzt werden.

Im Gegensatz zum $L \cdot di/dt$ Rauschen wird der Einbruch der Versorgungsspannung durch ohmsche Widerstände (IR-Drop) nicht durch die Veränderung, sondern durch den Absolutwert des Stromflusses durch das PDN hervorgerufen. In seinen Auswirkungen auf das Delay wird dieser

Effekt deutlich vom $L \cdot di/dt$ Rauschen dominiert. Für das FFGB wird hier ebenfalls ein experimentell ermittelter Wert aus [A 125] genutzt. Für den IR-Drop wird dort eine Variation des Delays um 0,5 % angegeben.

Neben den schnellen Störgrößen, wegen derer das residuale Guard Band in dieser Arbeit FFGB genannt wird, muss jedoch auch die begrenzte Messgenauigkeit der Sensoren durch das residuale Guard Band kompensiert werden. Diese wird in Unterabschnitt 3.8.4 untersucht. Die in der experimentellen Untersuchung festgestellte relative Abweichung der Sensorwerte wird zu dem Guard Band für schnelle Änderungen addiert. Ein Teil dieser Abweichung ist jedoch ebenfalls auf Voltage Drops zurückzuführen. Dadurch ist das gewählte residuale Guard Band etwas zu pessimistisch. Dies wird hier in Kauf genommen, da die Ermittlung der Abweichung der Sensormesswerte völlig frei von Voltage Drops schwer realisierbar ist.

3.8 Implementierung und experimentelle Auswertung

In diesem Kapitel wird die praktische Umsetzbarkeit der im Rahmen dieser Arbeit entwickelten AVS-Konzepte anhand einer prototypischen Implementierung belegt. Des weiteren erfolgt eine umfassende qualitative und quantitative Evaluation.

3.8.1 Testplattform

Als Testplattform dienen Xilinx ML605 Evaluation Boards. Der darauf verbaute Virtex-6-FPGA ist auf einer 40 nm Technologie produziert. Die verschiedenen Versorgungsspannungen für den FPGA werden durch zwei *UCD9240PFC* Voltage Regulator Modules (VRMs) von Texas Instruments bereitgestellt. Eine Anbindung per PMBus ermöglicht die Parametrisierung der VRMs sowie die Überwachung der Spannungspegel und fließenden Stromstärken. Dabei ist der Zugriff auf den PMBus über I/Os direkt vom FPGA oder über einen dedizierten Adapter per C# Application Programming Interface (API) von einem Personal Computer (PC) aus möglich. Letzteres ermöglicht eine umfassende Überwachung und Auswertung der Spannungspegel, Stromstärken und der Temperatur, ohne das Schaltungsdesign beziehungsweise -verhalten der implementierten Logik zu beeinflussen. Darüber hinaus bietet sie größtmögliche Flexibilität und die Möglichkeit, die angeforderte Versorgungsspannung nochmals zu prüfen, um eine Zerstörung des FPGA durch eine zu hohe Spannung im Falle einer Fehlfunktion der ACU oder fehlerhaften Datenübertragung zu verhindern. Deshalb wird diese Variante des Zugriffs auf den PMBus für die prototypische Implementierung genutzt. Folglich fordert die ACU die Versorgungsspannung in gewünschter Höhe von einer Kontrollsoftware an. Da dafür jeweils nur wenige Bytes benötigt werden, werden diese Befehle per serieller Schnittstelle an einen PC übertragen. Im Vergleich zu einem direkten Zugriff auf den PMBus hat dieses Vorgehen den Nachteil, dass ein zusätzliches Delay in die AVS-Regelschleife eingefügt wird. Ein solches beeinflusst das Verhalten des AVS-Systems negativ, da dessen Reaktionszeit auf schnelle Änderungen länger wird. Die genannten Vorteile rechtfertigen dieses Vorgehen für eine prototypische Implementierung dennoch, da das genaue Delay entlang der Regelschleife ohnehin je nach verwendeter Hardware variiert, weil es von den spezifischen Gegebenheiten, wie beispielsweise dem Zeitverhalten des gewählten VRMs, abhängig ist. Abbildung 3.17 zeigt den experimentellen Aufbau aus PC und Xilinx ML605 Evaluation Board.

3.8.2 Data Retention Voltage

Wie in Unterabschnitt 3.5.2 beschrieben, darf die Versorgungsspannung eines FPGAs die *Data Retention Voltage* nicht unterschreiten, da sonst die Konfiguration des FPGAs korrumpiert werden könnte. Konkret wird für den verwendeten Virtex-6 FPGA bei einer Versorgungsspannung von 1,0 V eine nominelle Data Retention Voltage von 0,75 V angegeben [A 130]. Ebenso wie die Versorgungsspannung muss auch die Data Retention Voltage ein Guard Band für Voltage Drops enthalten. Die Kalibrierungsprozedur wird jedoch außerhalb des Anwendungsbetriebes bei kontrollierten Bedingungen durchgeführt. Insbesondere ist die Schaltaktivität gering und große Lastwechsel sind ausgeschlossen. Deshalb kann die nominelle Data Retention Voltage um das in Abschnitt 3.7 diskutierte Guard Band für Voltage Drops verringert werden. Zudem bezieht sich die spezifizierte Data Retention Voltage auf die tatsächlich am FPGA anliegende Spannung. Bei dem auf der Testplattform verwendeten VRM wurde als tatsächlich am FPGA anliegende Spannung stets eine größere als die eingestellte Spannung gemessen.

Zur Ermittlung der tatsächlich erforderlichen Data Retention Voltage wird ein Voltage Sweep durchgeführt. Dieser wird gestoppt, sobald die nominelle Data Retention Voltage als Versorgungsspannung gemessen wird. Dann werden die Online-Sensoren ausgelesen. Anschließend wird die Versorgungsspannung so lange weiter abgesenkt, bis die Sensorwerte gerade um das Guard Band für Voltage Drops geringer ausfallen. Die zu diesem Zeitpunkt vom VRM angeforderte Spannung ist die tatsächliche Data Retention Voltage für das hier verwendete experi-

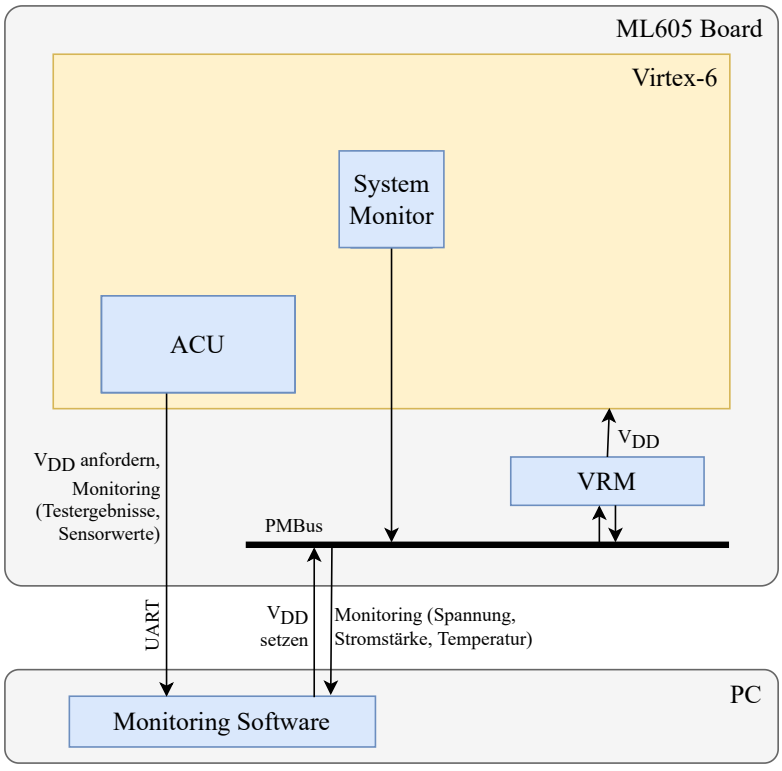


Abbildung 3.17: Testplattform für die experimentelle Evaluation

mentelle Setup. Im weiteren Text ist mit „Data Retention Voltage“ immer der so ermittelte Wert gemeint.

Da im Anwendungsbetrieb mit Voltage Drops gerechnet werden muss, wird durch die ACU sichergestellt, dass außerhalb der Kalibrierung keine Versorgungsspannung unterhalb der nominellen Data Retention Voltage vom VRM eingestellt wird.

3.8.3 Heater-Design

In der Wissenschaft, aber auch im praktischen Einsatz, beispielsweise zur Fehleranalyse, oder für den Burn-in-Prozess vor Auslieferung eines Produktes ist es erforderlich, einen FPGA bei erhöhter Temperatur zu betreiben. Auch im Rahmen dieser Arbeit ist das Verhalten eines FPGA unter Einfluss einer veränderlichen Chiptemperatur von essentiellem Interesse. Klassischerweise werden für solche Untersuchungen Heizschränke verwendet. Diese Methode ist jedoch vergleichsweise aufwändig und teuer [A 131]. Deshalb wird in der Literatur vorgeschlagen, die Rekonfigurierbarkeit von FPGAs zu nutzen, um diese mit dedizierten *Heater Cores* auf die gewünschte Temperatur zu bringen [A 131], [A 132], [A 133], [A 134], [A 135]. Designziel eines solchen Heater Cores ist es folglich, möglichst viel Energie auf einer kleinen Fläche umzusetzen. Dadurch ergibt sich auch eine weitere Anwendungsmöglichkeit: das gezielte Herbeiführen eines Voltage Drops durch das gleichzeitige Aktivieren einer großen Zahl von Heatern. Dies wird beispielsweise für Seitenkanalangriffe über das PDN genutzt und ist deshalb für die Sicherheitsforschung von Interesse [A 136], [A 137], [A 138].

In der Literatur wurden bereits vielfältige Heater-Designs für FPGAs vorgeschlagen und auch miteinander verglichen [A 133], [A 134], [A 135]. Insbesondere [A 134] gibt einen sehr guten und fundierten Überblick über verschiedene Ansätze. Die vorgeschlagenen Heater-Designs lassen sich dabei einerseits nach der Art der Leistungsumsetzung und andererseits nach der Art der verwendeten FPGA-Ressourcen kategorisieren.

Bezüglich der **Art der Leistungsumsetzung** werden zum einen erzwungene Kurzschlüsse und zum anderen sehr hohe Schaltaktivitäten vorgeschlagen. Ein Aufheizen des FPGA durch erzwungene Kurzschlüsse wurde von Xilinx zur Fehleranalyse von Anwenderdesigns [A 139] und zur thermischen Charakterisierung von Integrated Circuit (IC)-Packages [A 132] vorgeschlagen und patentiert. Die Kurzschlüsse werden dabei erzeugt, indem die Ausgänge zweier Logikblöcke miteinander verbunden werden, von denen einer eine logische `'1'` und einer eine logische `'0'` treibt. Offensichtlich muss diese Art von Leistungsumsetzung sehr präzise kontrolliert werden, um eine Zerstörung des FPGA zu verhindern [A 131]. Dies ist aus Anwendersicht sehr schwer umzusetzen, weil präzise Informationen zur technischen Umsetzung und elektrischen Dimensionierung und Robustheit nur dem Hersteller zur Verfügung stehen. Folglich werden in der Wissenschaft, soweit dem Autor bekannt, ausschließlich Heater auf Basis sehr hoher Schaltaktivitäten genutzt.

Bezüglich der **Art der FPGA-Ressourcen** wurden in der Literatur sämtliche auf modernen FPGAs zur Verfügung stehenden Ressourcen untersucht. In [A 134] werden beispielsweise mit Zufallsdaten beaufschlagte DSP-Slices sowie exzessive Schreib- und Lesezugriffe auf BRAM genutzt. Solche, auf spezialisierte Ressourcen zugeschnittene Verfahren haben jedoch den Nachteil, dass diese Ressourcen nur in begrenzter Anzahl und an genau definierten Orten vorhan-

den sind. Darüber hinaus ist ein Heater-Design auf dieser Basis nicht ohne weiteres auf andere FPGA-Modelle, insbesondere von anderen Herstellern, übertragbar. Logikressourcen wie LUTs und Flipflops stehen dagegen in großer Zahl und überall auf dem FPGA zur Verfügung. Heater-Designs auf dieser Basis können generisch in einer Hardwarebeschreibungssprache beschrieben und somit problemlos auf andere FPGA-Modelle migriert werden.

Unter den beschriebenen Ansätzen, die auf Logikressourcen basieren, gibt es wiederum synchrone Ansätze, also solche, die wesentlich auf Flipflops basieren und deren Leistungsumsatz proportional zur Taktfrequenz ist. Typischerweise werden synchrone Heater als Shift-Register oder Abwandlungen von Shift-Registern, die auch LUTs enthalten, implementiert [A 134]. Eine neuartige, sehr interessante Sonderform der synchronen Heater sind solche auf Basis von Glitches, wie sie in [A 138], [A 140] und [A 141] vorgeschlagen werden. Dabei werden sehr lange Logikpfade mit vielen `XOR`-Gattern genutzt, wobei durch Glitches eine hohe Schaltaktivität entsteht. Diese Methode wurde vorgeschlagen, da die resultierenden Designs typischen Anwendungsschaltungen, wie etwa Advanced Encryption Standard (AES)-Hardwareimplementierungen, ähneln. Dadurch sind die Heater schwer in einem FPGA-Bitstring erkennbar. Dies ist für Seitenkanalattacken auf FPGAs im Cloud Computing von Interesse, da Cloud-Anbieter die Designs auf Heater prüfen, um eben solche Attacken zu verhindern [A 138]. Die Methode weist jedoch eine vergleichsweise geringe Flächeneffizienz bezüglich der Heizleistung auf und bietet deshalb außerhalb des sehr speziellen Anwendungsszenarios keine Vorteile. Für synchrone Heater im Allgemeinen gilt, dass sie laut vergleichenden Untersuchungen nur relativ wenig Leistung umsetzen können [A 131], [A 134]. Bisweilen werden externe Taktgeneratoren verwendet, da häufig sehr hohe Taktfrequenzen nötig sind, um den gewünschten Leistungsumsatz zu erzielen.

Asynchrone, RO-basierte Ansätze erweisen sich als deutlich effizienter bezüglich Heizleistung pro FPGA-Fläche. Um einen möglichst hohen Leistungsumsatz zu erreichen, werden dabei häufig sehr kurze ROs genutzt. Bisweilen wird nur ein einzelner Inverter verwendet [A 133], [A 134]. Diese extrem kurzen ROs erzielen durch die schnellen Umläufe der Signalflanke sehr hohe Schaltaktivitäten. Allerdings haben sehr kurze ROs als Heizelemente mehrere Nachteile:

- Sie benötigen umfangreiche Logik zur Ansteuerung sowie zur Ausleitung ihres Signalwertes. Letzteres ist nötig, damit die EDA-Tools die Heater nicht verwerfen. Die Zahl der LUTs, die für diesen Overhead benötigt werden, übersteigt die Zahl der tatsächlich Heizleistung erzeugenden LUTs um mehr als das 13-fache [A 134].

- Daraus ergibt sich, dass die enorme Heizleistung sehr punktuell in nur wenigen LUTs umgesetzt wird. Dies führt zu einer extremen Belastung der verwendeten Hardwareressourcen durch dreierlei Einflüsse: die extreme punktuelle Hitzebelastung, die extreme lokale Stromdichte in den V_{DD}- und GND-Netzen, welche zu Elektromigration führen kann sowie die enorme Anzahl an Gate Toggles, welche zu einer enorm beschleunigten Alterung führt.

- Die Heizleistung variiert stark [A 134].

- Durch die enorme Heizleistung kann es leicht zum Überhitzen des FPGA kommen [A 133].

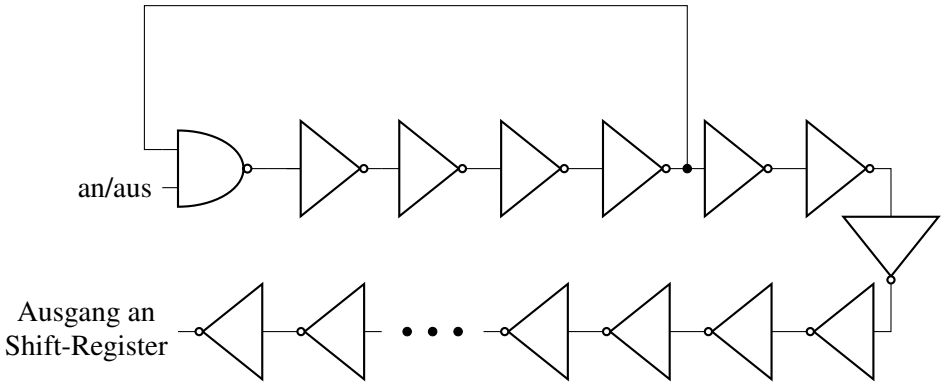


Abbildung 3.18: In dieser Arbeit verwendetes Heater-Design

- Durch den enormen, lokal konzentrierten Leistungsumsatz kann es zu unzulässigen Einbrüchen der Versorgungsspannung kommen, die zu einem Reset des FPGA führen [A 133].

Aus den genannten Gründen werden für die Charakterisierung des FPGA in dieser Arbeit keine sehr kurzen ROs verwendet. Ein weiterer Grund für diese Entscheidung ist, dass die Untersuchungen in dieser Arbeit zwar an Worst-Case-Szenarien ausgerichtet sind – dabei bezieht sich Worst-Case jedoch auf in realen Designs zu erwartende Werte und nicht auf die Durchführung von Seitenkanalangriffen.

Im Rahmen dieser Arbeit wurde deshalb ein anderes Heater-Design implementiert. Dieses beruht auf einer sehr langen Inverterkette. Dadurch wird die benötigte Overhead-Logik stark reduziert und die Leistung wird weniger punktuell, sondern verteilt über viele FPGA-Ressourcen umgesetzt. Somit werden die oben genannten Nachteile kurzer ROs vermieden. Die Taktflanken werden durch einen RO mittlerer Länge erzeugt. Dafür wird das Signal nach dem fünften invertierenden Element (ein `NAND`-Gatter und vier Inverter) wieder auf den Eingang zurückgeleitet. Dadurch laufen mehrere Signalflanken gleichzeitig durch die Inverterkette. Soweit dem Autor bekannt, wurde dies bisher nicht in der Literatur beschrieben. Abbildung 3.18 illustriert das Heater-Design. Konkret wird (inklusive Rückkopplungsschleife) eine Kette von 648 Invertern und einem `NAND`-Gatter verwendet. Die Platzierung der Heater wird mittels `AREA_GROUP`-Constraints kontrolliert.

Der Ausgang des letzten Inverters in der Kette wird nach außen geführt. Diese Ausgangssignale aller Heater werden dann in ein gemeinsames Shift-Register eingekoppelt, dessen Ausgang einen Ausgangspin des FPGA treibt. Dadurch wird ein unerwünschtes Entfernen des Heaters durch die Optimierungen des EDA-Tools verhindert. Da die einzelnen Heater jeweils sehr groß sind und viel Leistung abgeben, kann die absolute Zahl der Heater und somit auch der Overhead gering gehalten werden.

Das Heater-Design wurde auf der in Unterabschnitt 3.8.1 beschriebenen Testplattform, also einem Virtex 6 FPGA, getestet. Ein Heater verursacht bei Aktivierung im Schnitt einen zusätzlichen Stromfluss von 97,5 mA und belegt im Schnitt 649 LUTs. Eine beispielhaft getestete Konfiguration verwendet 100 Heater, die 64 949, beziehungsweise 43 % der verfügbaren LUTs

belegen. Diese 100 Heater verursachen einen zusätzlichen Stromfluss von 9,75 A. Das sind 48 % des maximal zulässigen Stromflusses des VRM [A 142]. Der Overhead zur Ansteuerung und Ausleitung der Signale liegt bei deutlich unter einem Prozent und entsteht lediglich durch das Shift-Register und eine State Machine. Diese aktiviert je nach Konfiguration Gruppen von Heatern nacheinander oder gleichzeitig.

3.8.4 Experimentelle Validierung der Online-Sensoren

In der vorliegenden Arbeit werden ROs als Online-Sensoren für ein FPGA-basiertes AVS genutzt. Die Sensoren werden dabei regelmäßig auf das Verhalten der kritischen Pfade kalibriert. In Abschnitt 3.6 wurde auf konzeptioneller Ebene beschrieben, dass es für diesen speziellen Anwendungsfall vorteilhaft ist, die ROs nicht wie sonst üblich per Hard Macro zu platzieren. Stattdessen werden `LOC...RANGE`-Constraints genutzt, um die Sensoren so zu platzieren, dass sie einen möglichst repräsentativen Teil der Sensorregion erfassen. Dadurch wird außerdem erreicht, dass das durch Routing verursachte Delay besser repräsentiert wird. In diesem Kapitel wird diese konzeptuelle Aussage experimentell validiert.

Für die Untersuchung wurde ein gesondertes Testsystem genutzt. Dieses basiert auf der in Unterabschnitt 3.8.1 beschriebenen experimentellen Anordnung. Das FPGA wird in Sensorregionen der gleichen Größe, wie sie auch in der AVS-Implementierung genutzt werden, eingeteilt. In jeder dieser Regionen wird zu Vergleichszwecken ein zentral platzierter Hard-Macro-Sensor instanziiert. Zusätzlich werden fünf weitere Sensoren pro Region platziert. Diese sind aus der gleichen RTL-Beschreibung synthetisiert und aus identischen Logikblöcken aufgebaut wie die Hard-Macro-ROs. Sie sind jedoch nicht per Hard Macro platziert, sondern vom Synthesetool innerhalb gewisser, durch `LOC...RANGE`-Constraints vorgegebener Grenzen frei platziert. Im Weiteren werden so platzierte Sensoren kurz als `LOC`-Sensoren bezeichnet. Da die genaue Platzierung der Logikkomponenten und insbesondere ihr Routing nicht festgelegt sind, werden sich in der Praxis diesbezüglich variierende Umsetzungen der Sensoren ergeben. Das Setup zur experimentellen Validierung muss also so angelegt sein, dass solche Variationen des Sensor-Placements und -Routings tatsächlich auftreten. Würde das EDA-Tool die Sensoren unbeabsichtigter Weise immer gleich implementieren, könnte mit dem Experiment keine Aussage über die Eignung von `LOC`-Sensoren für AVS-Systeme getroffen werden. Im hier verwendeten Setup wird durch die Platzierung mehrerer Sensoren pro Region und die hohe Auslastung des FPGA durch weitere Logik und Heater sichergestellt, dass die Sensoren tatsächlich auf verschiedene Logik- und Routing-Ressourcen abgebildet werden. Abbildung 3.19 zeigt die in dieser Arbeit genutzte RO-Variante. Sie besteht aus 21 invertierenden Elementen, einem `AND`-Gatter zum Aktivieren, sowie zwei Countern – einem Referenz-Counter, der die Messdauer auf 2^{12} Taktzyklen ($\sim 82\mu s$) festlegt und einem Counter, der die Anzahl der Signalflankendurchläufe durch den Inverterring zählt.

Innerhalb jeder Region werden 40 Logikpfade mit verschiedener Logiktiefe platziert, die jeweils an einem Register beginnen und enden. Die genaue Platzierung innerhalb der Region und das Routing sind dabei nicht näher festgelegt, sondern erfolgen frei durch die Xilinx-EDA-Software. Nun wird wiederholt die Versorgungsspannung stufenweise von der nominellen Versorgungsspannung bis zur Data Retention Voltage abgesenkt, wobei analog zur Kalibrierungsprozedur alle Pfade auf korrektes Timing getestet werden. Während jedes Tests werden auch die Sensoren

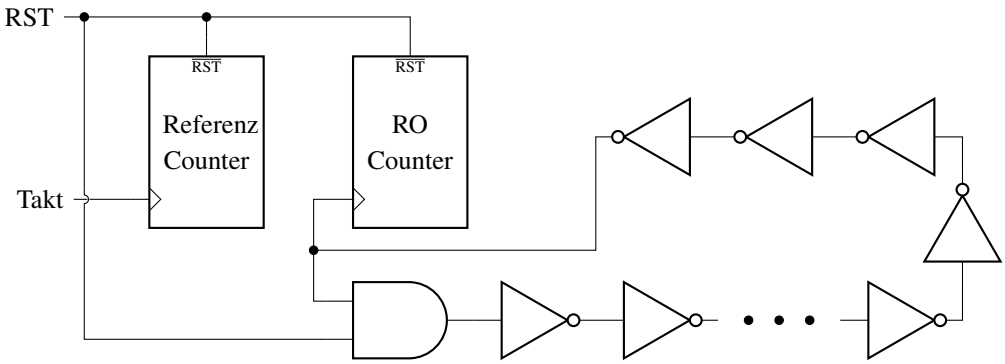


Abbildung 3.19: Ring-Oszillator-Implementierung

ausgelesen. Testergebnisse und Sensorwerte werden dann zur späteren Auswertung an einen PC übertragen.

Das Verhalten der Sensoren soll unter dem Einfluss verschiedener Störgrößen untersucht werden. Dazu wird ein Drittel des FPGAs nahezu vollständig mit Heater Cores gefüllt. Es verbleiben gerade noch genug Ressourcen, um die Sensoren und Logikpfade für den Versuch platzieren zu können. Durch die Aktivierung dieser Heater entstehen lokal abweichende Voltage Drops und Temperaturgradienten. Um verschiedene Betriebszustände in einem Testlauf abzubilden, werden drei Phasen durchlaufen.

- Direkt nach der Initialisierung des FPGA mit dem Bitfile wird im ersten Voltage Sweep die Kalibrierung durchgeführt. Danach wird der FPGA für weitere 20 Minuten bei abgeschalteten Heater Cores betrieben. Seine Temperatur ändert sich dabei im Wesentlichen nur durch die Abwärme der statischen Verlustleistung.
- Anschließend werden die Heater für 20 Minuten aktiviert. Dies führt zu einem Voltage Drop und einem starken Temperaturanstieg.
- Nach Deaktivierung der Heater werden für weitere 30 Minuten Voltage Sweeps durchgeführt, um auch die Phase des Abkühlens zu erfassen.

Insgesamt dauert der Testlauf also knapp anderthalb Stunden. Dabei wird nach Abschluss jedes Voltage Sweeps für zwei Minuten die nominelle Versorgungsspannung gehalten, um ein möglichst effektives Arbeiten der Heater zu ermöglichen. Um den Einfluss der Prozessvariation abzubilden, werden fünf baugleiche FPGAs untersucht, wobei jeweils das identische Bitfile zur Konfiguration genutzt wird.

Die während der Experimente aufgenommenen Datenreihen werden anschließend mit Python und dem Pandas-Framework ausgewertet. Wie beschrieben werden die Daten des ersten Voltage Sweeps für eine virtuelle Kalibrierung analog zur in Unterabschnitt 3.5.2 vorgestellten Kalibrierungsprozedur genutzt. Dabei wird für jede Kombination aus Pfad und Sensor innerhalb einer Region geprüft, wann erstmalig ein Timing-Fehler des Pfades auftritt. Der gleichzeitig gemessene Sensorwert wird dann als Kalibrierungswert für die jeweilige Pfad-Sensor-Kombination genutzt. Für alle weiteren, bei verschiedenen Bedingungen durchgeführten, Voltage Sweeps

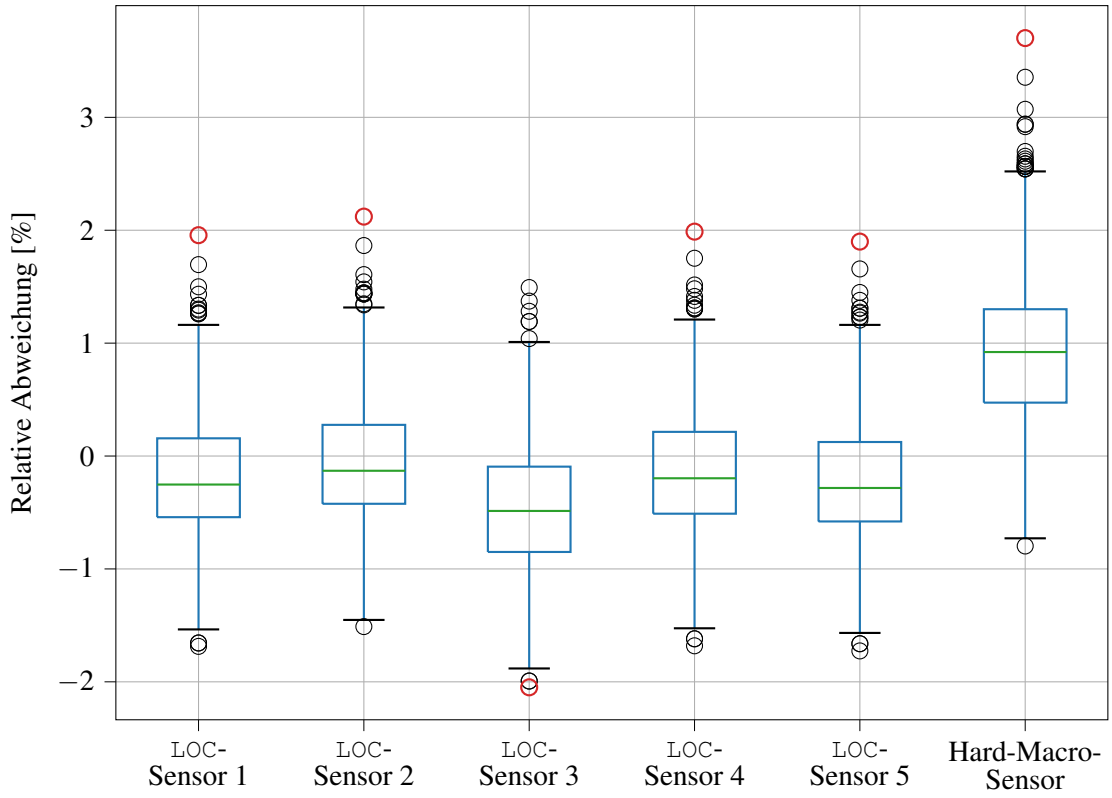


Abbildung 3.20: Beispielhafte Auswertung einer Sensorregion; Die betragsmäßig größte relative Abweichung jedes Sensors wurde rot hervorgehoben.

wird nun ebenfalls für jede Pfad-Sensor-Kombination der Sensorwert bei erstmalig auftretenden Timing-Fehlern ermittelt. Von diesen Werten wird dann – analog zur Bestimmung der Regelabweichung des AVS-Systems (vergleiche Abbildung 3.16 in Abschnitt 3.7) – der Kalibrierungssensorwert der jeweiligen Pfad-Sensor-Kombination subtrahiert. Nachfolgend wird also stets die Differenz zwischen aktuellem und Kalibrierungssensorwert bei erstmalig fehlschlagendem Timing betrachtet. Für die Anwendung als Online-Sensor für ein AVS-System ist es wünschenswert, dass das Auftreten von Timing-Fehlern reproduzierbar erst bei Unterschreiten eines bestimmten Sensorwertes erfolgt. Somit ist diese verbleibende Differenz die Abweichung von einem ideal wünschenswerten Sensorverhalten. Da die absoluten Zahlenwerte jedoch stark von der Dimensionierung des Sensors, insbesondere von der durch den Referenz-Counter festgelegten Messdauer, abhängig sind, werden die Differenzwerte gegen den Absolutwert der jeweiligen Sensormessung normiert. Das Ergebnis wird *relative Abweichung* genannt. Über alle Testfälle und Sensoren wird nun sowohl das durchschnittliche als auch das Worst-Case-Verhalten der Hard-Macro-Sensoren im Vergleich zu den LOC-Sensoren ermittelt.

Die Auswertung ergab ein charakteristisches Verhalten, das über alle Sensorregionen und getesteten FPGAs beobachtbar war. Zur Veranschaulichung des Verhaltens der Hard-Macro-Sensoren im Vergleich zu den LOC-Sensoren wird deshalb eine Sensorregion mit typischem Verhalten aus-

Tabelle 3.2: Auswertung über alle Sensorregionen und FPGAs, die zur Evaluation der LOC-Sensoren untersucht wurden

	LOC-Sensoren [%]	Hard-Macro-Sensoren [%]
Arithmetisches Mittel der vorzeichenbehafteten relativen Abweichung	-0,27	0,74
Arithmetisches Mittel der betragsmäßigen relativen Abweichung	0,52	0,8251
Größte positive relative Abweichung	2,88	3,70
Größte negative relative Abweichung	-2,91	-2,03
Betragsmäßig größte relative Abweichung	2,91	3,70

gewählt. Anhand dieser werden die Ergebnisse zunächst beispielhaft diskutiert. Im Anschluss daran werden die Ergebnisse über alle durchgeführten Versuche quantitativ ausgewertet. Abbildung 3.20 zeigt die Ergebnisse für die ausgewählte Sensorregion. Deutlich ist, dass sich die einzelnen LOC-Sensoren in ihrem Verhalten durchaus unterschieden – was auch zu erwarten war, da sich das genaue Placement und Routing der Sensoren unterscheidet. Der Hard-Macro-Sensor zeigt ein gegenüber den LOC-Sensoren deutlich abweichendes Verhalten, wobei die Abweichung deutlich größer ist als die der LOC-Sensoren untereinander. Zunächst fällt auf, dass die gesamte Verteilung deutlich in Richtung einer positive Abweichung verschoben ist. Das bedeutet, dass die Sensorwerte bei erstmaligem Auftreten eines Timing-Fehlers während eines Voltage-Sweeps tendenziell größer waren als während der Kalibrierung. Eine Abweichung in diese Richtung ist als besonders kritisch anzusehen, da sie bedeutet, dass in einem AVS-System das Timing eines Pfades verletzt werden könnte, obwohl sich die Sensorwerte im gewünschten Bereich befinden. Um dies zu verhindern wäre dann ein entsprechend größeres residuales Guard Band nötig. Diese Verschiebung der Verteilung der Hard-Macro-Sensor-Messwerte ist sehr charakteristisch und wurde über alle Messungen hinweg beobachtet. Wie in Abschnitt 3.6 bereits erläutert, ist eine der Grundideen der RO-Platzierung in dieser Arbeit, dass der Verzicht auf Hard Macros es ermöglicht, größere Sensoren zu verwenden. An deren Delay haben die Routing-Ressourcen einen größeren Anteil. Da auch der Anteil des Routing-Delays am Delay kritischer Pfade typischerweise groß ist, wird dadurch ein günstigeres Sensorverhalten erzielt. Der Autor dieser Arbeit hält es für plausibel, dass die systematische Verschiebung der Messwerte der Hard-Macro-Sensoren auf diese Tatsache zurückzuführen ist. Einen tatsächlichen experimentellen Beweis für diesen Zusammenhang bieten die Ergebnisse dieses Versuchs jedoch nicht. Ein solcher liegt auch außerhalb des Fokus dieser Arbeit – dem Entwurf eines AVS-Systems für FPGA.

Als weiteres Ergebnis lässt sich aus Abbildung 3.20 ablesen, dass die Streuung der relativen Abweichung der Sensorwerte für den Hard-Macro-Sensor deutlich größer ist. Da ein AVS-System stets mit einen fehlerfreien VF-Arbeitspunkt arbeiten muss, sind auch die Ausreißer unbedingt zu berücksichtigen. Die jeweils betragsmäßig größten Ausreißer wurden deshalb in der Abbildung rot markiert. Für die in Abbildung 3.20 dargestellte Sensorregion beträgt die betragsmäßig größte relative Abweichung des Hard-Macro-Sensors 3,70 %, während die größte relative Abweichung aller fünf LOC-Sensoren nur 2,12 % beträgt.

In der quantitativen Auswertung über alle Sensorregionen und getesteten FPGAs hinweg finden sich die vorhergehend beschriebenen Charakteristika wieder. Die Ergebnisse sind in Tabelle 3.2 aufgeführt. Insgesamt wurden dafür über 40 000 Datenpunkte, also relative Abweichungen des Sensorwertes bei fehlschlagendem Pfad-Timing für eine Pfad-Sensor-Kombination ausgewertet. Der für den Einsatzzweck als Online-Sensor in einem AVS-System wichtigste Parameter ist die betragsmäßig größte Abweichung. Diese beträgt für die Hard-Macro-Sensoren 3,7 % . Die `LOC`-Sensoren weisen einen günstigeren Wert von 2,91 % auf.

Zusammenfassend kann als Ergebnis der experimentellen Auswertung festgestellt werden, dass die in dieser Arbeit vorgeschlagene Sensorplatzierung durch `LOC...RANGE`-Constraints tatsächlich für den Einsatz in AVS-Systemen geeignet ist, sofern eine geeignete Sensorkalibrierung genutzt wird. Sie bietet dabei Vorteile gegenüber der Platzierung mittels Hard Macro sowohl bezüglich des Sensorverhaltens, als auch bezüglich der unproblematischeren Integration in den Design- und EDA-Arbeitsablauf.

3.8.5 Implementierung des AVS-Systems

Die Kalibrierung und sensorbasierte Regelung der Versorgungsspannung, wie sie in den Abbildungen 3.12 und 3.16 dargestellt sind, werden von einer zentralen Steuereinheit kontrolliert. Diese soll im Weiteren AVS Control Unit (ACU) genannt werden und wird als eine direkt in Hardware implementierte FSM realisiert. Die Aufgaben der ACU sind die Durchführung der Offline-Kalibrierung, die Steuerung und Auswertung der Online-Sensoren, das Berechnen der erforderlichen Versorgungsspannung sowie die Anforderung dieser von der VRM. Während der Kalibrierung unterschreitet die Versorgungsspannung den Wert der kritischen Versorgungsspannung. Somit ist das korrekte Timing des Designs nicht mehr gewährleistet. Dies ist notwendiger Teil der Kalibrierungsprozedur, wie sie in Unterabschnitt 3.5.2 beschrieben wird. Um dennoch einen zuverlässigen Ablauf der Kalibrierung zu gewährleisten, werden auf die ACU restriktivere Timing Constraints angewendet, sodass diese dennoch zuverlässig funktioniert. Das FPGA wird in 12 Sensorregionen unterteilt.

Wie in Unterabschnitt 3.5.1 erläutert, werden dedizierte Path Delay Test Patterns benötigt, um zuverlässig die kritischen Pfade anzuregen und somit das Worst-Case Delay einer kombinatorischen Logik beobachtbar zu machen. Geeignete Test Patterns können mit kommerzieller Automatic Test Pattern Generation (ATPG)-Software generiert werden. Konkret wurde *Synopsys TetraMax ATPG* genutzt. Allerdings ist diese Software, wie auch vergleichbare Produkte anderer Hersteller, für den Entwurf von ASICs gedacht. Sie sind also auf die Einbindung in eine ASIC-EDA-Toolchain ausgerichtet und beherrschen die in diesem Bereich üblichen Dateiformate, Konventionen und Teilmengen von Sprachen. So sind beispielsweise die am weitesten verbreiteten Hardwarebeschreibungssprachen Very High Speed Integrated Circuit Hardware Description Language (VHDL) und Verilog ursprünglich für Simulationen entwickelt wurden. Die Möglichkeit, sie zum Hardware-Entwurf zu nutzen, wurde erst nachträglich geschaffen [A 143]. Daraus hat sich ergeben, dass EDA-Tools typischerweise nur, je nach Anwendungsgebiet verschiedene, Teilmengen dieser Sprachen unterstützen [A 144].

Im konkreten Fall verwendet der Hersteller des FPGA, Xilinx, für die Netzliste Sprachelemente des Behavioural Modeling, während TetraMax ATPG nur eine Untermenge der strukturellen Sprachelemente unterstützt. Die Inkompatibilität der EDA-Tools betrifft aber nicht nur die

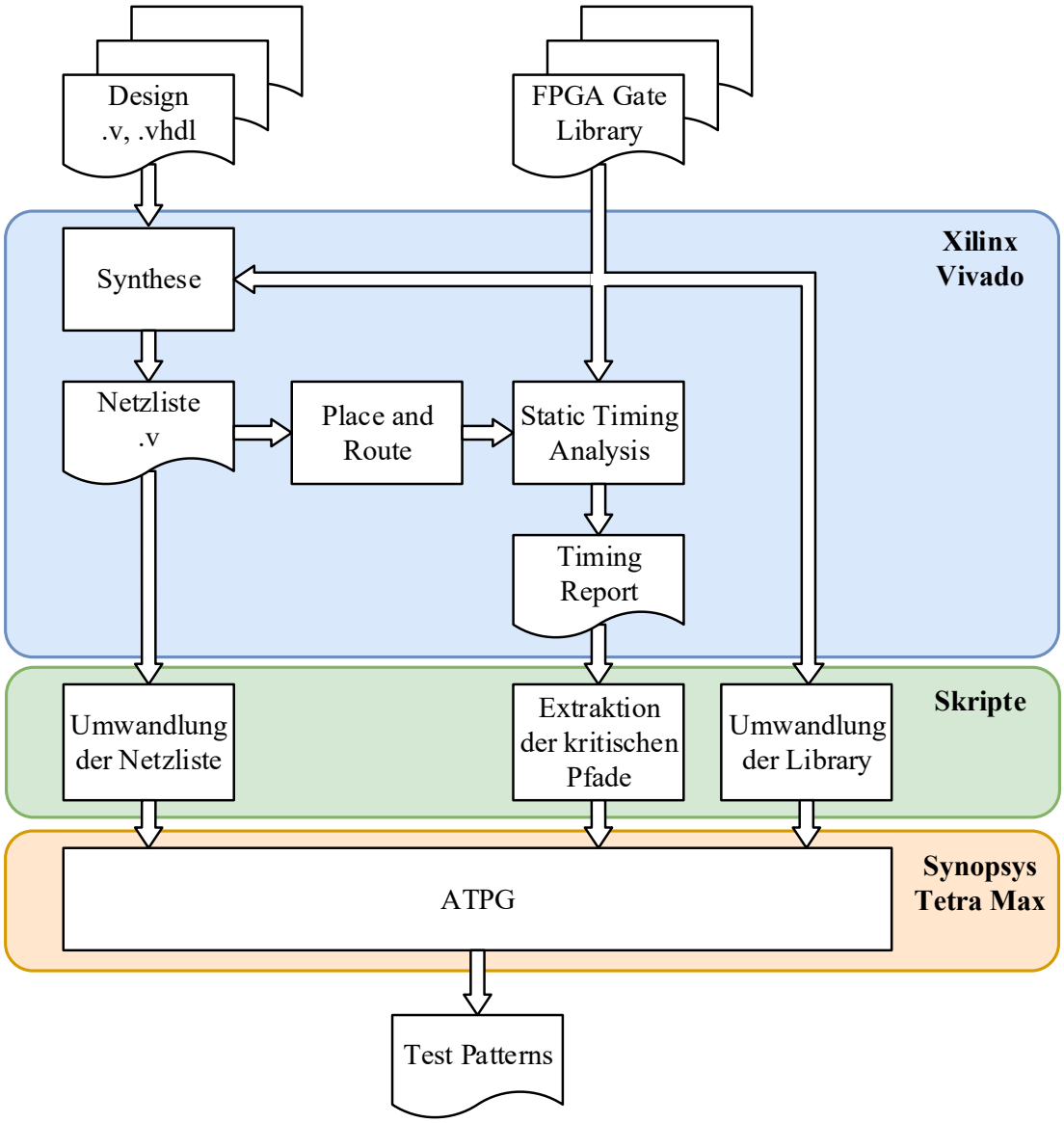


Abbildung 3.21: Prozess zum automatisierten Erstellen von Test Patterns für FPGA-Anwendungen (Abbildung übersetzt aus [B 1]; Original ©2019 IEEE)

Netzliste, sondern auch die Gate Libraries und Timing Reports. Zudem beeinflusst die Wahl der Sprachkonstrukte neben der prinzipiellen Unterstützung aber auch die Qualität der Ergebnisse. So werden von Xilinx große Wahrheitstabellen für die Repräsentation der LUTs genutzt. Diese werden von TetraMax ATPG zwar prinzipiell unterstützt, führen jedoch zu sehr schlechten Ergebnissen. Wobei unter schlechten Ergebnissen in diesem Zusammenhang zu verstehen ist, dass das Tool nur für einen kleinen Teil der angeforderten Pfade ein geeignetes Test Pattern erzeugen kann. Ähnliches gilt für das `Parameter` Statement, das von TetraMax nicht unterstützt, aber von Xilinx genutzt wird, um die Konfiguration der LUTs darzustellen. Schlussendlich müssen

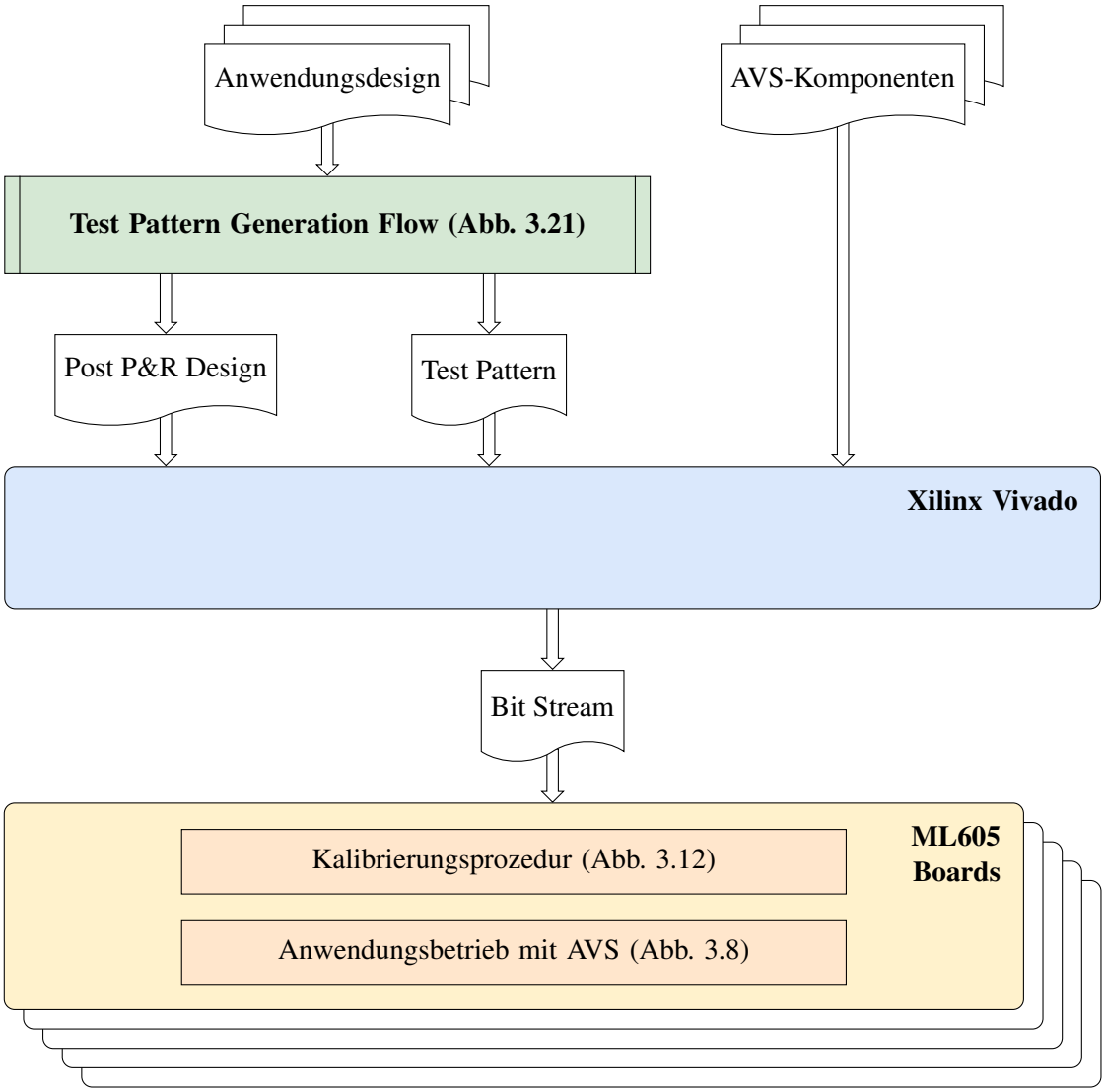


Abbildung 3.22: Gesamtablauf des vorgeschlagenen AVS-Systems für FPGAs (Abbildung übersetzt aus [B 2])

auch die Input/Output (IO)-Pads aus der Netzliste entfernt werden. Stattdessen wird direkt auf die in der Netzliste definierten Ein- und Ausgänge verbunden. Die Erzeugung eines Path Specification Files für TetraMax ATPG aus den Timing Reports ist hingegen trivial möglich. Im Rahmen dieser Arbeit wurden Skripte erstellt, um Xilinx Verilog-Netzlisten, Gate Libraries und Timing Reports automatisiert in ein von TetraMax ATPG unterstütztes Format umzuwandeln. Der Arbeitsablauf zum Generieren von Test Patterns ist in Abbildung 3.21 dargestellt.

Um den benötigten Speicherplatz für die Test Patterns möglichst klein zu halten, werden Linear Feedback Shift Register (LFSR)-Signaturen für die Test Result Compaction genutzt. Test Patterns und Signaturen werden in der prototypischen Implementierung FPGA-intern gespeichert.

Für den produktiven Einsatz ist dies jedoch keine geeignete Vorgehensweise. Stattdessen sollten externe nichtflüchtige Speicher genutzt werden. Der ohnehin für die Speicherung des Bitfiles zur Konfiguration des FPGAs benötigte Speicher kann perspektivisch für die Speicherung der Test Patterns entsprechend erweitert werden.

Das AVS-System (ohne Anwendungsdesign) bestehend aus ACU, Online-Sensoren, Kommunikations- und Monitoring-Infrastruktur und Signatur-LFSRs benötigt in der prototypischen Implementierung 3658 Flipflops und 3724 LUTs. Dies entspricht lediglich 1,21 % respektive 2,5 % der Ressourcen des FPGAs. Das Design wird bei einer Taktfrequenz von 50 MHz betrieben. Abbildung 3.22 gibt eine Übersicht über den Gesamtablauf der Implementierung des AVS-Systems für FPGA.

3.8.6 Anwendungsdesigns

Um die Umsetzbarkeit des im Rahmen dieser Arbeit entwickelten Konzepts zu zeigen, werden zwei Anwendungsdesigns um das AVS-System erweitert und getestet. Da dynamische und statische Verlustleistung verschieden stark von AVS beeinflusst werden, wird je ein dediziertes Anwendungsdesign untersucht. Konkret wird einerseits ein Design mit geringer Auslastung des FPGAs und geringer Schaltaktivität implementiert, dessen Verhalten von der statischen Verlustleistung dominiert wird. Andererseits wird ein großes Design mit hoher Schaltaktivität umgesetzt, dessen Verhalten von der dynamischen Leistungsaufnahme dominiert wird. Im Folgenden wird ersteres Design A und letzteres Design B genannt.

Design A besteht aus vier Instanzen eines Finite Impulse Response (FIR)-Filters zehnter Ordnung, ist also ein relativ kleines Design. Es wird von Xilinx Vivado auf 10957 Flipflops und 14220 LUTs abgebildet. Dies sind lediglich 3,63 % beziehungsweise 9,6 % der verfügbaren Ressourcen des FPGA. Zusätzlich wird die Leistungsaufnahme bestimmt, während an den Eingängen der Filter konstant der Wert Null anliegt. Dadurch ist die Schaltaktivität sehr gering. Folglich dominiert die statische Verlustleistung das Verhalten. Die Berücksichtigung eines solches Anwendungsszenarios, in dem die Verlustleistung durch Leckströme dominiert, ist für Low-Power-Techniken essentiell. Es bildet die Gegebenheiten aktueller Technologien und Designs ab, wie sie unter dem Schlagwort *Dark Silicon* diskutiert werden.

Design B hingegen besteht aus vier Instanzen eines FIR-Filters 68. Ordnung. Diese werden zudem in jedem Taktzyklus mit per LFSR erzeugten pseudozufälligen Eingangsvektoren beaufschlagt. Dieses Design belegt 45998 FlipFlops und 106322 LUTs. Somit ergibt sich eine Auslastung der FPGA-Ressourcen von 15,2 % beziehungsweise 70,5 %. Dies entspricht einer sehr hohen Auslastung des FPGA, insbesondere bezüglich der LUTs. Deutlich höhere Auslastungen sind praktisch kaum erreichbar, da sie mit Routing Congestions einhergehen, die das Timing Closure sehr schwierig oder sogar unmöglich machen.

An das **Timing der Anwendungsdesigns** müssen zusätzliche Anforderungen gestellt werden, um eine quantitative Evaluation des AVS-Konzepts zu ermöglichen. Konkret muss der Timing Slack vernachlässigbar klein sein. Dies ist notwendig, da andernfalls eine Absenkung der Versorgungsspannung nicht nur aufgrund der durch AVS ermöglichten Verringerung der Guard Bands, sondern auch durch Ausnutzen des freien Timing Slacks erfolgen würde. Dieser ließe sich jedoch auch – mit deutlich geringerem Aufwand – durch eine Anpassung des VF-Arbeitspunktes

zur Designzeit nutzen. Die Möglichkeit, Timing Slacks des Designs zur Energieersparnis zu nutzen, ist folglich lediglich ein (positiver) Nebeneffekt von AVS, dessen Ausmaß zudem vollständig vom konkreten Anwendungsdesign abhängig ist. Eine quantitative Evaluation eines AVS-Ansatzes ohne präzise gewählte Constraints für das Anwendungsdesign zur Vermeidung von signifikanten Timing Slacks führt somit zu fehlerhaften Ergebnissen. Einerseits kommt es zu einer systematischen Überschätzung der Energieersparnis durch AVS und andererseits zu einem erheblichen 'zufälligen' Fehler durch den individuellen Timing Slack des jeweiligen Anwendungsdesigns. Die Anwendungsdesigns in der vorliegenden Arbeit wurden so implementiert und mit Constraints versehen, dass ihr Timing Slack betragsmäßig kleiner als 150 ps ist. Diese konkrete Bedingung wurde willkürlich so gewählt, da sie einerseits technisch mit vertretbarem Aufwand realisierbar ist und andererseits sicherstellt, dass der Timing Slack bei einer Clock-Periode von 20 ns mindestens zwei Größenordnungen kleiner als die Clock Periode ist. Dies wurde im Rahmen dieser Arbeit und angesichts anderer Messungenauigkeiten als hinreichend erachtet. Nach Kenntnis des Autors wurde diese notwendige Bedingung für eine aussagekräftige quantitative Evaluation im relevanten Stand der Technik bisher nicht explizit diskutiert. Es ist aber selbstverständlich möglich, dass Autor:innen diesen Zusammenhang dennoch berücksichtigt haben.

3.9 Ergebnisse und Diskussion

Nachfolgend werden die Ergebnisse der experimentellen Auswertung des im Rahmen dieser Arbeit vorgeschlagenen AVS-Systems für FPGAs vorgestellt und analysiert.

Abbildung 3.23 zeigt den Spannungsverlauf während der Sensorkalibrierung und dem beginnenden Anwendungsbetrieb. Zu Beginn, also vor der Kalibrierung, wird die nominelle Versorgungsspannung von 1,0 V verwendet. Die tatsächlich vom VRM bereitgestellte Spannung ist jedoch etwas höher. Nach etwa 15 s beginnt die Kalibrierungsprozedur: die Versorgungsspannung wird schrittweise abgesenkt, wobei auf jeder Spannungsstufe die definierten Built-in Self-Tests (BISTs) durchgeführt werden. Sobald einer dieser Tests negativ ausfällt, also ein korrumpiertes Zeitverhalten der Schaltung offenlegt, wird für die getestete Sensorregion der Sollwert für die Spannungsregelung per AVS ermittelt.

Die kritische Versorgungsspannung wird durch die Tests auf einen Wert zwischen der Versorgungsspannung, bei der das Testergebnis negativ ist, und der letzten Versorgungsspannung, bei der der Test positiv ausfällt, eingegrenzt. Zur Bestimmung des Sollwertes wird zunächst die Versorgungsspannung wieder um eine Spannungsstufe erhöht – also auf jenes Spannungsniveau, auf dem letztmalig kein Timing-Fehler auftrat. In diesem Zustand wird der Sensor der getesteten Sensorregion ausgelesen. Wie in Abschnitt 3.7 diskutiert, wird ein FFGB benötigt, um bestimmte Störgrößen zu kompensieren, die nicht vom AVS-System ausgeglichen werden können. Dieses FFGB wird nun als Vorfaktor durch eine Multiplikation mit dem ausgelesenen Sensorwert angewendet. Das Ergebnis wird als neue Führungsgröße für die Regelung der Versorgungsspannung im AVS-System genutzt. Anschließend wird die Versorgungsspannung wieder auf jenes Niveau abgesenkt, bei dem der Timing-Fehler auftrat und die Kalibrierungsprozedur wird fortgesetzt.

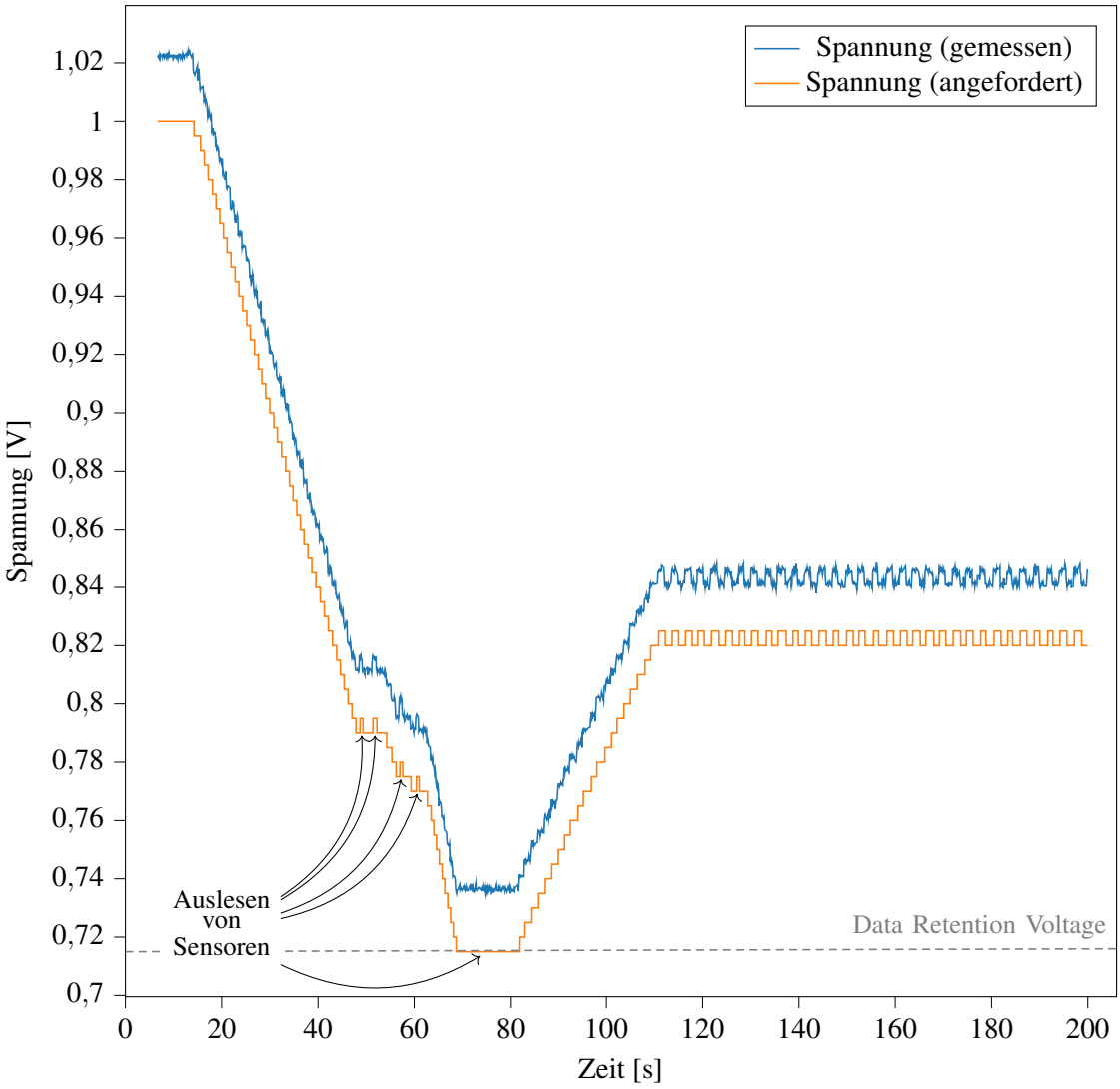
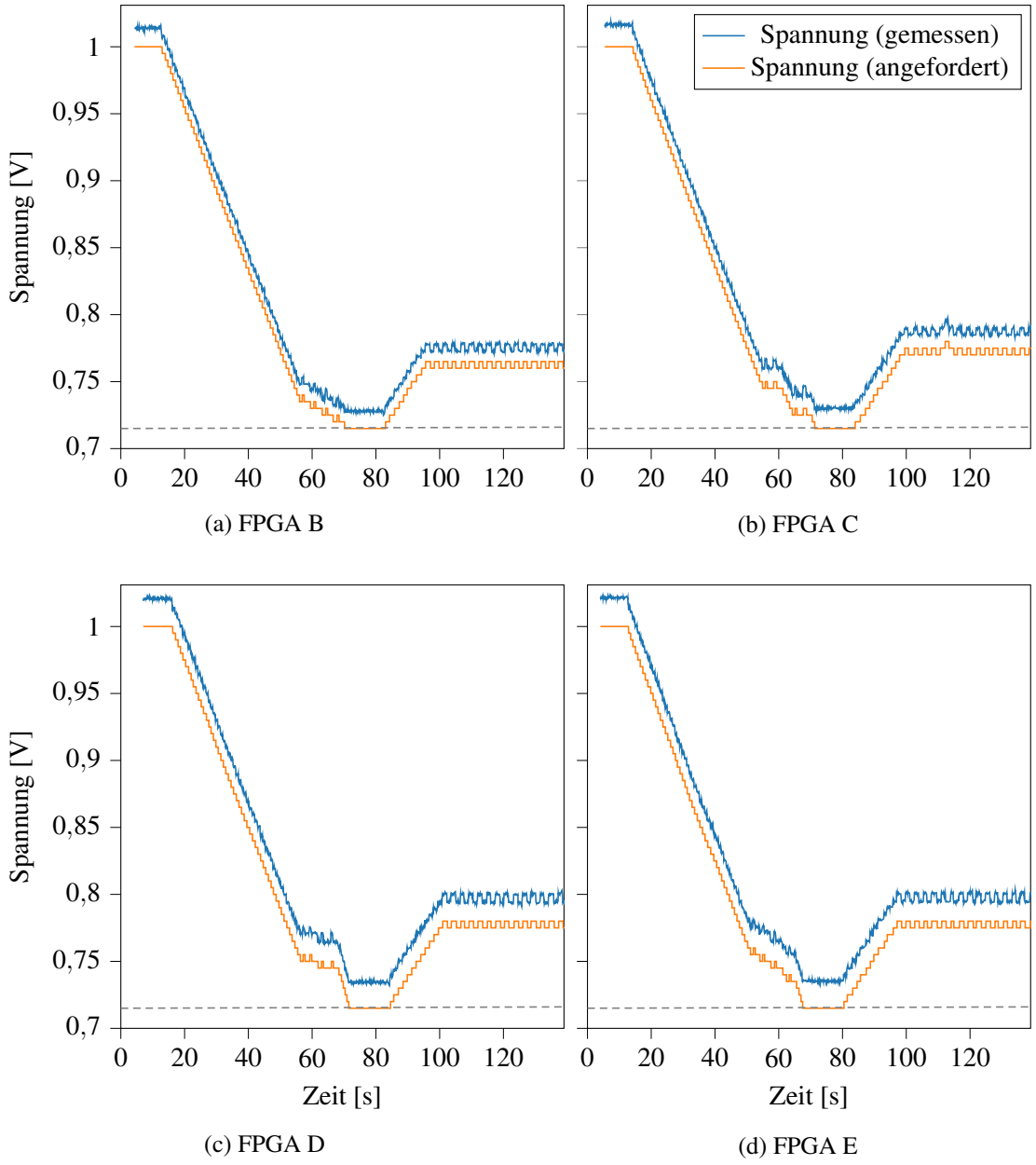


Abbildung 3.23: Verlauf der Spannung während der Kalibrierung und beginnendem Anwendungsbetrieb (Design A, FPGA A)

Dies wiederholt sich nun, sobald die Tests einer weiteren Sensorregion fehlschlagen. Für einige Sensorregionen wird auch bei Absenken der Versorgungsspannung bis auf die Data Retention Voltage kein Timing-Fehler festgestellt. Der Sensorwert wird dann bei anliegender Data Retention Voltage ausgelesen. In diesem Fall wird die Versorgungsspannung zum Auslesen der Sensoren nicht angehoben, da die Anwendungsschaltung in den betreffenden Sensorregionen bei anliegender Data Retention Voltage fehlerfrei funktioniert.

Abbildung 3.24 zeigt die Spannungsverläufe während der Kalibrierung für vier weitere FPGAs gleichen Typs, die mit dem identischen Bitfile initialisiert wurden. Es ist deutlich erkennbar, dass aufgrund von Prozessvariation und Alterung die kritischen Pfade bei einer anderen Versor-



(a) FPGA B

(b) FPGA C

(c) FPGA D

(d) FPGA E

Abbildung 3.24: Verlauf der Spannung während der Kalibrierung und beginnendem Anwendungsbetrieb; Alle FPGAs wurden mit demselben Bitfile (Design A) initialisiert, das auch für Abbildung 3.23 verwendet wurde.

gungsspannung erstmalig einen Timing-Fehler aufweisen. Die kritische Versorgungsspannung ist zudem nicht nur durch Die-to-Die-Variationen um einen Offset verschoben. Vielmehr unterscheiden sich durch OCVs und Alterung auch die relativen Abstände der kritischen Versorgungsspannungen der verschiedenen Pfade zueinander.

Nach Bestimmung des Sollwertes für alle Sensorregionen beginnt die Regelung der Versorgungsspannung per AVS. Dadurch steigt die Versorgungsspannung nun auf den neuen, im Anwendungsbetrieb verwendeten Wert. In den Messwerten bildet sich dieser Übergang zwischen Sekunde 82 und 112 ab. Danach schwankt die Versorgungsspannung aufgrund des Ausbleibens wesentlicher Störgrößen um einen Mittelwert von 0,843 V. Nach dem Systemstart benötigt die AVS-Sensorkalibrierung also etwa anderthalb Minuten. Bei der hier untersuchten prototypischen Implementierung wurde jedoch ausdrücklich keinen Wert auf eine kurze Dauer der Kalibrierung gelegt. Für eine entsprechende Optimierung gäbe es jedoch großes Potential. Insbesondere das Anheben der Versorgungsspannung von der Data Retention Voltage auf die Versorgungsspannung für den Anwendungsbetrieb könnte erheblich schneller erfolgen. Aber auch die Verweildauer auf den einzelnen Spannungsstufen während der Kalibrierung könnte verkürzt werden. In der prototypischen Implementierung wurden diese sehr lang gewählt, um Probleme durch das Zeitverhalten der Software auf dem PC zu vermeiden.

3.9.1 Reduktion der Verlustleistung durch AVS

Die Verlustleistung ergibt sich aus der Versorgungsspannung und der Stromstärke. Im vorigen Unterkapitel wurde gezeigt, dass die Versorgungsspannung durch das im Rahmen dieser Arbeit entwickelte AVS deutlich reduziert werden kann. Wie in Abschnitt 2.1 erläutert, ist der Stromfluss durch eine CMOS-Schaltung von der angelegten Versorgungsspannung abhängig. Im Folgenden wird die Reduktion der Verlustleistung durch AVS für die untersuchten Anwendungsdesigns A und B ausgewertet.

Design A – Dominanz der statischen Verlustleistung
Für das Design A wurde ein durchschnittliches Absinken des Stromflusses von 1 466 mA auf 633 mA beobachtet. Dieses überproportionale Absinken des Stromflusses liegt im erwarteten Bereich für ein Design, dessen Verhalten von der statischen Verlustleistung dominiert wird, da diese exponentiell von der Versorgungsspannung abhängig ist. Aus den gemessenen Werten für Versorgungsspannung und Stromfluss ergibt sich eine durchschnittliche Reduktion der Leistungsaufnahme von 1,49 W bei nomineller Versorgungsspannung auf 0,51 W bei aktivem AVS. Dies entspricht einer durchschnittlichen Energieersparnis von 66 %.

Design B – Dominanz der dynamischen Verlustleistung
Wie in Unterabschnitt 3.8.6 dargelegt, wird das Verhalten von Design B durch die dynamische Leistungsaufnahme dominiert. Da in diesem Fall das Verhalten bei Lastwechseln von großem Interesse ist, wurden solche ebenfalls untersucht. Die gemessenen Verläufe von Spannung, Stromstärke und sich ergebender Leistungsaufnahme mit aktivem AVS werden in Abbildung 3.25 dargestellt. Für die quantitative Auswertung wurden jedoch nur die von der dynamischen Verlustleistung dominierten Zeitabschnitte mit hoher Schaltaktivität ausgewertet. Dabei wurde eine Reduktion des Stromflusses um durchschnittlich 38 % beobachtet. Daraus ergibt sich ein Absinken der durchschnittlichen Leistungsaufnahme von 4,17 W auf 2,05 W. Das entspricht einer Energieersparnis von 51 %. Dies liegt im erwarteten Bereich, da die Versorgungsspannung quadratisch in die dynamische Verlustleistung eingeht. In Abbildung 3.25 ist ein sprunghafter Anstieg von Stromfluss und Leistungsaufnahme bei Beginn der Kalibrierung erkennbar. Dieser wird jedoch nicht von der Kalibrierung verursacht, sondern tritt immer (also auch ohne AVS) beim Übergang zwischen der Initialisierung des FPGAs mit dem Bitfile und dem normalen Betrieb

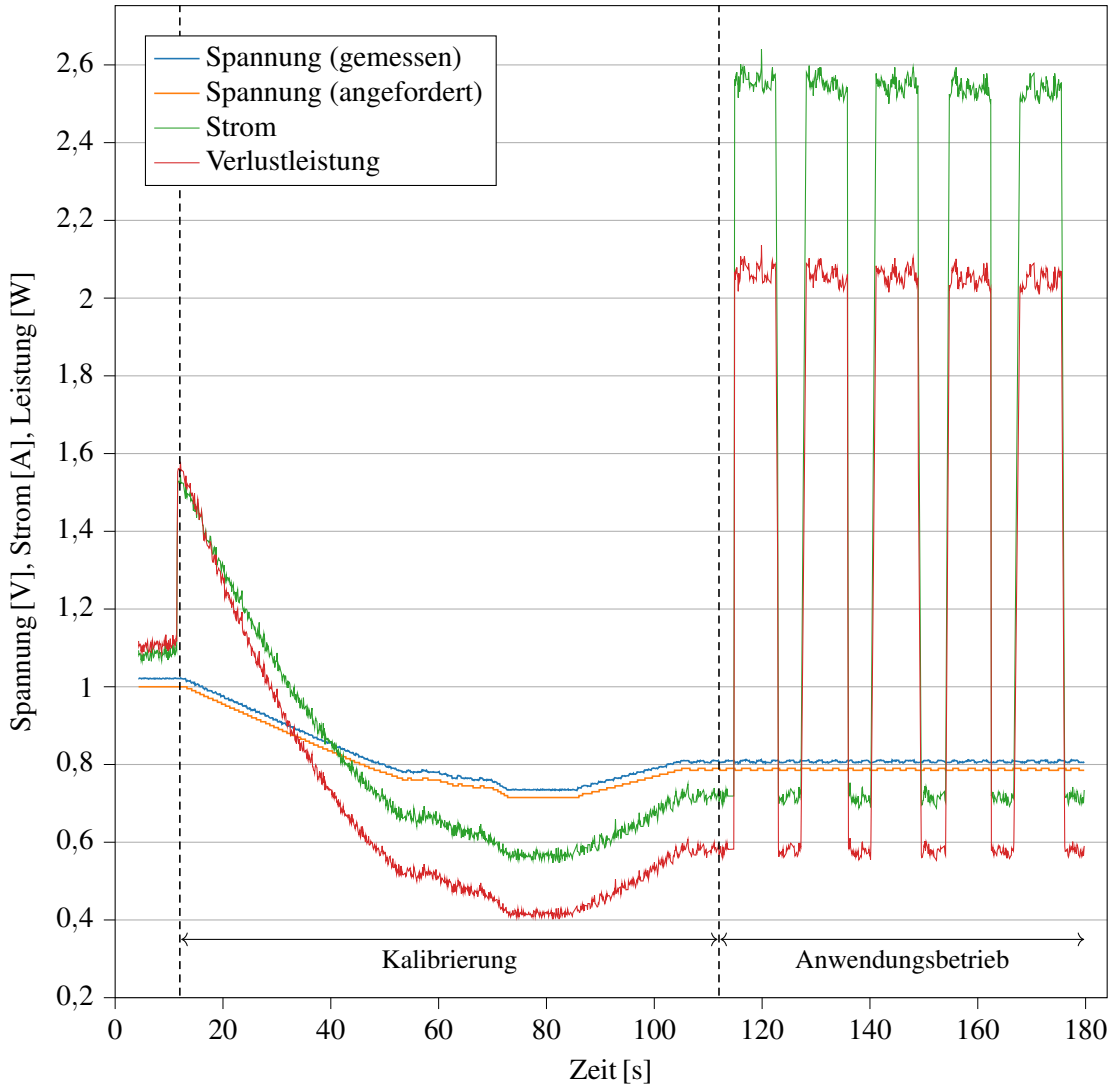


Abbildung 3.25: Versorgungsspannung, Strom und Leistungsaufnahme bei angewandtem AVS und Kalibrierungsprozedur (Design B, FPGA E)

auf. Der Autor dieser Arbeit vermutet, dass für die Initialisierung eine deutlich geringere Taktfrequenz genutzt wird. Das Anlegen der Taktfrequenz an das Clock-Netzwerk nach Abschluss der Initialisierung würde den plötzlichen Anstieg des Stromflusses erklären. Die Initialisierung dauert insgesamt etwa 20 Sekunden. In Abbildung 3.25 wird jedoch nur ein kurzer Zeitabschnitt davon gezeigt.

3.9.2 Untersuchung des Einflusses der Prozessvariation

Um den Einfluss der Prozessvariationen auf AVS zu berücksichtigen, wurde das AVS-System auf fünf FPGAs des gleichen Typs experimentell untersucht. Konkret wurden fünf baugleiche Xi-

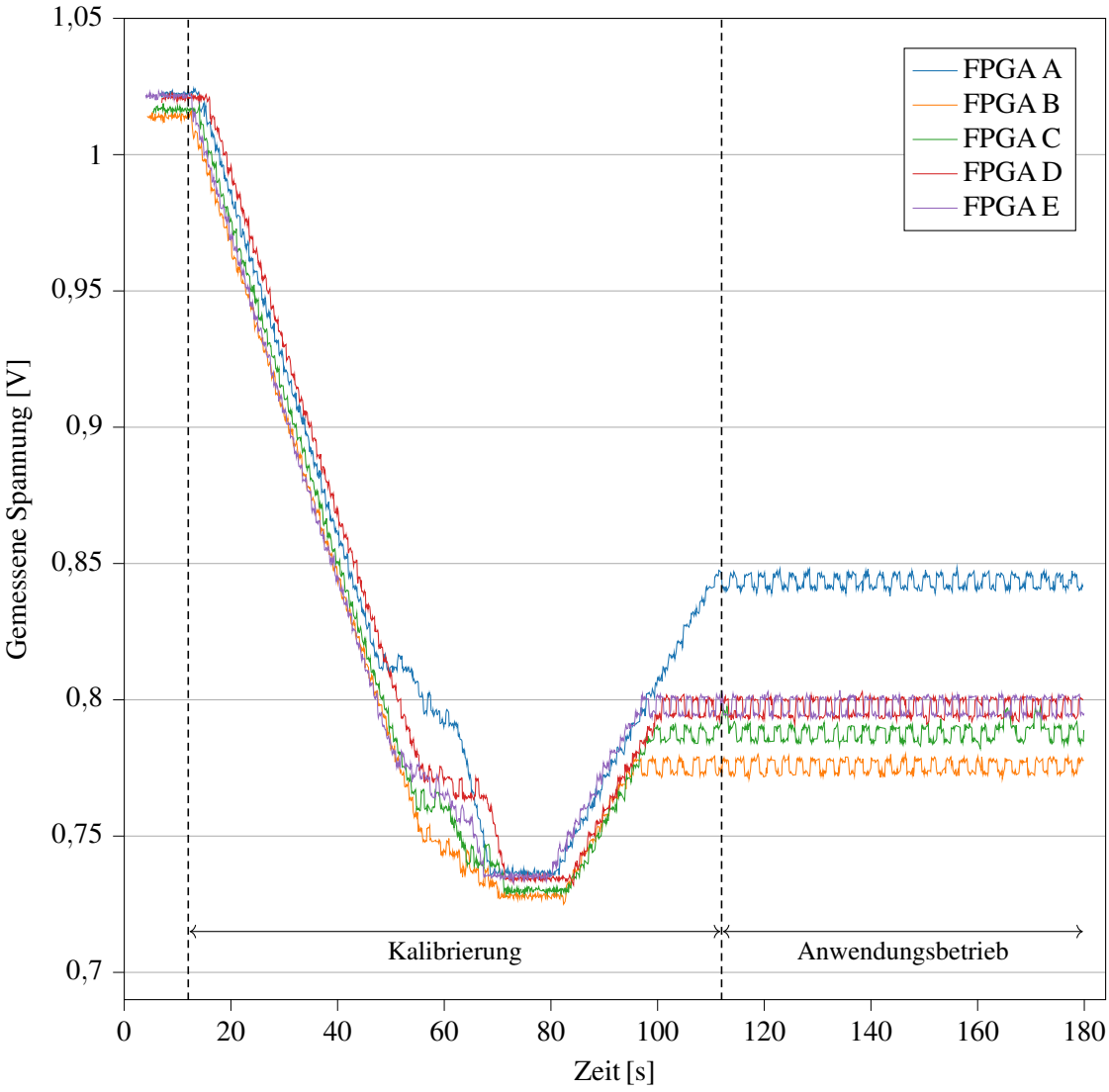


Abbildung 3.26: Bei angewandtem AVS stellen sich auf Instanzen von baugleichen FPGAs verschiedene Versorgungsspannungen ein.

linx ML605 Evaluation Boards verwendet, die jeweils einen FPGA genau gleicher Spezifikation (Xilinx Virtex-6 *XC6VLX240T-1FFG1156*) enthalten. Um Einflüsse des Designs und der EDA-Tools auszuschließen, wurde jeweils das exakt gleiche Bitfile auf jeden der FPGAs geladen. Darüber hinaus wurden alle Tests bei vergleichbarer Umgebungstemperatur durchgeführt. Da alle anderen Einflussgrößen weitestgehend konstant gehalten wurden, kann davon ausgegangen werden, dass die Streuung der Messwerte in den Tabellen 3.3 und 3.4 durch Prozessvariationen und die unterschiedliche Alterung der FPGAs verursacht wird. Abbildung 3.26 zeigt die gemessenen Spannungsverläufe während der Kalibierung und dem beginnenden Anwendungsbetrieb

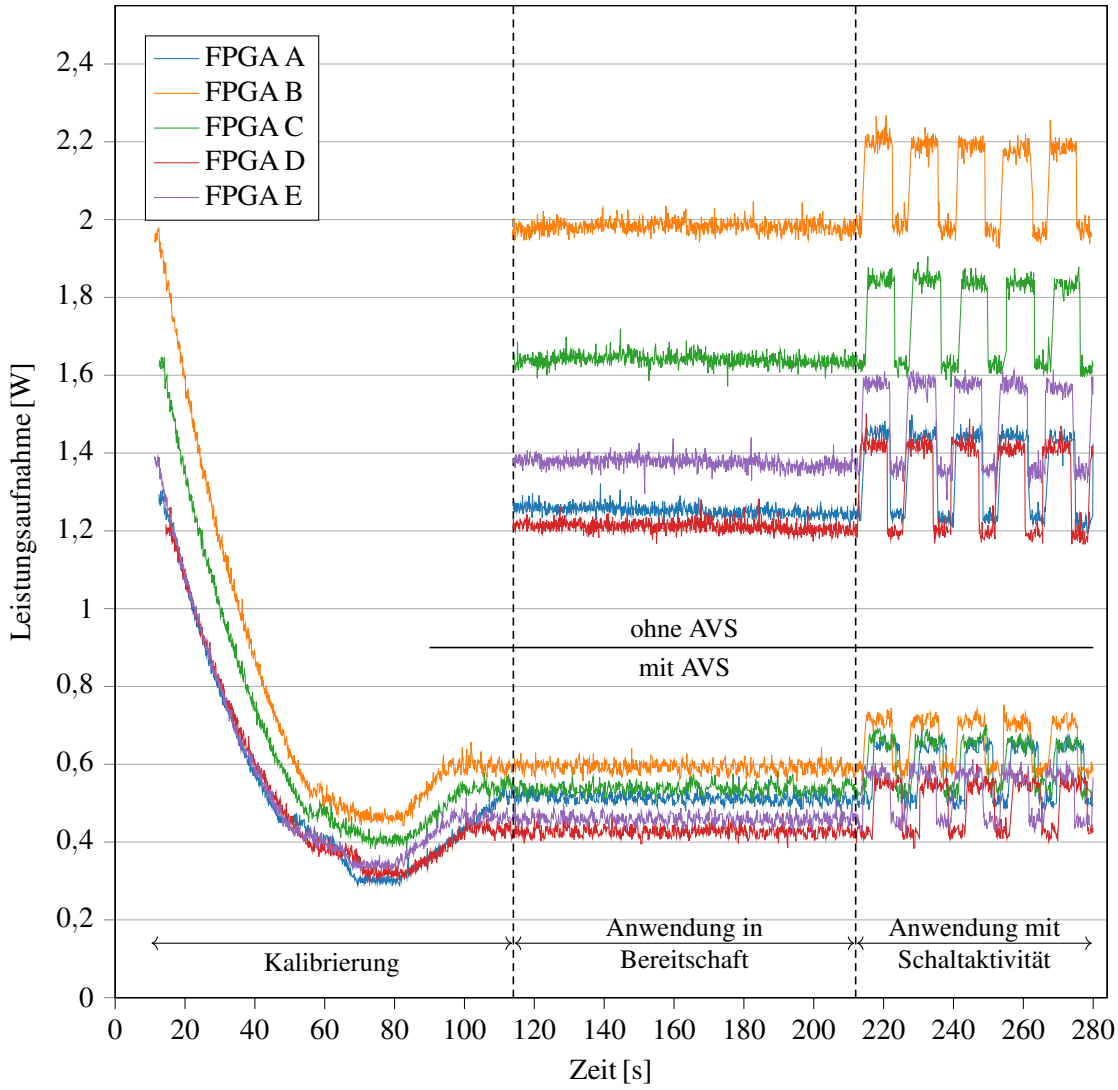


Abbildung 3.27: Leistungsaufnahme verschiedener Instanzen des gleichen FPGA-Modells mit und ohne AVS (Design A)

für alle fünf FPGAs. Die durch das AVS gewählten Versorgungsspannungen unterscheiden sich deutlich.

Abbildung 3.27 zeigt die daraus resultierenden Verlustleistungen im Vergleich zum Betrieb mit der nominellen Versorgungsspannung. Interessanterweise ist das FPGA B, welches mit aktivem AVS die niedrigste Spannung benötigt, keineswegs jenes, das auch die niedrigste Leistungsaufnahme zeigt. Dies ist der Fall, da ein Chip, dessen Versorgungsspannung mit AVS sehr weit abgesenkt werden kann, ein vergleichsweise 'schneller' Chip ist – nur dann ist der Chip trotz weit abgesenkter Versorgungsspannung noch in der Lage, die Timing-Anforderungen zu erfüllen. Einige wesentliche Ausprägungen der Prozessvariationen, die zu 'schnellen' Chips füh-

ren, verursachen zugleich aber auch eine vergleichsweise hohe Leistungsaufnahme. Da für sehr schnelle Chips, die bei nomineller Versorgungsspannung eine große Verlustleistung aufweisen, die Spannung am stärksten abgesenkt werden kann, unterscheidet sich die Leistungsaufnahme der einzelnen FPGAs bei angewandtem AVS weniger stark. Auch dies wird in Abbildung 3.27 klar deutlich.

3.9.3 Zuverlässigkeit

Im Rahmen der experimentellen Evaluation wurde auch die Zuverlässigkeit geprüft. Dazu werden die Anwendungsdesigns während der AVS-Anwendung mit pseudozufälligen Eingangsvektoren beaufschlagt. Diese werden durch ein geeignetes LFSR erzeugt. Mit einem weiteren LFSR wurden die Ausgänge komprimiert und eine Signatur ermittelt. Diese wird mit einem vorher ermittelten Referenzwert verglichen. Während aller Experimente funktionierte das System stets fehlerfrei. Wie in Unterabschnitt 3.5.1 gezeigt, sind auf zufälligen Eingangsvektoren basierende Untersuchungen nicht geeignet, um potentielle Timing-Fehler durch ein zu aggressives AVS auszuschließen. Aus diesem Grund werden sie im hier vorgeschlagenen Konzept auch nicht verwendet. Die ohnehin durchgeführten ausgiebigen Tests zur Bestimmung der Leistungsaufnahme wurden aber genutzt, um durch die Überwachung der Ausgangsvektoren eine zusätzliche Prüfung auf korrekte Funktionsweise vorzunehmen. Der Autor hält dies für wichtig, da beim Test mit pseudozufälligen Eingangsvektoren prinzipiell auch solche Fehler auftreten könnten, die nicht erwartet wurden und für die folglich kein Test Pattern vorgesehen ist.

Tabelle 3.3: Messwerte für Anwendungsdesign A (Leakage dominiert die Verlustleistung); Alle Werte sind Durchschnittswerte.

	Versorgungs-spannung ohne AVS	Versorgungs-spannung mit AVS	Stromstärke ohne AVS	Stromstärke mit AVS	Leistungs-aufnahme ohne AVS	Leistungs-aufnahme mit AVS	Relative Reduktion der Verlustleistung
FPGA A	1,020 V	0,843 V	1,228 A	0,607 A	1,252 W	0,511 W	59,12 %
FPGA B	1,014 V	0,776 V	1,957 A	0,764 A	1,984 W	0,592 W	70,15 %
FPGA C	1,016 V	0,788 V	1,616 A	0,682 A	1,642 W	0,537 W	67,27 %
FPGA D	1,024 V	0,797 V	1,184 A	0,536 A	1,212 W	0,427 W	64,74 %
FPGA E	1,023 V	0,798 V	1,346 A	0,575 A	1,377 W	0,459 W	66,67 %

Tabelle 3.4: Messwerte für Anwendungsdesign B (dominante dynamische Verlustleistung). Alle Werte sind Durchschnittswerte.

	Versorgungs-spannung ohne AVS	Versorgungs-spannung mit AVS	Stromstärke ohne AVS	Stromstärke mit AVS	Leistungs-aufnahme ohne AVS	Leistungs-aufnahme mit AVS	Relative Reduktion der Verlustleistung
FPGA A	1,022 V	0,836 V	3,810 A	2,585 A	3,896 W	2,161 W	44,52 %
FPGA B	1,015 V	0,779 V	4,633 A	2,673 A	4,703 W	2,082 W	55,72 %
FPGA C	1,018 V	0,791 V	4,219 A	2,533 A	4,293 W	2,006 W	53,28 %
FPGA D	1,021 V	0,799 V	3,826 A	2,457 A	3,906 W	1,962 W	49,76 %
FPGA E	1,021 V	0,808 V	3,958 A	2,528 A	4,041 W	2,043 W	49,47 %

3.9.4 Einordnung im Vergleich zum Stand der Technik

Um dem/der Leser:in einen möglichst guten Überblick über die vorliegende Arbeit im Vergleich zum Stand der Technik zu geben, werden ausgewählte Ansätze in Tabelle 3.5 verglichen. Eine ausführliche Erläuterung und Einordnung aller in der Tabelle aufgeführten und weiterer Ansätze wird in Abschnitt 3.3 vorgenommen. Die quantitativen Angaben für die Energieersparnis von AVS-Ansätzen sind nach Sicht des Autors dieser Arbeit jedoch mit einer erheblichen Unsicherheit behaftet und nur begrenzt vergleichbar. Die Gründe dafür werden bereits an verschiedenen Stellen in dieser Arbeit diskutiert, hier aber noch einmal zusammenfassend aufgeführt.

- Ein sehr grundlegendes Problem bei der Vergleichbarkeit quantitativer Angaben zur Energieersparnis ist die Größe und Auswahl der Stichprobe von FPGAs. Aus praktischen Gründen wird im universitären Umfeld typischerweise mit eher kleinen Stichproben gearbeitet. Oft sogar mit nur einem einzigem FPGA. Selbst wenn eine gewisse Anzahl von FPGAs verwendet wird, kann üblicherweise nicht sichergestellt werden, dass die Stichprobe für die Gesamtheit der produzierten FPGAs eines Typs repräsentativ ist. Somit unterliegen die quantitativen Angaben einer gewissen Schwankung durch die zufällige Wahl der Stichprobe. Dies betrifft ausdrücklich auch die in der vorliegenden Arbeit ermittelten Messwerte.
- Wie in Abschnitt 3.7 erläutert, benötigen AVS-Systeme ein residuales Guard Band um einen fehlerfreien Betrieb zu sicherzustellen. Viele Arbeiten verwenden ein willkürlich gewähltes residuales Guard Band. Dies ist häufig deutlich zu optimistisch gewählt. Teilweise wird sogar komplett auf ein residuales Guard Band verzichtet. Bei einem zu klein gewählten residualen Guard Band wird scheinbar eine größere Energieersparnis erzielt. Diese basiert jedoch auf einem fehleranfälligem VF-Arbeitspunkt und lässt sich deshalb nicht praktisch realisieren.
- Die Umgebungsbedingungen während der Experimente beeinflussen die Ergebnisse. So beeinflussen die Raumtemperatur und die Ausgestaltung von Luftzufuhr und Lüfter die Temperatur des Chips und dadurch auch den gewählten VF-Arbeitspunkt.
- Auch die Wahl des Anwendungsdesigns, anhand dessen ein AVS-Ansatz demonstriert wird, beeinflusst die quantitativen Ergebnisse. Einerseits wird ein eventueller Timing-Slack des Anwendungsdesigns durch AVS zur Wahl eines günstigeren VF-Arbeitspunktes genutzt. Dadurch schwanken die Ergebnisse mit dem Timing-Slack des Anwendungsdesigns, sofern dieses Verhalten nicht durch geeignete Constraints verhindert wird. Andererseits bestimmt das Anwendungsdesign das Verhältnis von dynamischer und statischer Verlustleistung. Da diese, wie in Unterabschnitt 2.1.3 beschrieben, verschieden stark von einer Absenkung der Versorgungsspannung beeinflusst werden, bestimmt die Wahl des Anwendungsdesigns mittelbar auch die erreichbare Energieersparnis, die ein bestimmter VF-Arbeitspunkt ermöglicht. In der vorliegenden Arbeit wurden deshalb zwei Designs genutzt um die Bandbreite der möglichen Energieersparnis abzubilden.

Tabelle 3.5: Vergleich mit ausgewählten Arbeiten aus dem Stand der Technik [✓ = ja, o = mit Einschränkungen, ✗ = nein]

Referenz	Ansatz	Energie-ersparnis	Residuales Guard Band ermittelt	Nicht von Zufallsdaten abhängig	OCV des Clock Tree berücksichtigt	Alterung berück-sichtigt	Temperatur-einfluss ausgeregelt	Kommentar
Offline-Ex-Situ-Verfahren								
Maragos et al. [A 74]	Kartierung des FPGA und optimierte Platzierung kritischer Pfade	N/A[3.1]	✗	N/A[3.2]	✗	✗	✗	
Offline-In-Situ-Verfahren								
Li et al. [A 78]	Rekonfiguration kritischer Pfade in ROs	N/A	✗	✓	✗	✓	✗	Zielt eigentlich auf die Messung der Alterung ab
Ahmed et al. [A 80]	Replikation kritischer Pfade	33 %	o	✓	✓	o	✗	Abweichendes Fanout der Pfad-Replikation
Zhao et al. [A 81]	Replikation kritischer Pfade	40 %	o	✓	✓	o	o	Abweichendes Fanout der Pfad-Replikation
Online-Ex-Situ-Verfahren								
Chow et al. [A 83]	Inverterkette	54 %	✗	✗	✗	o	✓	
Nunez-Yanez et al. [A 85]	Inverterkette	N/A	✗	✓	✗	o	✓	Länge der Inverterkette nach Logiktiefe gewählt
Maragos et al. [A 86]	Inverterkette	16-27 %	✗	✓	✗	o	✓	Länge der Inverterkette per STA bestimmt
Online-In-Situ-Verfahren								
Brant et al. [A 93]	Razor	N/A[3.1]	N/A[3.3]	✓	✓	✓	✓	Overhead durch OR-Tree und große Hold Time
Nunez-Yanez et al. [A 94], [A 95], [A 96], [A 97]	Monitoring-Flipflops ohne Fehlerkorrektur	bis zu 70 %	✗	✗	✓	✓	✓	Prinzipbedingte Fehler; siehe Abschnitt 3.3
Levine et al. [A 98], [A 99], [A 100]	Monitoring-Flipflops ohne Fehlerkorrektur	33,5 %	✗	✗	✓	✓	✓	Prinzipbedingte Fehler; siehe Abschnitt 3.3
Diese Arbeit	Online-ROs mit Kalibrierung	51-66 %	✓	✓	✓	✓	✓	

[3.1] Der VF-Arbeitspunkt wird ausschließlich durch eine Manipulation der Taktfrequenz beeinflusst. Eine Senkung der Leistungsaufnahme des FPGA findet nicht statt. Der Ansatz ließe sich aber prinzipiell auch für eine Absenkung der Versorgungsspannung nutzen.

[3.2] Die eigentliche Kartierung nutzt keine Zufallsdaten. Für die Anpassung des VF-Arbeitspunkts werden vermutlich Zufallswerte genutzt.

[3.3] Durch die Fehlerkorrektur benötigen Ansätze nach dem Razor-Prinzip kein residuales Guard Band.

3.10 Zwischenfazit

Die realen Eigenschaften eines ICs können zur Designzeit nur geschätzt werden, da sie einer erheblichen Variabilität unterliegen. Diese wird durch Prozessvariationen sowie den Einfluss von Temperatur, Alterung und Schwankungen der Versorgungsspannung verursacht. Um die geforderte Zuverlässiges dennoch sicherzustellen, wird der VF-Arbeitspunkt durch die Addition von Guard Bands für Worst-Case-Bedingungen ausgelegt. Im typischen Fall ist dieser VF-Arbeitspunkt jedoch suboptimal und führt zu einer höheren Verlustleistung. Adaptive Voltage Scaling (AVS) ist ein Ansatz, um durch Laufzeitwissen günstigere VF-Arbeitspunkte zu ermöglichen und somit Energie zu sparen. In diesem Kapitel wurden mehrere Beiträge zu diesem Thema geleistet.

Mit einer Simulation wurde gezeigt, dass **Monitoring-Flipflop-basierte AVS-Ansätze** einen fehlerfreien Betrieb nicht garantieren können und die konkrete Fehlerrate von den verarbeiteten Daten abhängig ist. Weiterhin wurde belegt, dass auch Fehler auftreten, bevor von den Monitoring-Flipflops eine Annäherung an die kritische Versorgungsspannung festgestellt wurde. Auch das Auftreten solcher nicht detektierbarer Fehler ist von den statistischen Eigenschaften der verarbeiteten Daten abhängig. Da Monitoring-Flipflop-basierte AVS-Ansätze jedoch erhebliche Vorteile aufweisen, wie etwa einen geringen Overhead und eine vielversprechende Reduktion der Leistungsaufnahme, sollten sie als eine Möglichkeit des Approximate Computing betrachtet werden. Dazu sind weiterführende Untersuchungen beispielsweise bezüglich der schaltungsabhängigen Charakteristik der in Kauf genommenen Fehler notwendig.

Sensorbasierte AVS-Systeme wurden analysiert und aufgezeigt, dass zu deren Realisierung neben dem Design des eigentlichen Sensors weitere wesentliche Fragen zu klären sind, die in der Literatur häufig vernachlässigt werden. Als solche wurden die Sensorkalibrierung und die verwendeten Constraints zur Sensorplatzierung identifiziert.

Es wurde eine **Offline-Sensorkalibrierungsprozedur** vorgestellt. Sie basiert auf der schrittweisen Absenkung der Versorgungsspannung, wobei auf jeder Spannungsstufe mittels geeigneter Path Delay Tests geprüft wird, ob bei der jeweiligen Versorgungsspannung Timing-Fehler auftreten. Dieses Vorgehen ermöglicht es, unabhängig von der Charakteristik der zu verarbeitenden Anwendungsdaten die kritische Versorgungsspannung direkt im Datenpfad zu ermitteln. Dabei können im Gegensatz zu Critical Path Replicas auch OCVs und die Variabilität des Clock Tree berücksichtigt werden.

Als **Online-Sensoren** wurden Ringoszillatoren (ROs) genutzt. Die Anforderungen für den speziellen Anwendungsfall als Online-Sensor für AVS wurden herausgearbeitet und daraus abgeleitet, dass andere Optimierungsziele und Implementierungsstrategien benötigt werden als sie für ROs in anderen Anwendungsszenarien üblich sind. Insbesondere ist es durch die ohnehin notwendige Kalibrierung vorteilhaft, auf eine Platzierung mittels Hard Macro zu verzichten. Stattdessen werden lediglich eine Struktur und eine räumliche Begrenzung vorgegeben, innerhalb derer das EDA-Tool die Sensorelemente frei platzieren kann. Dadurch werden Placement und Routing Congestions vermieden. In der experimentellen Auswertung wurde zudem eine bessere Qualität der Sensormesswerte für den gegebenen Anwendungsfall festgestellt.

Zur experimentellen Auswertung wurde das entworfene AVS-System für zwei Anwendungsdesigns prototypisch umgesetzt. Dabei handelt es sich einerseits um ein vergleichsweise kleines

Anwendungsdesign, bei dem die Leistungsaufnahme durch die Leakage Power dominiert wird und andererseits um ein größeres Design mit hoher Schaltaktivität, bei dem die dynamische Verlustleistung dominiert. Beide Designs wurden auf fünf baugleichen FPGAs getestet, um den Einfluss der Prozessvariation und Alterung zu berücksichtigen. Die Versorgungsspannung konnte für Design A und B im Durchschnitt um 21,5 % beziehungsweise 21,3 % reduziert werden. Dadurch ergab sich für das durch die dynamische Verlustleistung dominierte Design B eine um durchschnittlich 51 % geringere Leistungsaufnahme. Für das kleinere Design A, das durch die Leakage Power dominiert wird, ergibt sich sogar eine Energieersparnis von 66 %.

4 Für FPGA optimierte approximierte Multiplizierer

4.1 Motivation

Der Energiebedarf eines Very Large Scale Integration (VLSI)-Systems ist von mehreren Einflussgrößen abhängig. Jede dieser Einflussgrößen stellt gleichzeitig einen Ansatzpunkt zur diesbezüglichen Optimierung dar. Neben der Versorgungsspannung, deren Optimierung per Adaptive Voltage Scaling (AVS) in Kapitel 3 diskutiert wurde, fällt auch der Schaltaktivität, sowie der Anzahl und Dimensionierung der benötigten Transistoren und des Routings ein wesentlicher Einfluss zu. Gelingt es, eine Operation mit weniger Logikelementen oder einer geringeren Schaltaktivität zu realisieren, kann die Leistungsaufnahme gesenkt werden. Gesucht wird also eine Möglichkeit, eine benötigte Operation unter bestimmten Randbedingungen mit einer minimalen Anzahl an Logikelementen und einer minimalen Anzahl an Schaltvorgängen zu realisieren.

Für arithmetische Operationen sind solche Randbedingungen beispielsweise:

- Dynamikumfang
- Latenz
- Durchsatz

Der Ressourcenbedarf wird stark von diesen Randbedingungen bestimmt. Beispielsweise sind in einer digitalen Zahlendarstellung die betragsmäßig kleinste und größte darstellbare Zahl durch das Zahlenformat festgelegt. Für die hier betrachteten Festkommadarstellungen werden beide durch die Anzahl der Bitstellen und die definierte Anzahl an Nachkommastellen determiniert. Dadurch hat der geforderte Dynamikumfang einen erheblichen Einfluss auf den Ressourcenbedarf. Als Beispiel sei hier die Multiplikation genannt. Soll diese in einem Taktzyklus erfolgen, beträgt die Anzahl der Logikelemente in Abhängigkeit von der Zahl der Bitstellen asymptotisch für viele verbreitete Multiplizierer-Designs $O(n^2)$ [A 145]. Folglich wird die Anzahl der Bitstellen so klein wie möglich gewählt, um die Anforderungen einer Anwendung gerade noch zu erfüllen. Anders ausgedrückt ist die Wahl der Wortbreite in Bit eine Abwägung zwischen Genauigkeit bzw. akzeptiertem Quantisierungsfehler und Ressourcenbedarf. In Anwendungsbereichen mit einer gewissen Fehlertoleranz der Anwendung bei gleichzeitig hoher geforderter Rechenleistung, wie beispielsweise Machine Learning (ML) und insbesondere bei der Inferenz von Deep Neural Networks (DNNs), wird häufig eine geringe Bitbreite gewählt und dafür eine leicht schlechtere Erkennungsrate in Kauf genommen [A 146], [A 147]. Neben der Wahl der Wortbreite in Bit gibt es weitere Möglichkeiten, durch die Wahl einer Genauigkeit bzw. die Inkaufnahme bestimmter Fehler den Ressourcenbedarf zu beeinflussen. Diese werden als *Approximate Computing* bezeichnet. Thema dieser Arbeit ist die Steigerung der Energieeffizienz Field Programmable Gate Array (FPGA)-basierter Anwendungen. Folglich steht nicht Approximate Computing im Allgemeinen, sondern seine Anwendung auf FPGAs im Fokus, wobei insbesondere eine Senkung des Energiebedarfs erreicht werden soll.

Ein wichtiger Grundbaustein vieler Anwendungen, die den Einsatz von Approximate Computing erlauben, sind Integer- beziehungsweise Festkomma-Multiplizierer. Der Autor dieser Arbeit hat eine Methodik zum Entwurf von approximierten Multiplizierern veröffentlicht, welche für

FPGAs optimiert sind [B 4]. In einer Diskussion mit Vojtech Mrazek, einem der Autor:innen von [A 148], wurde deutlich, dass unterschiedliche Annahmen und Definitionen bei der Evaluation von approximierten Multiplizierern zu erheblichen Abweichungen, beispielsweise um den Faktor zwei beim Delay, führen. In dieser Arbeit wird deshalb eine Methodik für eine umfassende Evaluation von approximierten arithmetischen Designs entwickelt und beschrieben. Diese wertet fünf verschiedene Gütefunktionen aus. Bezüglich des Ressourcenbedarfs wird neben den üblichen Größen *Delay* und *Logikressourcen* auch die *Verlustleistung* berücksichtigt.

Im Folgenden werden die Anforderungen und Optimierungsziele für einen solchen Approximate-Computing-Ansatz diskutiert. Anschließend wird ein neuer Ansatz für approximierte Multiplizierer vorgestellt und gezeigt, dass er bezüglich der gewählten Gütefunktion Pareto-optimal ist.

4.2 Anforderungen und Gütemaße für Approximate Computing

Approximate Computing ist eine vielversprechende Möglichkeit, Energie und andere Ressourcen zu sparen. Jedoch nimmt Approximate Computing willentlich eine gewisse Ungenauigkeit in Kauf. Folglich ist es nur für Anwendungen geeignet, die eine gewisses Maß an Ungenauigkeit tolerieren. Beispielsweise ermöglicht die begrenzte Präzision der menschlichen Sinnesorgane einen Einsatz von Approximate Computing. Typischerweise wird es in Verbindung mit einer (ohnehin angewandten) Verkürzung der Wortbreite der Datenwörter, welche aus selbigen Grund möglich ist, genutzt.

Da die Toleranz gegenüber Ungenauigkeiten häufig durch die menschliche Wahrnehmung bedingt ist, sollten Approximate-Computing-Designs deren Charakteristika, die sogenannte Psychophysik, berücksichtigen. Entscheidend dafür ist insbesondere die Wahrnehmungsschwelle, also die kleinste wahrnehmbare Abweichung von einem gegebenen Ausgangswert. Nach Webers Gesetz [A 149] gilt:

$$k = \frac{\Delta S}{S} \tag{17}$$

Die kleinste wahrnehmbare Abweichung ΔS von einem Stimulus S ist folglich kein absoluter Wert. Vielmehr ist sie proportional zur Größe des Stimulus. Da Webers Gesetz also gewissermaßen das Toleranzverhalten der menschlichen Wahrnehmung gegenüber Ungenauigkeiten widerspiegelt, sollte dieser Zusammenhang sowohl beim Entwurf von Approximate-Computing-Designs als auch bei der Wahl der Gütemaße eine zentrale Überlegung darstellen. Einige der am weitesten verbreiteten Metriken berücksichtigen diesen Sachverhalt jedoch nicht. Dies trifft insbesondere auf den Mean Absolute Error (MAE), auch Mean Error Distance (MED) genannt, und die Normalized Mean Error Distance (NMED) zu. Beide Metriken werden in [A 150] diskutiert.

Die MED ist als

$$MED = \sum_{\forall i} \frac{\left| O^{(i)}_{exakt} - O^{(i)}_{approx} \right|}{N} \tag{18}$$

definiert, wobei $O^{(i)}_{exakt}$ das Ergebnis bei exakt ausgeführter Operation und $O^{(i)}_{approx}$ der tatsächliche Ausgabewert des Approximate-Computing-Designs jeweils für den i-ten Eingangsvektor

sind. N ist die Anzahl aller möglichen Eingangsvektoren. Ein Eingangsvektor definiert dabei alle Eingangssignale der exakt ausgeführten Operation. Im Falle eines $n \cdot n$-Bit-Multiplizierers hat der Eingangsvektor also $2 \cdot n$ Bitstellen und es gilt $N = 2^{2n}$.

Die NMED dagegen normiert die auftretenden Fehler gegen den größtmöglichen Ausgangswert der exakten Operation unter Berücksichtigung aller möglichen Eingangsvektoren.

$$NMED = \frac{MED}{\max\limits_{\forall i}\left(O^{(i)}_{exakt}\right)} \tag{19}$$

Dadurch wird es möglich, Approximate-Computing-Techniken, die für verschiedene Wortbreiten implementiert wurden, zu vergleichen. Wenn Designs der gleichen Wortbreite verglichen werden, unterscheiden sich beide Metriken lediglich durch eine Konstante. Somit sind sie in ihrer Aussagekraft äquivalent. Da hier nur approximierte Designs gleicher Wortbreite verglichen werden, wird im weiteren Verlauf dieser Arbeit nur die NMED berücksichtigt.

Analog zur NMED, kann auch der größte auftretende Fehler auf den größtmöglichen Ausgangswert der exakten Operation normiert werden. Die sich ergebende Metrik wird in dieser Arbeit als Worst-Case Normalized Error Distance (WCNED) bezeichnet.

$$WCNED = \frac{\max\limits_{\forall i}\left|O^{(i)}_{exakt} - O^{(i)}_{approx}\right|}{\max\limits_{\forall i}\left(O^{(i)}_{exakt}\right)} \tag{20}$$

Durch die Normierung mit dem größtmöglichem Wert ähneln diese Metriken in ihrem Verhalten in gewisser Hinsicht dem Peak Signal-to-Noise Ratio (PSNR). Aus Webers Gesetz (vgl. Gleichung 17) ergibt sich aber, dass eine Metrik in Anlehnung an das Signal-to-Noise Ratio (SNR) besser geeignet ist. Dies trifft nicht nur für Anwendungen zu, die auf die menschliche Wahrnehmung abzielen, sondern auch auf viele technische Anwendungen, für die das SNR ein bedeutender Kennwert ist.

Das SNR selbst ist allerdings erheblich von der Anwendung und sogar von den konkret zu verarbeitenden Daten abhängig. Für einen Basis-Baustein wie einen Multiplizierer wird in der vorliegenden Arbeit daher in Anlehnung an [A 151] der Mean Relative Error (MRE) als Metrik bevorzugt. Diese wird als Grundlage für Design-Entscheidungen und als primäres Vergleichskriterium zu anderen veröffentlichten Ansätzen herangezogen. Weitere gebräuchliche Gütemaße werden für einen umfassenden Vergleich jedoch ebenfalls bestimmt und diskutiert. Im Folgenden werden die verschiedenen Gütemaße definiert.

Der Relative Error (RE) ergibt sich analog zu Gleichung 17, indem die Abweichung vom exakten Ergebnis durch selbiges geteilt wird. Für ein exaktes Ergebnis mit dem Wert Null ergibt sich dadurch allerdings eine undefinierte Metrik. Ein möglicher Ausweg wäre eine asymptotische Betrachtung: Geht der Nenner eines Bruchs gegen Null, geht der Wert des Bruchs gegen Unendlich. Da beispielsweise ein Fehler in einer Audio-Berechnung in absoluter Stille tatsächlich extrem störend wäre, ergibt diese Interpretation durchaus Sinn. Allerdings würde diese Vorgehensweise den praktischen Nutzen einer Metrik stark einschränken. Deshalb wird im angesprochenen Fall mit dem betragsmäßig kleinsten darstellbaren Zahlenwert ungleich Null normiert.

Für Operationen auf ganzzahligen Werten ist das die Eins. Somit ergibt sich der RE für den i-ten Eingangsvektor als:

$$RE_i = \frac{\left| O^{(i)}_{exakt} - O^{(i)}_{approx} \right|}{\max\left(O^{(i)}_{exakt}, 1 \right)} \tag{21}$$

Daraus ergeben sich zwei Metriken von entscheidender Wichtigkeit für die Beurteilung von Approximate-Computing-Designs: der MRE und der Worst-Case Relative Error (WCRE).

$$MRE = \frac{\sum_{\forall i} RE_i}{N} \tag{22}$$

ist der Mean Relative Error (MRE). Um eine vom konkreten Anwendungs-Datenstrom unabhängige Metrik zu berechnen, wird eine vollständige Simulation aller möglichen Eingangswerte verwendet. Wo dies aufgrund der mit der Wortlänge exponentiell wachsenden Anzahl von möglichen Eingangswerten nicht realisierbar ist, werden gleichverteilte Zufallswerte verwendet. Neben dem gemittelten Verhalten, wie vom MRE beschrieben, ist auch das ungünstigste Verhalten eines Designs von großem Interesse. Dieses wird durch den Worst-Case Relative Error (WCRE) beschrieben:

$$WCRE = \max_{\forall i} (RE_i) \tag{23}$$

Eine weitere gebräuchliche Metrik ist die Error Rate (ER). Diese gibt die Wahrscheinlichkeit an, einen vom Ergebnis der exakten Operation abweichenden Ausgangswert zu erhalten.

$$ER = P\left(O^{(i)}_{exakt} \neq O^{(i)}_{approx} \right) \tag{24}$$

Im Gegensatz zu den zuvor betrachteten Gütemaßen wird hier das Ausmaß der Fehler vernachlässigt, was die Aussagekraft der Metrik stark einschränkt. Um einen möglichst umfassenden Vergleich der verschiedenen Approximate-Computing-Techniken zu ermöglichen, wird sie dennoch betrachtet.

4.3 Stand der Wissenschaft und Technik

In diesem Kapitel wird der Stand der Wissenschaft und Technik bezüglich approximierter Multiplizierer dargestellt. Da es vergleichsweise wenige Veröffentlichungen für FPGA-basierte approximierte Multiplizierer gibt, werden auch einzelne Veröffentlichungen berücksichtigt, die ursprünglich auf Application-specific Integrated Circuits (ASICs) abzielen. Leider sind die verwendeten Metriken zur Beurteilung der Qualität sehr verschieden und teilweise nicht präzise definiert. Einige Arbeiten stehen als Open Source zur Verfügung und konnten dadurch mit der in Abschnitt 4.5 beschriebenen Evaluationsmethodik ausgewertet werden. Zudem wurden einzelne Arbeiten nachimplementiert, um einen Vergleich zu ermöglichen. Eine umfassende quantitative Evaluation des Standes der Wissenschaft und Technik und des in dieser Arbeit vorgestellten approximierten Multiplizierers wird in Unterabschnitt 4.6.1 vorgenommen. In diesem Kapitel

werden deshalb nur vereinzelt Zahlenwerte angegeben und primär die grundlegenden Ideen der verschiedenen Arbeiten dargelegt sowie eine qualitative Einordnung vorgenommen.

Mody et.al. schlagen einen Multiplizierer auf Basis neuartiger approximierter Kompressoren [4.1] vor [A 152]. Obwohl die Autor:innen explizit FPGAs als Zielplattform benennen, wird darauf abgestellt, Logikgatter einzusparen. Da FPGAs auf Lookup Tables (LUTs) basieren, ist ein solcher Ansatz in der Regel nicht geeignet, um das volle Potential der Plattform auszuschöpfen. Allerdings wird vorgeschlagen, den internen Carry-Pfad von Kompressoren zu vernachlässigen, was unabhängig von der Zielplattform und dem konkreten Vorgehen erhebliche Vorteile bietet und in dieser Arbeit aufgegriffen wird. Insbesondere wird das Design durch die Verkürzung des kritischen Pfades erheblich schneller. Allerdings führt das vorgeschlagene Design zu fehlerhaften Ausgangswerten bei 4 von 16 möglichen Eingangsvektoren. Unter Nutzung des Potentials der LUTs zu Abbildung beliebiger Logikfunktionen kann hier ein besseres Verhalten erreicht werden. Ein für LUT optimiertes Design wird in Unterabschnitt 4.4.1 diskutiert. Da in der Veröffentlichung keinerlei Metriken aufgeführt werden, die eine Beurteilung der Qualität der Approximation erlauben, wurde das Design anhand der Veröffentlichung präzise nachimplementiert. Wie bereits aufgrund der Eigenschaften des approximierten Kompressors vermutet, verursacht die Approximation einen erheblichen Fehler, der sich in einer NMED von 44 % widerspiegelt. Es ist zweifelhaft, ob dies für praktische Applikationen sinnvoll eingesetzt werden kann.

Ebenfalls auf approximierten Kompressoren beruht die Arbeit von Toan et al. [A 153]. Es werden diverse Kompressor-Designs vorgeschlagen und aus ihnen Multiplizierer konstruiert. Es wird ein beeindruckend niedriger Ressourcenbedarf erreicht. Die Approximation ist jedoch deutlich ungenauer als die in dieser Arbeit vorgeschlagene, was sich in einem um eine Größenordnung höheren MRE niederschlägt. Für Anwendungen, die ein größeres Maß an Ungenauigkeit tolerieren können, bieten sie ein sehr gutes Verhältnis von Genauigkeit und Ressourcenbedarf.

Einen grundsätzlich verschiedenen Ansatz verwenden Osta et al. [A 154]. Es wird eine approximierte Multiplikation durch geschicktes Runden auf im Format 2^n darstellbare Zahlen und somit durch wenige Shift-and-Add-Operationen erreicht. Zusätzlich wird die Addition durch approximierte Addierer realisiert. Ein präziser Ressourcenbedarf und Qualitäts-Metriken werden nicht angegeben. Auch sind die Quelldateien der Implementierung nicht verfügbar, was eine eigene Evaluation verhindert. Der WCRE beträgt jedoch mehr als 18 % und in mehr als drei Vierteln der Fälle beträgt der RE mehr als 1 %.

Einen nochmals gänzlich anderen Ansatz stellt der rekursive Aufbau von größeren Multiplizierern aus kleineren Basis-Multiplizierern dar. Ullah et al. [A 155] schlagen einen approximierten 4x2-Bit-Basis-Multiplizierer vor, der dediziert für moderne FPGA-Architekturen optimiert ist. Daraus werden rekursiv 4x4-, 8x8- und 16x16-Bit-Multiplizierer konstruiert. Die dabei notwendige Addition kann entweder exakt oder approximiert realisiert werden. Es werden zwei unterschiedlich genau approximierende Designs *Ca* und *Cc* vorgeschlagen, die sich in diesem Punkt unterscheiden.

Einen ähnlichen Ansatz verfolgen Guo et al. [A 156], wobei drei verschiedene Basis-Multiplizierer der Größe 4x4 Bit vorgeschlagen werden. Aus diesen werden unter Verwendung

[4.1] Anwendung und Aufbau von Kompressoren werden in Unterabschnitt 4.4.1 erläutert.

exakter oder approximierter Addition acht verschiedene 8x8-Bit-Multiplizierer konstruiert und evaluiert. Leider sind die Quellendateien nicht frei zugänglich. Da jedoch die Beschreibung der Designs im Paper sehr detailliert ist und die veröffentlichten Qualitäts-Metriken vielversprechend sind, wurden alle acht Designs vom Autor dieser Arbeit zu Vergleichszwecken nachimplementiert. Die auf dem Prinzip der rekusiven Konstruktion basierenden Multiplizierer-Designs erreichen eine hohe Ressourcenersparnis und eine sehr gute Qualität der Approximation. Die Designs *[A 156] T1*, *[A 156] T2* und *[A 155] Ca* sind dabei die engsten Konkurrenten des in dieser Arbeit vorgeschlagenen Multiplizierer-Designs (siehe dazu Unterabschnitt 4.6.1).

Ein äußerst interessanter Ansatz ist die automatisierte Erzeugung von approximierten Multiplizierern mittels „Multi-objective Cartesian Genetic Programming“ von Mrazek et al. [A 151]. Die Ergebnisse sind von beeindruckender Qualität und die Quellen stehen unter Open-Source-Lizenz zur Verfügung. Dieser Ansatz zielt jedoch auf ASICs ab. Es wird also für ein Mapping auf Logikgatter und nicht auf LUTs optimiert. In [A 155] zeigen die Autor:innen, dass dedizierte FPGA-Ansätze, die gezielt für die auf FPGAs vorhandenen Logikressourcen entworfen wurden, wesentlich bessere Ergebnisse liefern. Allerdings erscheint es durchaus vielversprechend, das zu Grunde liegende Prinzip von [A 151] so anzupassen, dass eine Optimierung für FPGA-Logikressourcen durchgeführt wird. In [A 148] schlagen die Autor:innen vor, mit Hilfe von ML eine für FPGAs geeignete Auswahl aus der von Mrazek et al. [A 151] publizierten Bibliothek von ASIC-Designs vorzunehmen. Die so ausgewählten Designs wurden auf einer FPGA-Toolchain evaluiert. Die Ergebnisse sind von hoher Qualität und viele der Designs erreichen Pareto-optimale Ergebnisse in unserer Analyse in Unterabschnitt 4.6.1.

4.4 Approximierte Multiplizierer zur optimalen Nutzung von FPGA-Logikressourcen

Approximate Computing ist ein Ansatz zur Senkung des Ressourcenbedarfs durch Inkaufnahme einer gewissen Ungenauigkeit der Ergebnisse. Typischerweise wird diese erreicht, indem die logischen Funktionen so abgeändert werden, dass sie sich auf weniger Logikgatter abbilden lassen. Dabei werden abweichende Ausgangswerte für einige wenige Eingangsvektoren toleriert. Da FPGAs logische Funktionen jedoch nicht direkt durch Complementary Metal–Oxide–Semiconductor (CMOS)-Logikgatter realisieren, sondern sie durch Static Random-Access Memory (SRAM)-basierte LUTs abbilden, ist dieser Ansatz für FPGAs im Allgemeinen nicht optimal. Vielmehr muss eine abweichende logische Funktion gesucht werden, welche sich optimal auf die vorhanden Ressourcen des FPGA abbilden lässt. Im Allgemeinen handelt es sich hierbei um LUTs, aber auch um die in modernen FPGAs üblichen Carry Chains. In modernen FPGAs werden LUTs verwendet, die aus sechs Eingangswerten einen Ausgangswert bilden (6:1-LUT). Typischerweise kann eine solche 6:1-LUT alternativ in zwei 5:1-LUTs umkonfiguriert werden, wobei die fünf Eingangssignale identisch sein müssen [A 157]. Im Gegensatz zu Logikgatter-basierten Technologien wie beispielsweise ASICs ist der Ressourcenbedarf dabei vollkommen unabhängig von der konkreten logischen Funktion, die eine LUT realisiert. Approximate-Computing-Designs für FPGA müssen dies unbedingt berücksichtigen, um optimale Ergebnisse zu erzielen.

Multiplizierer sind ein wichtiger Grundbaustein vieler FPGA-Anwendungen. Insbesondere basieren Anwendungen der Signalverarbeitung und Hardwareimplementierungen von DNNs ganz

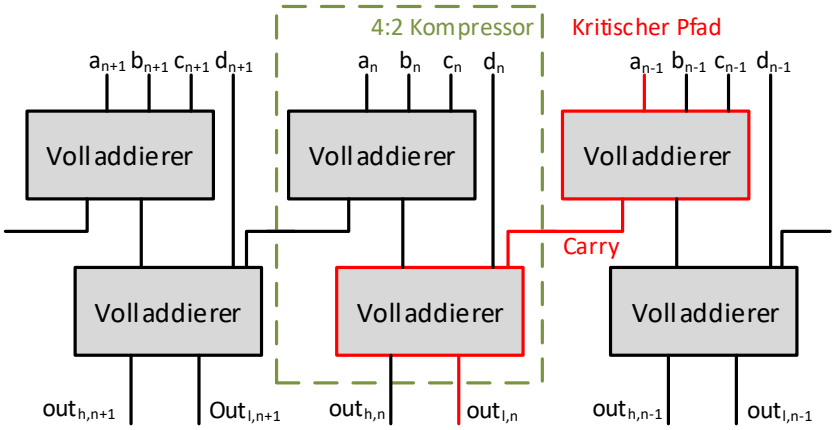


Abbildung 4.1: Kritischer Pfad eines 4:2-Kompressors (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE)

wesentlich auf einer großen Zahl von Multiplizierern. Im Folgenden wird ein im Rahmen dieser Arbeit entwickelter Ansatz für approximierte Unsigned-Multiplizierer beschrieben. Im Gegensatz zu einigen wichtigen Vergleichsarbeiten wie [A 155] und [A 156] wird in dieser Arbeit der Multiplizierer konventionell aus Partialproduktbildung und Kompressionsbaum aufgebaut. Im Folgenden werden zunächst die verwendeten Kompressoren beschrieben, da sich die verwendete Partialproduktbildung aus der Entscheidung für einen bestimmten approximierten Kompressor ergab.

4.4.1 Approximierte Kompressoren

Einer der wichtigsten Grundbausteine für arithmetische Funktionseinheiten sind Kompressoren. Als solche werden Logikschaltungen bezeichnet, die eine im Zählcode (also im Unärsystem) vorliegende Zahl in eine Zahl im (binären) Stellenwertsystem umsetzten. Da hierfür weniger oder maximal gleich viele Stellen benötigt werden, ist die Anzahl der Eingänge typischerweise größer als die der Ausgänge. Die Anzahl der Ein- und Ausgänge wird im Format *Anzahl der Eingänge* : *Anzahl der Ausgänge* beschrieben [A 145]. Kompressoren sind beispielsweise Grundbaustein für die Reduktionsbäume in Wallace- und Dadda-Multiplizierern. In der digitalen Signalverarbeitung werden sie darüber hinaus in der Akkumulatorstufe für Multiply-Accumulate (MAC)-Operationen verwendet. Typischerweise werden 3:2- oder 4:2-Kompressoren genutzt. Designs, die auf letzteren basieren, haben tendenziell eine regelmäßigere Struktur [A 145]. Dies ist ein entscheidender Vorteil, da eine solche Struktur zu einer Verminderung von Routing Congestions führt und tendenziell ein geringeres Delay verursacht. Deshalb stehen 4:2-Kompressoren im Fokus dieser Arbeit.

In Implementierungen für ASICs werden 4:2-Kompressoren typischerweise aus zwei Volladdierern aufgebaut. Dabei verläuft der kritische Pfad durch beide Volladdierer. Abbildung 4.1 veranschaulicht, dass der kritische Pfad dabei durch das interne Carry-Signal verläuft. In einer 5:1-LUT-basierten Implementierung dominiert der Carry-Pfad das Delay. Demzufolge führen Approximationen, die auf das Carry-Signal verzichten, zu einem überlegenen Verhalten bezüg-

Tabelle 4.1: Wahrheitstabelle des approximierten 4:2-Kompressors [B 4]

a	b	c	d	Out_h	Out_l	Relative Abweichung
0	0	0	0	0	0	0 %
0	0	0	1	0	1	0 %
0	0	1	0	0	1	0 %
0	0	1	1	1	0	0 %
0	1	0	0	0	1	0 %
0	1	0	1	1	0	0 %
0	1	1	0	1	0	0 %
0	1	1	1	1	1	0 %
1	0	0	0	0	1	0 %
1	0	0	1	1	0	0 %
1	0	1	0	1	0	0 %
1	0	1	1	1	1	0 %
1	1	0	0	1	0	0 %
1	1	0	1	1	1	0 %
1	1	1	0	1	1	0 %
1	1	1	1	1	1	25 %

lich des Delays. Es wurden verschiedene solcher Approximationen vorgeschlagen: unter anderem [A 152] für FPGAs und [A 158] für ASICs. Im weiteren Verlauf wird gezeigt, dass eine der von Yang et al. [A 158] für ASICs vorgeschlagene Approximationen für FPGAs besonders gut geeignet ist.

Die vier Eingänge eines 4:2-Kompressors sind alle gleichwertig und repräsentieren jeweils den Zahlenwert 'Eins'. Damit lassen sich im Sinne einer redundanten Zahlendarstellung die Zahlen im Intervall $[0;4]$ darstellen. Demzufolge muss ein exakt arbeitender 4:2-Kompressor mit seinen Ausgangssignalen ebendiesen Zahlenbereich darstellen können. Da die beiden Ausgänge die Stellen-Wertigkeit 2^0 und 2^1 aufweisen und somit nur Zahlen von null bis drei darstellbar sind, wird, wie bereits beschrieben und in Abbildung 4.1 veranschaulicht, ein internes Carry mit der Wertigkeit 2^1 verwendet. Wird dieses Carry-Signal in der Approximation aus den genannten Gründen nicht realisiert, entsteht ein Fehler für den Fall, dass alle Eingänge den Wert `'1'` annehmen. Alle anderen Eingangsvektoren können bei günstiger Wahl der logischen Funktion korrekt auf die Ausgänge abgebildet werden. Die gewählte logische Funktion kann aus Tabelle 4.1 entnommen werden. Die Abweichung im nicht korrekt abgebildeten Fall des Eingangsvektors `"1111"` beträgt lediglich 'Eins'. Dies ist die kleinstmögliche Abweichung. Da sie darüber hinaus im Falle des größtmöglichen korrekten Ausgangswertes auftritt, ist die relative Abweichung vergleichsweise klein. Diese relative Abweichung ist jedoch, wie in Abschnitt 4.2 erläutert, entscheidend für die Qualität der Approximation.

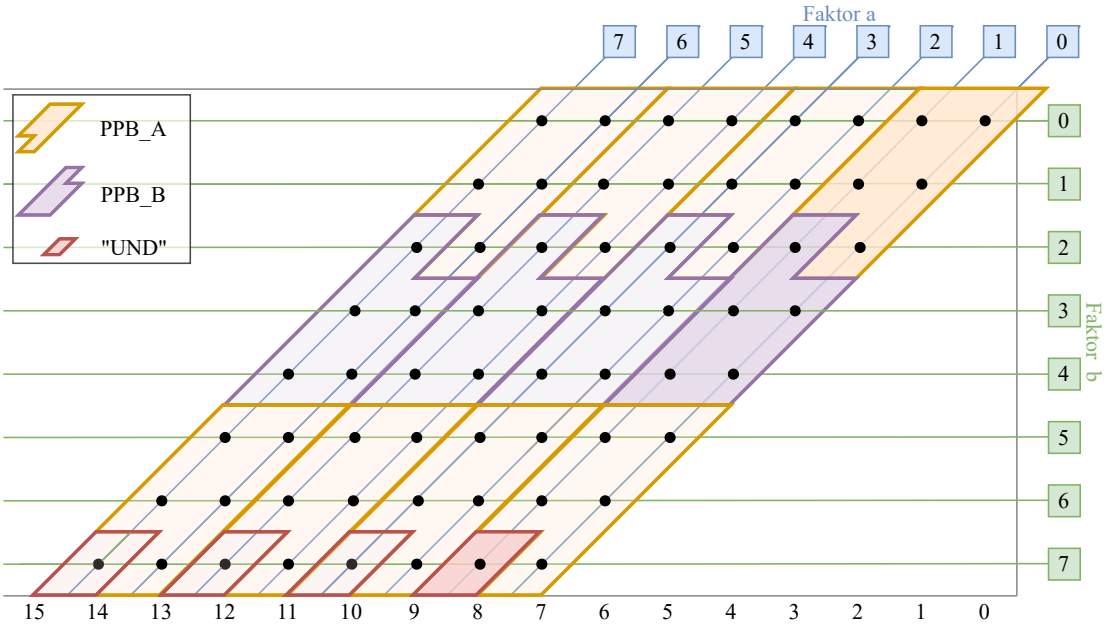


Abbildung 4.2: Anordnung der Partialprodukt-Blöcke (PPBs) für einen 8x8-Bit-Multiplizierer (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE)

4.4.2 Für FPGA-Logikressourcen optimierte Partialproduktbildung

Es gibt verschiedene Möglichkeiten, die Partialprodukte für eine Multiplikation zu erzeugen. Allerdings kann nicht einfach ein Ansatz zur Partialprodukterzeugung aus der Literatur für exakte Multiplizierer übernommen werden, da die Art der Partialprodukterzeugung sich durch die approximierten Kompressoren auf die Qualität der Approximation auswirkt. So wurden im Rahmen dieser Arbeit beispielsweise verschiedene Varianten von Modified-Booth-Encoding getestet, die jedoch aufgrund der Signed-Digit-Darstellung zu einer erheblich schlechteren Qualität der Approximation des Multiplizierers führten. In der Literatur zu approximierten Multiplizierern auf Basis von approximierten Kompressoren werden, soweit dem Autor bekannt, `AND`-Verknüpfungen zweier Eingänge zur Partialproduktbildung genutzt. Bei diesem „klassischen“ Weg der Partialproduktbildung werden diese durch die Verknüpfung

$$PP_{i,j} = A_i \wedge B_j \tag{25}$$

gebildet, wobei $PP_{i,j}$ die Bitstelle j des i-ten Partialproduktes darstellt. A_i und B_i sind das jeweils i-te Bit der Faktoren A und B. Auf einem ASIC kann diese Operation als `AND`-Gatter und „fest verdrahtete“ Shift-Operation sehr effizient abgebildet werden. Auf einem FPGA dagegen wird diese Funktion auf 2:1-LUTs abgebildet, wobei eine 6:1-LUT in zwei 2:1-LUTs mit unabhängigen Eingängen konfiguriert werden kann. Die naheliegende Annahme, dass dies die Fähigkeit von LUTs, beliebig komplexe logische Funktionen abzubilden, nicht optimal nutzt, ist Ausgangspunkt für die Konzeption der Partialprodukterzeugung in dieser Arbeit.

Die Fähigkeit von LUTs, komplexe logische Funktionen abzubilden, wird genutzt, um die eigentliche Partialproduktbildung mit einer teilweisen Kompression der Partialprodukte zu kom-

Tabelle 4.2: Anzahl der PPB_A Ausgangsbits im Vergleich mit einer 'konventionellen' Partialproduktbildung basierend auf AND-Gattern für dieselben Multiplizierer-Eingänge [B 4]

Zahlenwert der Bitstelle	Anzahl 'konventioneller' Partialprodukt-Bits	Anzahl der PPB_A Ausgangsbits
2^0	1	1
2^1	2	1
2^2	2	1
2^3	0	1
Darstellbarer Wertebereich	[0,13]	[0,15]

Tabelle 4.3: Anzahl der PPB_B Ausgangsbits im Vergleich mit einer 'konventionellen' Partialproduktbildung basierend auf AND-Gattern für dieselben Multiplizierer-Eingänge [B 4]

Zahlenwert der Bitstelle	Anzahl 'konventioneller' Partialprodukt-Bits	Anzahl der PPB_B Ausgangsbits
2^0	2	1
2^1	2	1
2^2	1	1
2^3	0	1
Darstellbarer Wertebereich	[0,10]	[0,15]

binieren. Die dadurch entstehende komplexere logische Funktion wird so gestaltet, dass sie sich günstig auf LUTs abbilden lässt. Der entstehende funktionelle Block wird als Partialprodukt Block (PPB) bezeichnet. Ein solcher PPB darf von maximal sechs Eingangssignalen abhängig sein, um auf eine einzelne LUT abbildbar zu sein. Allerdings scheint es günstiger, PPBs mit lediglich fünf Eingängen zu nutzen, da in diesem Fall die doppelte Anzahl Ausgänge pro LUT erzeugt werden kann. Insbesondere wenn es gelingt, eine gerade Anzahl an Ausgangsbits pro PPB zu nutzen, ergibt sich so eine vorteilhafte Ressourcennutzung. Der Grund dafür ist, dass sich jede 6:1-LUT in zwei 5:1-LUTs konfigurieren lässt, die allerdings zwingend die gleichen Eingangssignale verarbeiten müssen. Bei einer ungeraden Anzahl von Ausgangsbits würde also eine 5:1 LUT ungenutzt bleiben, da eine sinnvolle Verwendung aufgrund der festgelegten Eingangssignale kaum vorstellbar ist.

Um diese beiden Anforderungen zu erfüllen, werden zwei verschiedene Typen von PPBs kombiniert, die im Folgenden PPB_A und PPB_B genannt werden. In Abbildung 4.2 wird die Anordnung beider PPB-Typen veranschaulicht. Wie gefordert hat jeder PPB-Block fünf Eingänge. So hat beispielsweise der hervorgehobene PPB_A die Eingänge a_0, a_1, b_0, b_1 und b_2. Der ebenfalls hervorgehobene PPB_B hat die Eingänge a_0, a_1, b_2, b_3 und b_4, wobei a_i und b_i jeweils das $i-te$ Bit der jeweiligen Faktoren a und b der Multiplikation sind. Jeder PPB deckt dabei fünf 'konventionelle' Partialproduktbits ab. Dabei sind mit 'konventionellen' Partialproduktbits jene gemeint, die mit Formel 25 entstehen würden. Sie werden jedoch nicht wirklich erzeugt, sondern dienen nur als Hilfsmittel zum Verständnis der Partialproduktbildung mittels PPBs. Die

Tabelle 4.4: Ressourcenbedarf zur Erzeugung der Partialprodukte eines 8x8-Bit-Multiplizierers [B 4]

	„Konventionell“ (AND-Gatter)	Vorgeschlagenes PPB-Design
Anzahl der AND-Gatter	64	4
LUTs für AND-Gatter	32	2
PPBs	0	12
LUTs für PPBs	0	24
Gesamt-Ressourcenbedarf (LUTs)	32	26
Anzahl von Bits zur Darstellung der Partialprodukte	64	52

’konventionellen’ Partialproduktbits innerhalb eines PPB repräsentieren verschiedene Zahlenwerte, wobei einige Zahlenwerte mehrfach auftreten. In ihrer Gesamtheit können die von einem PPB repräsentierten ’konventionellen’ Partialproduktbits als eine redundante Zahlendarstellung aufgefasst werden. Diese werden von den PPBs komprimiert und als vier Ausgangsbits in nichtredundanter Zahlendarstellung ausgegeben. Die jeweilige Anzahl der Ein-und Ausgangsbits mit ihren jeweiligen Wertigkeiten ist in den Tabellen 4.2 und 4.3 dargestellt.

In jedem PPB werden nur fünf „konventionelle“ Partialprodukte verarbeitet, obwohl sich aus den Eingangssignalen jedes PPB sechs ableiten lassen würden. Allerdings wäre in diesem Fall zur Darstellung des Ausgangs-Wertebereichs ein zusätzliches Ausgangsbit notwendig. Dies würde zum einen den Ressourcenbedarf um gleich zwei 5:1-LUTs vergrößern, da eine ungerade Zahl Ausgangsbits entstünde. Zum anderen würde sich die erreichte Kompression verringern, was sich in einem größeren nachfolgenden Kompressionsbaum niederschlagen würde. Somit belegt jeder PPB zwei 6:1-LUTs, die jeweils als zwei 5:1-LUTs konfiguriert werden.

Eine wichtige Eigenschaft einer Methodik zur Partialprodukterzeugung ist das asymptotische Delay. Eine ungünstige asymptotische Laufzeit ist für die Konstruktion von Multiplizierern größerer Bitbreite ungeeignet. Da die Anordnung der PPBs einer regelmäßigen Struktur folgt, können solche größere Multiplizierer durch Fortführen des in Abbildung 4.2 dargestellten Musters erzeugt werden. Da keine Verbindung zwischen den einzelnen PPBs besteht, kann das asymptotische Zeitverhalten mit $O(1)$ beschrieben werden. Somit ist dieser Ansatz auch für die Konstruktion von Multiplizierern mit großer Bitbreite geeignet.

Wie Abbildung 4.2 zeigt, müssen je nach Größe des Multiplizierers einige wenige Partialprodukt-Bits außerhalb der PPBs erzeugt werden. Diese einzelnen Partialproduktbits werden auf ’konventionelle’ Art erzeugt: durch eine als AND-Gatter konfigurierte LUT. Moderne 6:1-LUTs lassen sich auch in zwei 2:1-LUTs mit unabhängigen Eingängen konfigurieren [A 157]. Somit können zwei solche AND-Gatter durch eine LUT dargestellt werden. Im Falle eines 8-Bit-Multiplizierers werden folglich lediglich zwei zusätzliche LUTs benötigt. Somit bleibt der Mehraufwand durch die Erzeugung dieser einzelnen Partialproduktbits gering.

In Tabelle 4.4 ist der Ressourcenbedarf dieses Ansatzes im Vergleich zum „konventionellen“ AND-Gatter-basierten Ansatz dargestellt. Die hier vorgeschlagene Methode benötigt 19 % weni-

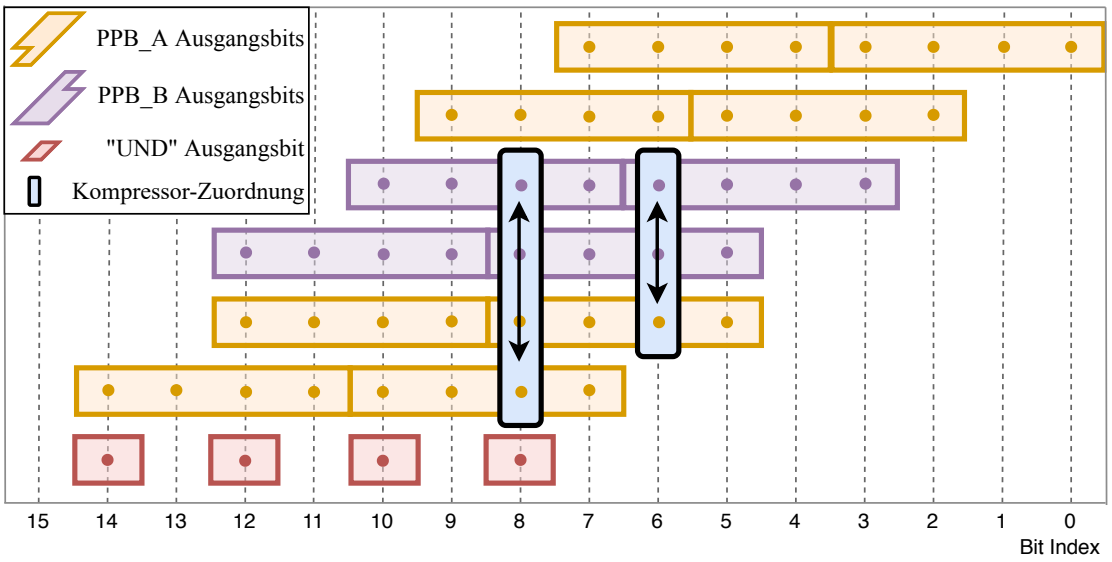


Abbildung 4.3: Zuordnung der PPB-Ausgänge zur ersten Kompressionsstufe (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE)

ger Logikressourcen. Darüber hinaus benötigt sie nur 81 % der Partialproduktbits eines „konventionellen“ `AND`-Gatter-Ansatzes zur Darstellung der Partialprodukte, wodurch der nachfolgende Kompressionsbaum entsprechend kleiner, also ressourcensparender ausfallen kann.

4.4.3 Statistische Optimierung des Kompressionsbaumes

Die in Unterabschnitt 4.4.1 beschriebenen approximierten Kompressoren liefern für 15 von 16 möglichen Eingangsvektoren das korrekte Ergebnis. Da lediglich der Fall, in dem alle vier Eingänge den Wert `'1'` annehmen, ein fehlerhaftes Ergebnis hervorruft, sollte dieser Fall möglichst vermieden werden. Anders ausgedrückt: die Wahrscheinlichkeit seines Auftretens sollte minimiert werden.

In Abbildung 4.3 wird illustriert, dass eine gewisse Wahlfreiheit bei der Zuordnung der einzelnen PPB-Ausgänge zu den Kompressor-Eingängen besteht. Die Ausgänge der PPBs nehmen mit unterschiedlicher Wahrscheinlichkeit den Wert `'1'` an. Demzufolge ist es möglich, durch geschickte Zuordnung der PPB-Ausgänge zu den Kompressor-Eingängen die Wahrscheinlichkeit zu minimieren, dass alle vier Eingänge eines Kompressors gleichzeitig den Wert `'1'` annehmen. Die genaue Statistik der Ausgangs-Wahrscheinlichkeiten der PPBs ist neben der internen Funktionsweise der PPBs auch von der statistischen Verteilung der Multiplizierer-Eingangswerte abhängig. Allerdings unterscheidet sich ebendiese Verteilung je nach Anwendungsdomäne und sogar je nach konkret zu verarbeitendem Datensatz. Eine Berücksichtigung dieser statistischen Eigenschaften ist demzufolge nur für einen speziellen Fall möglich. Die dafür erforderliche Simulation erfordert allerdings eine bezüglich der Bitbreite des Multiplizierers exponentielle Simulationszeit. Um dennoch Aussagen über das statistische Verhalten der PPB-Ausgänge treffen zu können, wurde dieses durch eine erschöpfende Simulation aller möglichen Eingangsvektoren eines PPB bestimmt. Die Ergebnisse sind in den Tabellen 4.5 und 4.6 dargestellt. Auf

Tabelle 4.5: PPB_A Statistik [B 4]

Ausgangs-Bit O_i	$P(O_i = 1)$
0	25,0 %
1	37,5 %
2	37,5 %
3	9,375 %

Tabelle 4.6: PPB_B Statistik [B 4]

Ausgangs-Bit O_i	$P(O_i = 1)$
0	37,5 %
1	37,5 %
2	21,875 %
3	6,25 %

Grundlage dieser Statistik wurde die Optimierung der Zuordnung der PPB-Ausgänge zu den Kompressor-Eingängen vorgenommen. Die Qualität der Approximation konnte damit deutlich verbessert werden.

4.5 Evaluationsmethodik

Approximierte Multiplizierer lassen sich prinzipiell sehr gut vergleichen. Dazu stehen eine Vielzahl von aussagekräftigen Kennziffern bereit. Dies sind zum einen Gütemaße zur Bewertung der Approximation, wie sie in Abschnitt 4.2 diskutiert werden. Zum anderen muss jedoch auch die Ressourceneinsparung der Approximation ermittelt werden. Im hier relevanten Fall von approximierten Designs für eine FPGA-Plattform sind dies:

- die benötigten Logikressourcen,
- das Delay des kritischen Pfades
- und die Leistungsaufnahme.

Diese Werte sind jedoch – im Gegensatz zu den Gütemaßen für die Qualität der Approximation – abhängig von der konkreten Zielplattform, der verwendeten Electronic Design Automation (EDA)-Toolchain, sowie der Parametrisierung selbiger. Darüber hinaus stellte sich heraus, dass sich die einer quantitativen Auswertung zugrundeliegenden Definitionen der Gütemaße und Ressourcenbedarfe von Publikation zu Publikation unterscheiden. Zudem werden diese Definitionen häufig nicht oder nicht präzise angegeben. Um einen aussagekräftigen quantitativen Vergleich zu ermöglichen, wurde im Rahmen dieser Arbeit eine umfassende Evaluationsmethodik erarbeitet. Diese wird nachfolgend beschrieben, wobei die getroffenen Entscheidungen hergeleitet und begründet werden.

Die gesamte Evaluation sowohl bezüglich der Gütemaße der Approximation als auch bezüglich des Ressourcenbedarfs wird automatisiert durchgeführt, wobei als Input lediglich eine Hardwarebeschreibung im Format Very High Speed Integrated Circuit Hardware Description Language (VHDL) oder Verilog benötigt wird. Für Synthese, Implementierung und Schätzung der Leistungsaufnahme wurde *Xilinx Vivado* [A 159] verwendet. Es wird gezielt der Non-Project Mode genutzt, in dem die Tool-Parametrisierung ausschließlich von den generierten Skripten determiniert wird, um die Reproduzierbarkeit der Ergebnisse zu verbessern. Die benötigten Simulationen wurden mit Siemens/Mentor Graphics 'Questa Advanced Simulator' [A 160], nachfolgend *QuestaSim* genannt, durchgeführt. Der Ablauf dieser Evaluation wird in Abbildung 4.4 dargestellt.

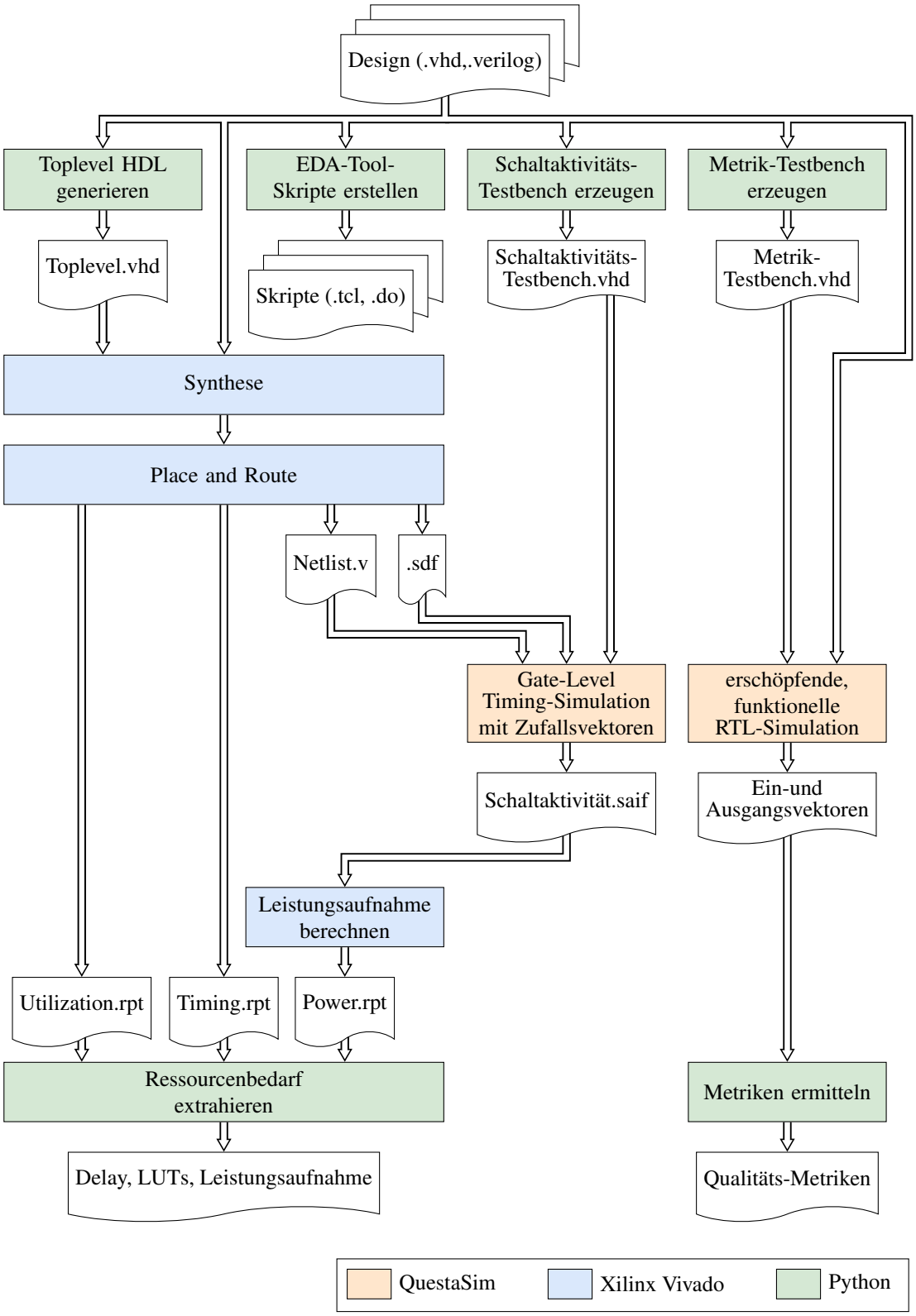


Abbildung 4.4: Ablauf der automatisierten Evaluation von approximierten Multiplizierern

4.5.1 Gütemaße der Approximation

Zur Bestimmung der in Abschnitt 4.2 diskutierten Gütemaße werden die Designs zunächst funktional auf Register Transfer Level (RTL) simuliert. Aus den Eingangs- und Ausgangsvektoren dieser Simulation werden dann die Gütemaße berechnet. Die relativ geringe Bitbreite der Eingangsvektoren von zwei mal acht Bit ermöglicht es, alle Eingangsvektoren erschöpfend zu simulieren. Dadurch können auch die Worst-Case-Metriken exakt bestimmt werden.

4.5.2 Definition des Ressourcenbedarfs

Ein wichtiges Optimierungsziel für approximierte Designs auf FPGAs ist die Reduktion der benötigten FPGA-Ressourcen. Dies ist auch im Sinne der Energieeffizienz wichtig, da der Bedarf an Logikressourcen des Designs (häufig) die Größe des benötigten FPGAs bestimmt. Die statische Verlustleistung wiederum wird primär von der Größe des FPGAs bestimmt [A 161]. Als Metrik für den Ressourcenbedarf wird meist die Anzahl der benötigten LUTs genutzt. Vereinzelt werden jedoch auch die instantiierten Carry-Chains berücksichtigt und als Metrik die Summe aus LUTs und Carry-Chains genutzt. Die Carry-Chains sind jedoch ohnehin vorhanden und jeweils fest in einen einzelnen Logik-Slice integriert [A 157]. Eine Verwendung ohne die dazugehörigen LUTs des Logik-Slices ist kaum vorstellbar und dem Autor dieser Arbeit nicht bekannt. Somit spielt es für den Ressourcenbadarf keine Rolle, ob nur die LUTs eines Logik-Slices genutzt werden und die Carry-Chain ungenutzt bleibt, aber auch nicht für andere Zwecke nutzbar ist, oder ob diese tatsächlich genutzt wird. In dieser Arbeit wird deshalb ausschließlich auf die Anzahl der belegten LUTs abgestellt.

4.5.3 Definition des Delays

Neben dem Einfluss verschiedener Zielplattformen sowie der verwendeten Toolchain und deren Parametrisierung hat auch die genaue Definition des Delays einen entscheidenden Einfluss auf die quantitativen Angaben bezüglich dieser Zielgröße. Durch die Auswertung von Designs aus dem Stand der Technik, die als Open Source verfügbar sind, wurde deutlich, dass die Definition des Delays das dominierende Problem bei der Vergleichbarkeit mit Literaturangaben ist. So wurde beispielsweise für das Design *[A 148] mul8u_7g9* bei identischer Zielplattform und Toolchain je nach Definition ein Delay von 4.667 ns oder 9.858 ns ermittelt. Diese Abweichung um mehr als Faktor zwei macht eine genauere Definition des Delays offensichtlich erforderlich.

Im Rahmen dieser Arbeit wird das Delay eines Multiplizierers definiert als das Delay des kritischen Pfades zwischen zwei Registern an Eingang beziehungsweise Ausgang des Multiplizierer-Designs. Dies wird auch in Abbildung 4.5 verdeutlicht. Die grün eingefärbten Elemente werden bei dieser Definition des Delays berücksichtigt. Die verschiedenen Multiplizierer werden dazu jeweils als `component` in einem Toplevel-Design instantiiert, welches Eingangs- und Ausgangsregister definiert. Dieses Design wird dann für eine festgelegte Taktfrequenz synthetisiert und implementiert. Für alle hier untersuchten Designs wurde eine Zielfrequenz von $125\,MHz$ gewählt. Das entspricht einer Taktperiode von $T_{clk} = 8\,ns$. Das Delay wird dann mittels Post-Place and Route (P&R)-Static Timing Analysis (STA) durch Abzug des Worst Negative Slack (WNS) von der Taktperiode bestimmt.

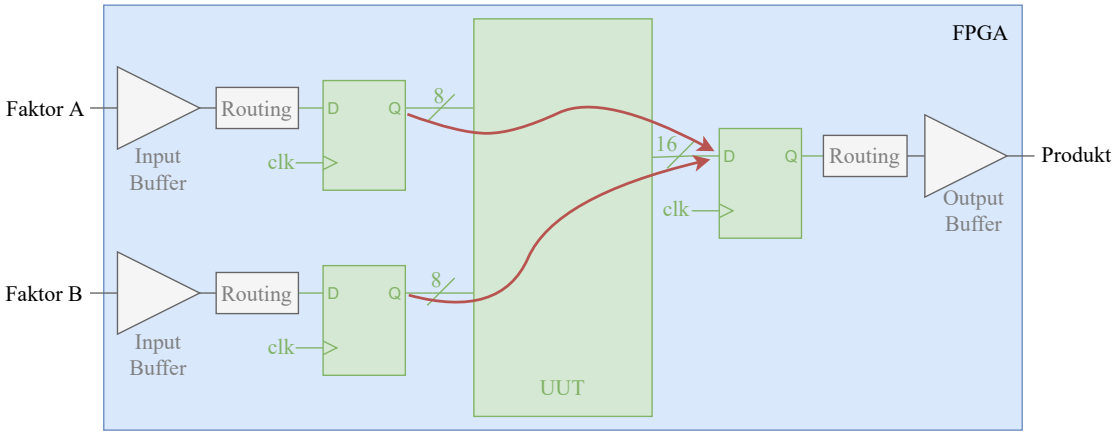


Abbildung 4.5: Delay bei Vivado-Timing-Analyse mit kombinatorischer Logik als Pipeline-Stufe zwischen Registern

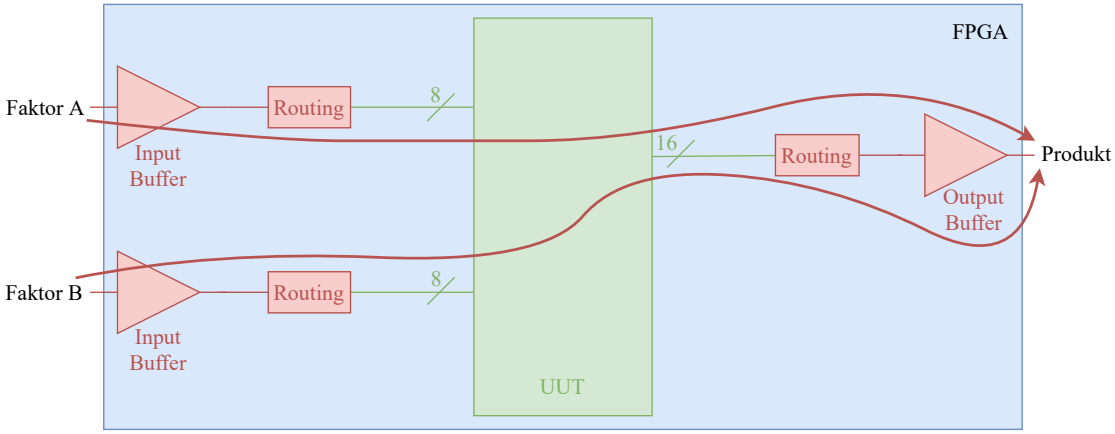


Abbildung 4.6: Input- und Output-Buffer dominieren das Ergebnis der Vivado-Timing-Analyse, wenn eine rein kombinatorische Logik als Toplevel-Design implementiert wird.

Im Vergleich dazu wird in der Literatur offensichtlich häufig direkt der Multiplizierer als Toplevel-Design synthetisiert und es werden die Ergebnisse der Timing-Analyse in der Default-Einstellung genutzt. In diesem Fall implementiert Vivado jedoch eine rein kombinatorische Schaltung auf dem FPGA. Wie in Abbildung 4.6 illustriert, ist das Timing in diesem Fall von Eingangs- zu Ausgangspin definiert. Es beinhaltet also auch das Delay der Eingangs- und Ausgangsbuffer, sowie eines eventuell sehr langen Routings zwischen den Pins und dem eigentlichen Design. Insbesondere die Ausgangsbuffer sind dabei kritisch. So beträgt allein das Delay der Ausgangsbuffer des Virtex-7 FPGA 2.321 ns und ist somit größer als das Delay mancher approximierter Multiplizierer.

Die hier verwendete Definition des Delays wurde aus folgenden Gründen gewählt:

- Die Nutzung eines FPGA als rein kombinatorische Schaltung ist nicht üblich.

- Typischerweise wird die Multiplikation, eventuell als MAC-Operation um eine Addition erweitert, als eigene Pipeline-Stufe implementiert. Diese bestimmt zudem häufig die mögliche Taktrate des Gesamtsystems. Somit ist das Timing-Verhalten eines approximierten Multiplizierers als Pipeline-Stufe zwischen Registern von besonderem Interesse.
- Auch für Anwendungen, in denen ein Multiplizierer mit anderen kombinatorischen Elementen innerhalb der selben Pipeline-Stufe kombiniert wird, liefert die hier verwendete Definition des Delays eine realistischere Bewertung als eine, die Ein- und Ausgangsbuffer sowie das zugehörige Routing enthält.

Es sei darauf hingewiesen, dass die in diesem Kapitel im Zusammenhang mit der Evaluation von Approximate-Computing-Designs angewandte Definition des Delays von sonst üblichen Konventionen abweicht. So handelt es sich explizit nicht um die Zeitdauer zwischen dem Überschreiten des $0,5 \cdot V_{DD}$ Signalpegels an Ein- und Ausgang einer kombinatorischen Schaltung. Vielmehr wird, wie vorhergehend beschrieben, das Delay eines approximierten Designs aus dem Ergebnis der STA eines synchronen Designs abgeleitet. Dadurch gehen beispielsweise die Setup-Time der Register und der Clock-Skew mit in das Delay ein. Folglich kann das Delay in diesem Sinne interpretiert werden als die minimale Taktperiode, bei der das Design in einer Pipeline-Stufe betrieben werden kann. Für den Vergleich verschiedener Designs beziehungsweise die Auswahl eines Designs für eine konkrete Anwendung ist diese Definition nach Einschätzung des Autors dieser Arbeit jedoch zielführend.

4.5.4 Ermittlung der Leistungsaufnahme

Um eine möglichst präzise Analyse der Leistungsaufnahme zu ermöglichen, werden die Schaltaktivitäten per Simulation ermittelt. Hierbei ist es von entscheidender Bedeutung, eine für den Anwendungsfall möglichst repräsentative Abfolge von Eingangsvektoren zu erzeugen [A 162]. Die erschöpfende Simulation aller Vektoren, wie sie zur Bestimmung der Gütemaße angewandt wird, wird systematisch durch zwei verschachtelte For-Schleifen erzeugt. Dadurch entsteht aus Sicht des simulierten Designs eine sehr spezielle Abfolge von Eingangsvektor-Übergängen, welche offensichtlich stark korreliert und nicht repräsentativ für andere Anwendungen ist. Die genauesten Schaltaktivitätswerte bezüglich einer konkreten Anwendung liefern natürlich repräsentative Eingangsvektoren aus genau dieser Anwendung. Um allgemeingültige Aussagen treffen zu können, werden im Rahmen dieser Arbeit pseudozufällige Eingangsvektoren verwendet, die einem gleichverteilten weißen Rauschen entsprechen. Die Simulation wird als Timing-Simulation der „back-annotated“ Netzliste des Post-P&R-Designs durchgeführt und die Ergebnisse im Switching Activity Interchange Format (SAIF) von Vivado ausgewertet. Um eine gute Vergleichbarkeit der Leistungsaufnahme der verschiedenen Designs zu ermöglichen, wurde die Simulation für alle Designs mit der gleichen Taktfrequenz von 125 MHz durchgeführt. Die konkrete Taktfrequenz wurde gewählt, da alle betrachteten Designs bei ihr erfolgreich implementiert werden können und dabei einen gewissen Timing Slack aufweisen. Wäre die gewählte Frequenz für einige Designs nur gerade so erreichbar, könnten die Timing-Closure-Strategien des EDA-Tools zu einem zusätzlichen Bedarf an Logikressourcen und somit zu einer Verzerrung der Ergebnisse führen.

Die Vivado-Power-Analyse enthält die gesamte Leistungsaufnahme des FPGA. Das beinhaltet insbesondere auch die Leistungsaufnahme der Input und Output Buffer sowie die gesamte sta-

tische Verlustleistung des im Vergleich zu einem einzelnen Multiplizierer sehr großen FPGAs. Analog zur Timing-Analyse beeinflussen diese Faktoren auch bei der Analyse der Leistungsaufnahme signifikant das Gesamtergebnis. Deshalb wird auf die Leistungsaufnahme des eigentlichen Multiplizierers als Intellectual Property Core (IP Core) abgestellt.

Die naheliegende Lösung wäre, aus dem hierarchischen Report die Werte für das Multiplizierer-Untermodul zu extrahieren. Allerdings ist nach Erfahrung des Autors die Zuordnung von Ressourcen zu den Hierarchieebenen in Vivado teilweise nicht präzise. Der Grund dafür dürfte in der Hierarchieebenen übergreifenden Optimierung liegen. Stattdessen wird der nach Hardware-Primitiven aufgeschlüsselte Bericht genutzt und hier die Leistungsaufnahme der Komponenten, aus denen sich die Multiplizierer zusammensetzen, extrahiert. Konkret sind das die Leistungsaufnahme durch Logikressourcen sowie die durch das Routing der Signale. Diese werden aufsummiert und als Verlustleistung des Multiplizierers interpretiert.

Man beachte, dass diese Werte lediglich die dynamische Verlustleistung repräsentieren. Ein aussagekräftiger Vergleich bezüglich der statischen Verlustleistung lässt sich nur schwer realisieren. So ordnet auch Xilinx den Designs lediglich ihre dynamische Verlustleistung zu [A 162]. Der Grund dafür ist, dass die statische Verlustleistung eines FPGA-Designs hauptsächlich durch die Größe des konkreten FPGA bestimmt wird [A 161]. Zwar ermöglichen kleinere Designs potentiell auch die Verwendung kleinerer FPGAs und somit eine deutliche Reduktion der statischen Verlustleistung. Allerdings ist die Zuordnung von Verlustleistung zu einem Ressourcenbedarf stark nichtlinear und zudem von vielen weiteren Einflussgrößen abhängig, die an dieser Stelle nur willkürlich gewählt werden könnten. Die Größe des Designs im Sinne der erforderlichen Logikressourcen wird in dieser Arbeit ohnehin ausgewertet und getrennt ausgewiesen. Auf eine zwangsläufig von willkürlichen Annahmen dominierte Ableitung eines Zahlenwertes für die statische Verlustleistung wird deshalb verzichtet.

4.6 Evaluation und Analyse

Die Evaluation wird durch einen Vergleich zum Stand der Technik vorgenommen. Da die digitale Bildverarbeitung eines der wichtigsten Anwendungsfelder von Approximate Computing ist, wurde dafür eine für diesen Anwendungsbereich typische Wortbreite von 8 Bit gewählt. Diese wird auch im Bereich des ML, insbesondere der Inferenz von DNNs verwendet [A 163], [A 164], [A 147] und unter anderem von Intels Xeon-Prozessoren und dem dazugehörigen Framework „IntelCaffee“ explizit unterstützt [A 165]. Die verschiedenen Ansätze werden anhand der in Abschnitt 4.2 diskutierten Gütemaße sowie hinsichtlich ihres Ressourcenbedarfs, ihres Delays und ihrer Leistungsaufnahme verglichen. Wie in den meisten vergleichbaren Arbeiten wird ein Virtex-7 FPGA des Marktführers Xilinx als Zielplattform gewählt – konkret das Modell *xc7vx485tffg1761-2*.

Um die Vergleichbarkeit der Angaben sicherzustellen, wurden die Kennzahlen und Metriken des veröffentlichten Stands der Technik grundsätzlich komplett neu ermittelt. Dazu wurden Quelldateien genutzt, die als Open Source verfügbar sind. Wo dies nicht der Fall war, aber aufgrund der angegebenen Kennzahlen ein besonderes Interesse zu einem genauen Vergleich mit dem in dieser Arbeit vorgestellten approximierten Multiplizierer besteht, wurden auch ausgewählte Multiplizierer-Designs nachimplementiert. Wo dies nicht möglich war, oder der Aufwand nicht

gerechtfertigt erschien, wird auf Literaturangaben zurückgegriffen und dies entsprechend gekennzeichnet. Die 8-Bit-Implementierung des in dieser Arbeit vorgeschlagenen approximierten Multiplizierers steht ebenfalls als Open Source unter `https://github.com/niemann-c/approx-mult-for-fpga` zur Verfügung.

4.6.1 Ergebnisse

Einige Ansätze erzielen eine deutlich höhere Ressourcen-Einsparung als andere. Allerdings schwankt auch die Qualität der Approximation um mehrere Größenordnungen. Deshalb werden die Designs bezüglich der verschiedenen Gütemaße der Approximation und Kostengrößen (benötigte Logik-Ressourcen, Leistungsaufnahme und Delay) in Diagrammen paarweise ausgewertet.

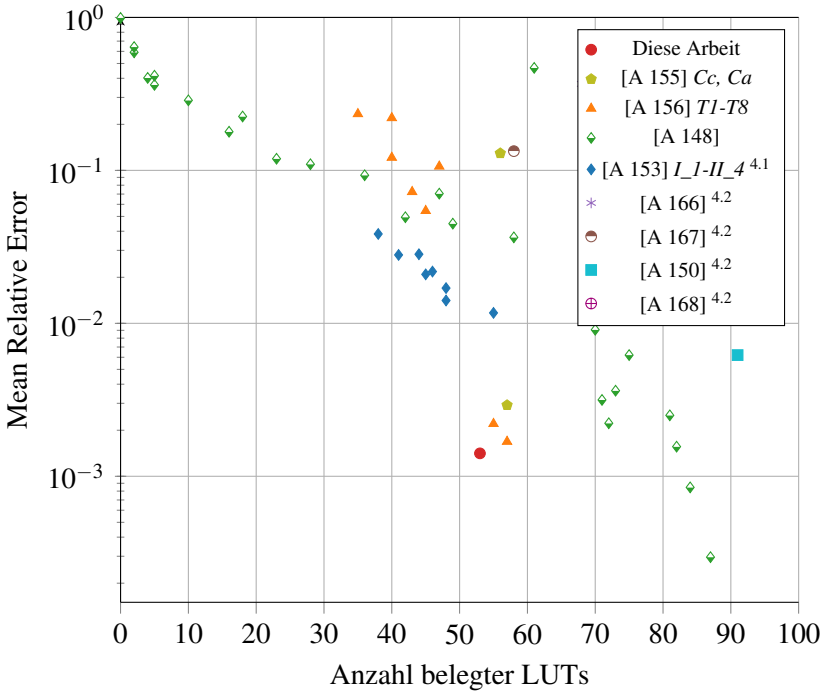


Abbildung 4.7: Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Mean Relative Error (MRE) (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE)

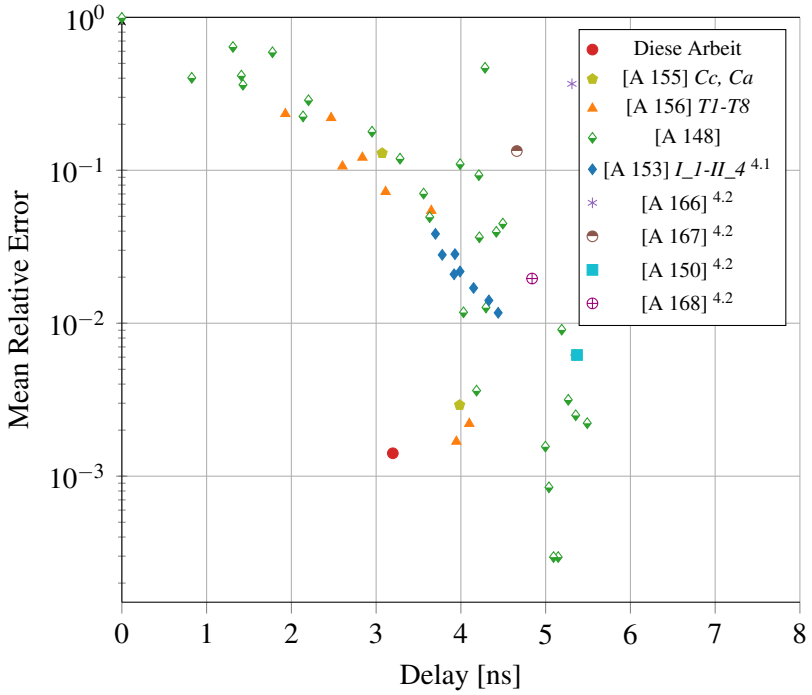


Abbildung 4.8: Vergleich mit dem Stand der Technik bezüglich Delay und MRE (Abbildung übersetzt aus [B 4]; Original ©2021 IEEE)

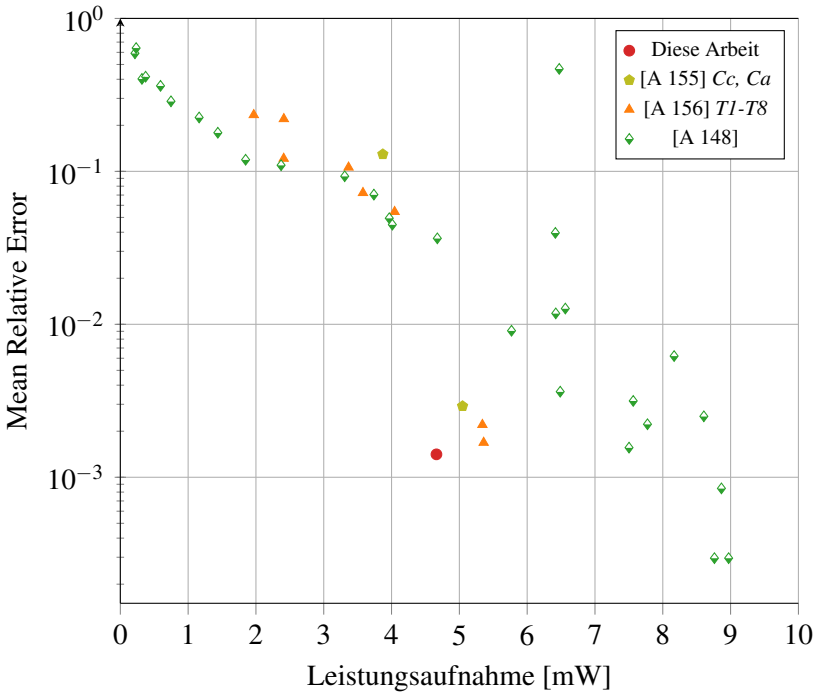


Abbildung 4.9: Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und MRE

Wie in Abschnitt 4.2 dargelegt, wird der Mean Relative Error (MRE) im Rahmen dieser Arbeit als entscheidende Zielgröße sowohl zur Optimierung als auch zur Bewertung angesehen. Die Abbildungen 4.7, 4.8 und 4.9 zeigen, dass der in dieser Arbeit entwickelte approximierte Multiplizierer bezüglich dieses Gütemaßes gegenüber dem Stand der Technik eine deutliche Verbesserung darstellt. Dies gilt bezüglich aller drei betrachteten Kostengrößen. Der durchschnittliche relative Fehler des Designs beträgt lediglich 0,14 %.

Da in der Literatur auch andere Gütemaße verbreitet sind, werden auch diese ausgewertet, um einen möglichst umfassenden Vergleich zu ermöglichen. Alle verwendeten Gütemaße wurden in Abschnitt 4.2 definiert und erläutert.

[4.1] Die Daten wurden aus der angegebenen Quelle [A 153] entnommen. Zielplattform ist ein Spartan-6 FPGA.
[4.2] Die Daten wurden aus [A 153] entnommen. Zielplattform ist ein Spartan-6 FPGA.

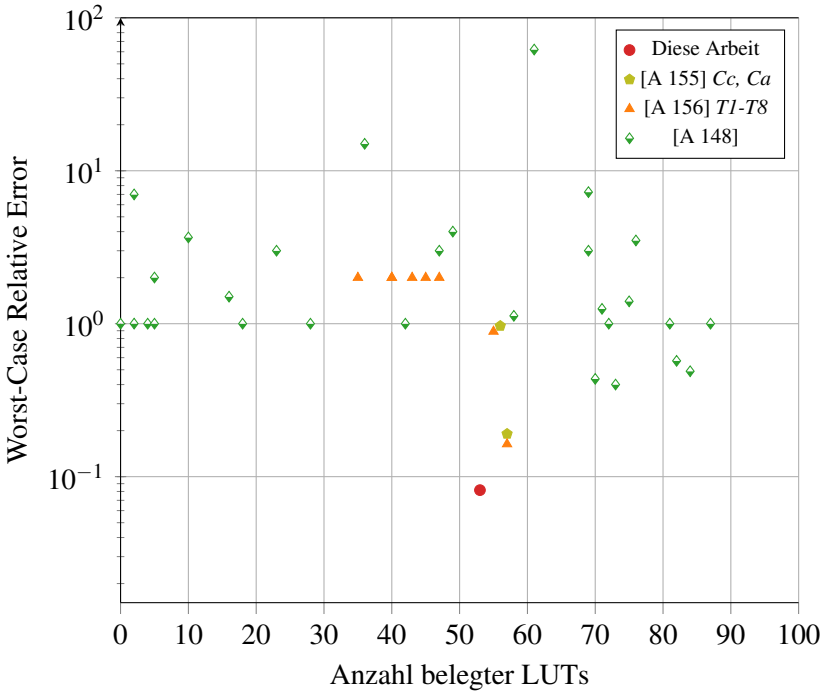


Abbildung 4.10: Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Worst-Case Relative Error (WCRE)

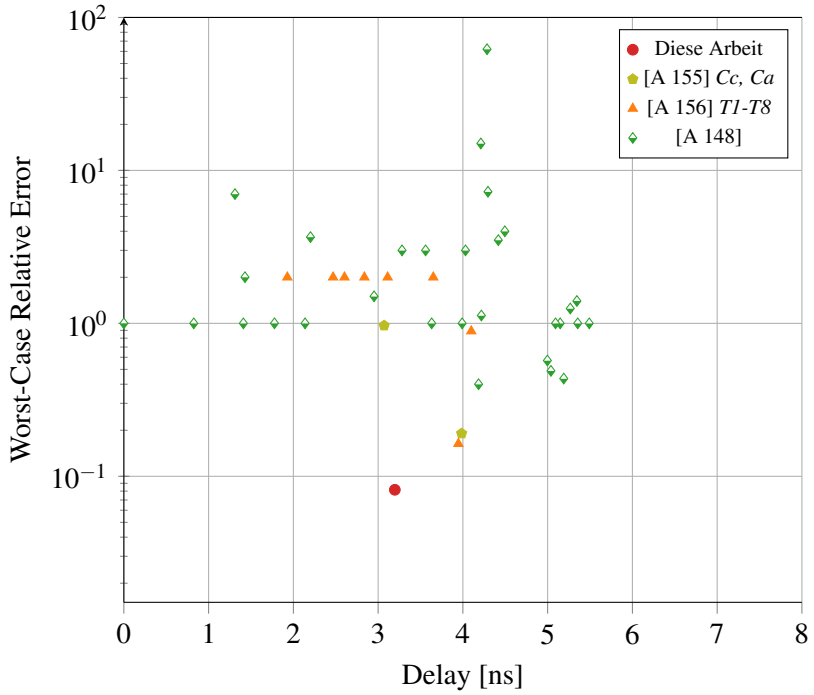


Abbildung 4.11: Vergleich mit dem Stand der Technik bezüglich Delay und WCRE

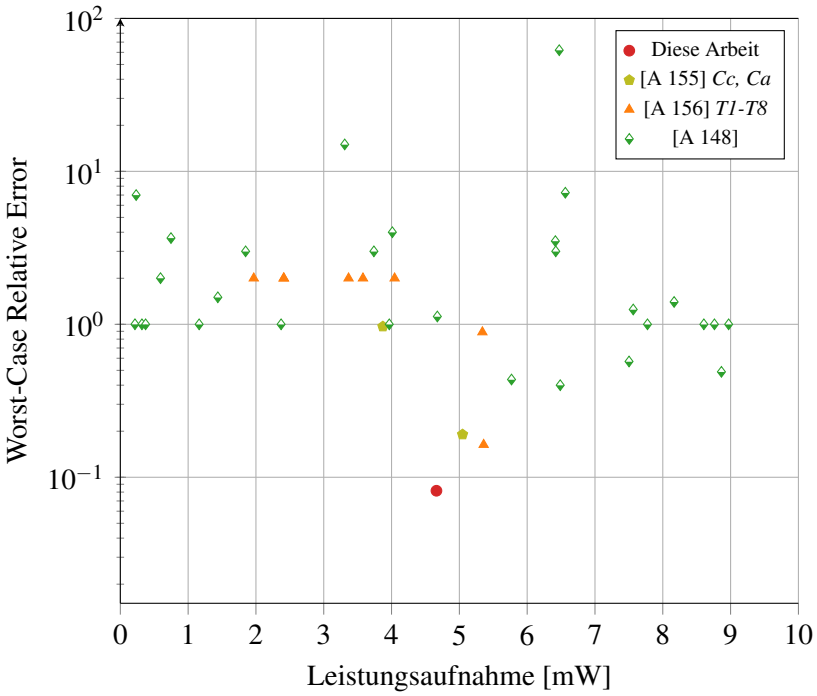


Abbildung 4.12: Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und WCRE

In den Abbildungen 4.10, 4.11 und 4.12 wird der Worst-Case Relative Error (WCRE) betrachtet, der an Stelle des Mittelwerts den ungünstigsten Fall betrachtet. Auch bezüglich dieser Metrik stellt der hier vorgestellte approximierte Multiplizierer ein Pareto-Optimum dar. Er weist darüber hinaus sogar den geringsten WCRE aller verglichenen Designs auf. Sein größter Fehler in Relation zum exakten Output ist mit 8,16 % erheblich geringer als der in dieser Hinsicht engste Konkurrent *[156] Design T1* mit 16,3 %. Bemerkenswert ist auch, dass die meisten Designs aus dem Stand der Wissenschaft und Technik einen WCRE von 100 % oder mehr aufweisen. Bei ungünstigen Kombinationen von Eingangsvektoren ist die Abweichung bei der Mehrheit der Designs also mindestens so groß wie der korrekte Output. Im Extremfall des Designs *[A 148] mul8u_1G9M* beträgt der gemachte Fehler sogar 6200 % des korrekten Wertes. Natürlich muss berücksichtigt werden, dass Prabakaran et al. ihre Designs nicht nach diesem Gütemaß ausgewählt haben und Anwendungen vorstellbar sind, in denen der relative Fehler keine entscheidende Rolle spielt. Dennoch zeigen die Ergebnisse aus Sicht des Autors dieser Arbeit deutlich, dass relative Fehlermaße bei der Konstruktion und Bewertung von approximierten Multiplizierern stärker berücksichtigt werden sollten.

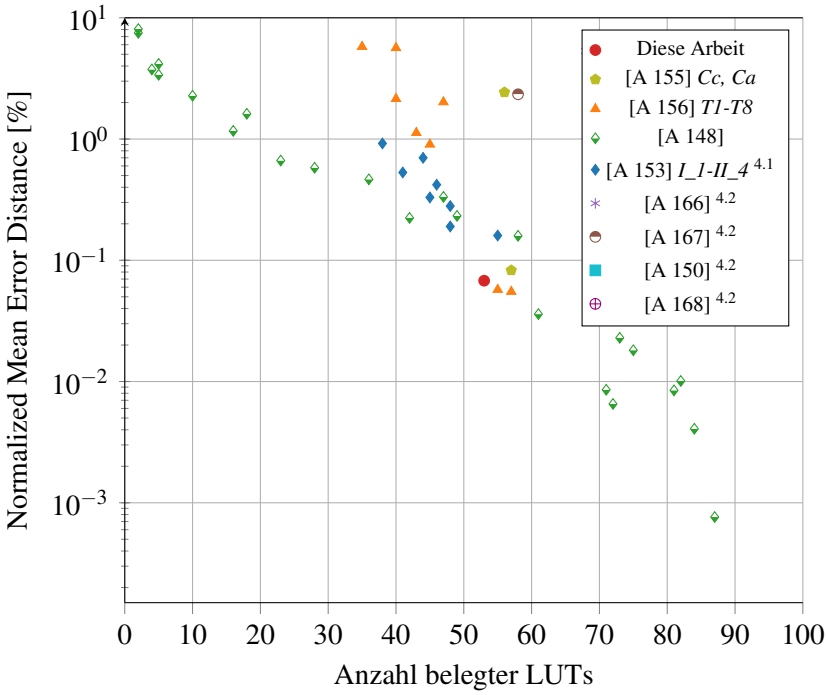


Abbildung 4.13: Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Normalized Mean Error Distance (NMED)

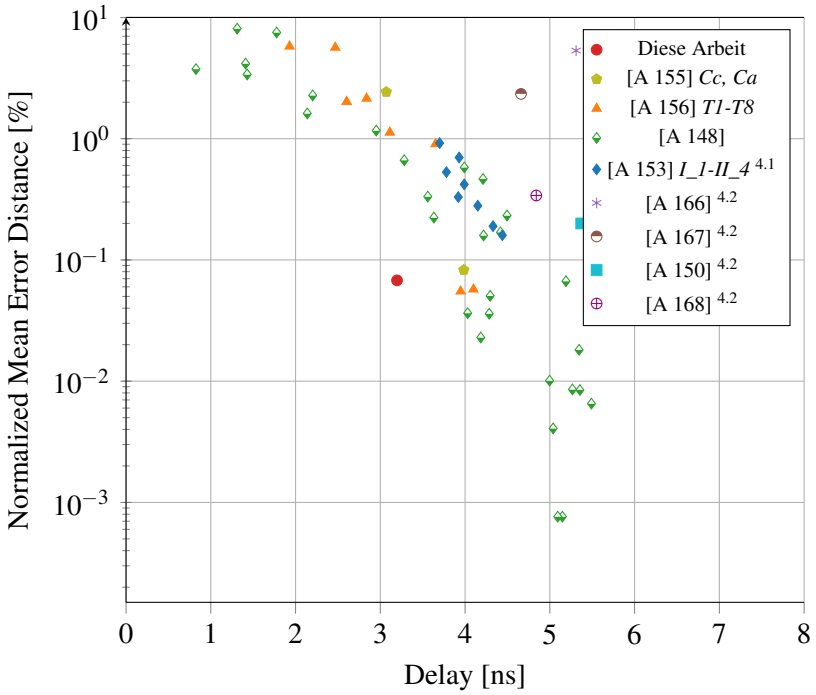


Abbildung 4.14: Vergleich mit dem Stand der Technik bezüglich Delay und NMED

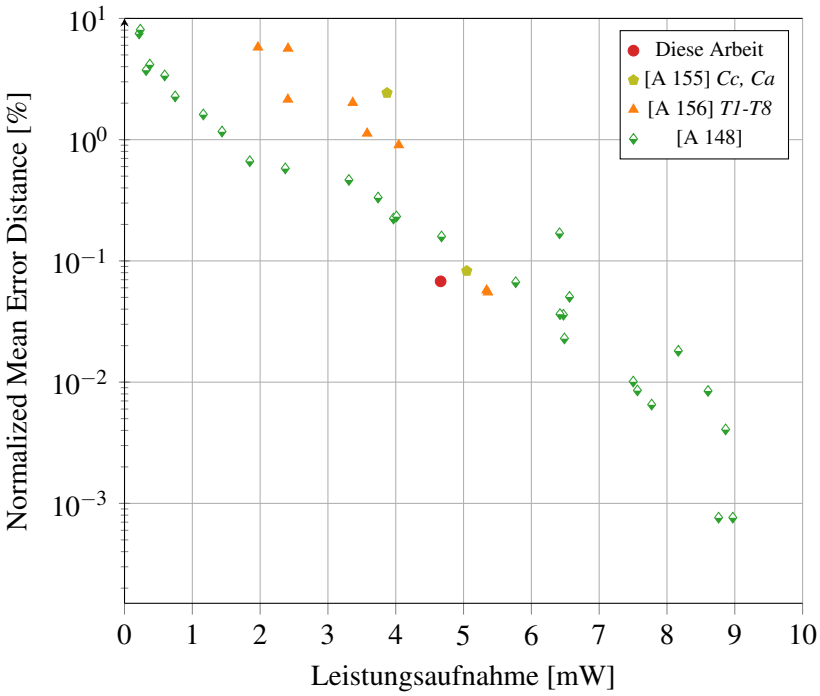


Abbildung 4.15: Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und NMED

Ein in der Literatur relativ gebräuchliches Gütemaß ist die Normalized Mean Error Distance (NMED). Dabei handelt es sich um den mittleren absoluten Fehler, der auf den größtmöglichen korrekten Ausgangswert normiert wird. Die Abbildungen 4.13, 4.14 und 4.15 zeigen, dass der hier vorgestellte Ansatz auch bezüglich der NMED, also einer Metrik, für die er nicht konzipiert und optimiert wurde, Pareto-optimal ist.

[4.3] Die Daten wurden aus der angegebenen Quelle [A 153] entnommen. Zielplattform ist ein Spartan-6 FPGA.
[4.4] Die Daten wurden aus [A 153] entnommen. Zielplattform ist ein Spartan-6 FPGA.

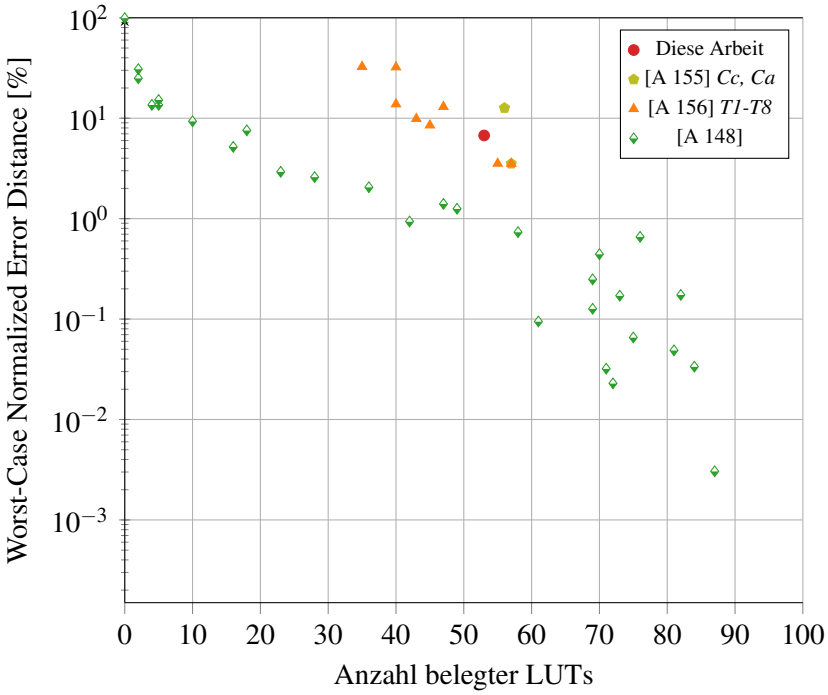


Abbildung 4.16: Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Worst-Case Normalized Error Distance (WCNED)

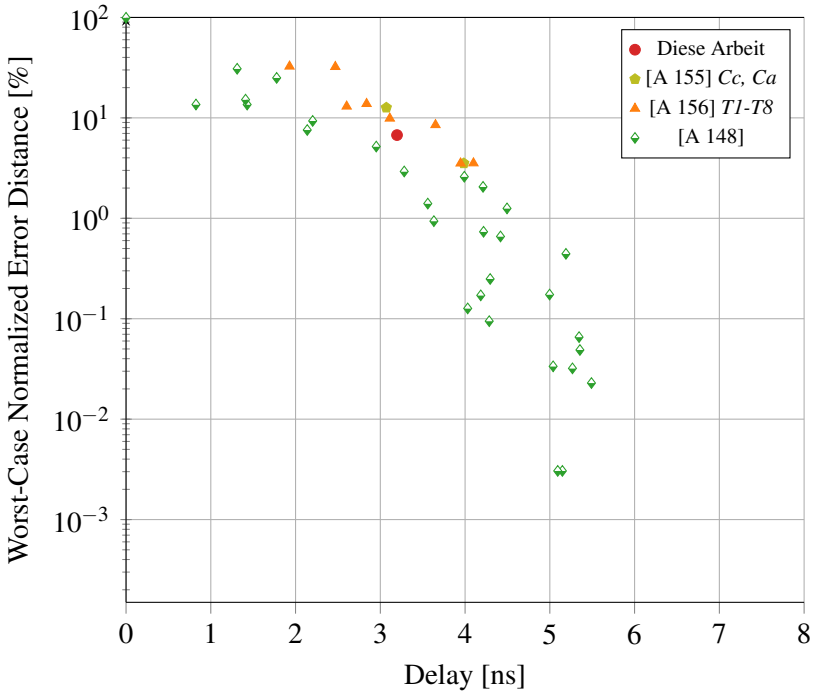


Abbildung 4.17: Vergleich mit dem Stand der Technik bezüglich Delay und WCNED

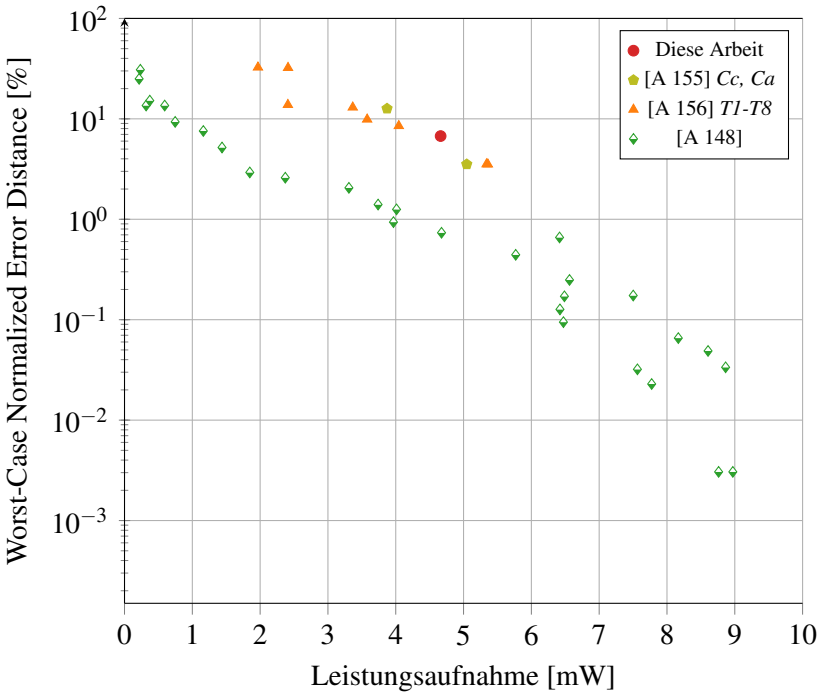


Abbildung 4.18: Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und WCNED

Wie bei den relativen Fehlermaßen lässt sich auch in Anlehnung an die NMED ein Gütemaß definieren, das auf den ungünstigsten Fall abstellt. Dieses wird hier Worst-Case Normalized Error Distance (WCNED) genannt und berechnet sich nach Gleichung 20. Die Abbildungen 4.16, 4.17 und 4.17 zeigen die Auswertung der verschiedenen Designs bezüglich dieser Metrik. Im Falle der WCNED ist der in dieser Arbeit entwickelte approximierte Multiplizierer nicht Pareto-optimal. Dies war auch so zu erwarten, da im Rahmen dieser Arbeit, wie in Abschnitt 4.2 erläutert, der relative Fehler, genauer der MRE, als geeignete Zielgröße der Entwurfsmethodik für approximierte Multiplizierer identifiziert wurde. Dementsprechend wurde die Approximation so gestaltet, dass die sehr guten Ergebnisse bezüglich relativer Fehlermetriken wie MRE und WCRE möglich sind. Konkret wird ein approximierter Kompressor eingesetzt, der nur beim größtem exaktem Ausgabewert einen Fehler macht. Demzufolge ist jedoch der absolute Fehler im ungünstigsten Fall vergleichsweise groß. Das sehr gute Abschneiden des hier vorgestellten Multiplizierer-Designs bezüglich der ebenfalls auf Absolutwerte abstellenden NMED ist dagegen eher nicht zu erwarten gewesen und belegt seine Qualität.

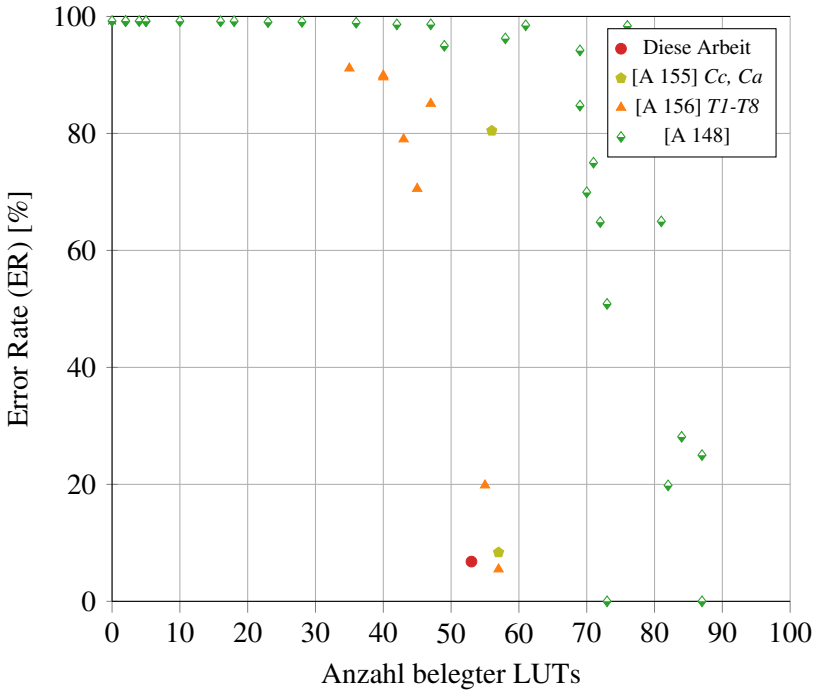


Abbildung 4.19: Vergleich mit dem Stand der Technik bezüglich Ressourcenbedarf und Error Rate (ER)

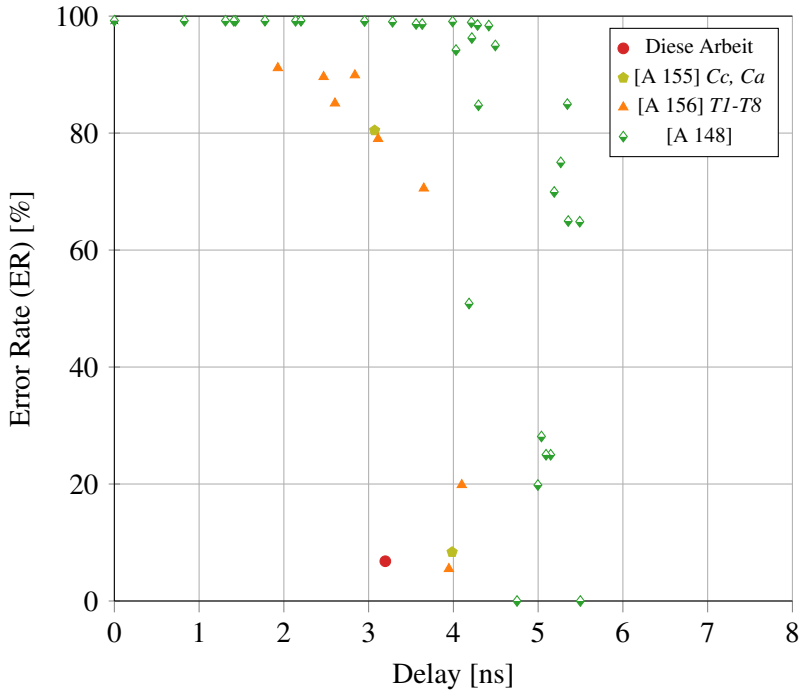


Abbildung 4.20: Vergleich mit dem Stand der Technik bezüglich Delay und ER

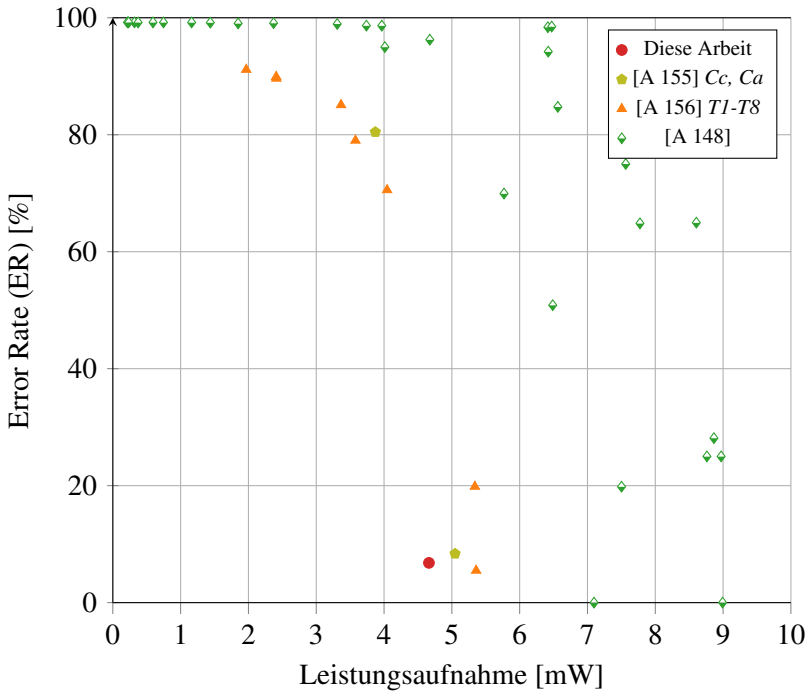


Abbildung 4.21: Vergleich mit dem Stand der Technik bezüglich Leistungsaufnahme und ER

Bezüglich der Error Rate (ER), also der Häufigkeit fehlerhafter Ergebnisse, zeigt der hier vorgestellte approximierte Multiplizierer ein sehr günstiges Verhalten und ist diesbezüglich Paretooptimal. In den Abbildungen 4.19, 4.20 und 4.21 wird diese Arbeit im Vergleich mit dem Stand der Wissenschaft und Technik bezüglich dieses Gütemaßes ausgewertet.

Die ermittelten genauen Zahlenwerte aller betrachteten Metriken und Kostengrößen ausgewählter Designs sind in Tabelle 4.7 aufgeführt.

Tabelle 4.7: Vergleich mit dem Stand der Technik von approximierten 8-Bit-Multiplizierern. Die Ergebnisse wurden, wo nicht anders gekennzeichnet, durch Implementierung auf einer identischen Toolchain und erschöpfende Simulation ermittelt.

Design	Ansatz	Fläche [LUTs]	P_{dyn} [mW]	Delay [ns]	MRE	WCRE	NMED [%]	WCNED [%]	ER [%]
Vivado-Synthese	exakter Multiplizierer	72	6,34	4,156	0,0000	0,0000	0,0000	0,0000	0,0000
[A 155] Design Cc	Rekursive Synthese aus kleineren Multiplizierern	56	3,87	3,071	0,1294	0,9675	2,4296	12,6467	80,4611
[A 155] Design Ca	Rekursive Synthese aus kleineren Multiplizierern	57	5,05	3,985	0,0029	0,1905	0,0827	3,5279	8,3649
[A 156] T1	Rekursive Synthese aus kleineren Multiplizierern	57	5,36	3,947	0,0017	0,1633	0,0551	3,5279	5,4932
[A 156] T2	Rekursive Synthese aus kleineren Multiplizierern	55	5,34	4,099	0,0022	0,8889	0,0572	3,5401	19,8486
[A 156] T3	Rekursive Synthese aus kleineren Multiplizierern	45	4,05	3,651	0,0544	2,0000	0,9030	8,5084	70,5566
[A 156] T4	Rekursive Synthese aus kleineren Multiplizierern	43	3,58	3,112	0,0724	2,0000	1,1258	9,8756	79,0207
[A 156] T5	Rekursive Synthese aus kleineren Multiplizierern	47	3,37	2,603	0,1062	2,0000	2,0174	13,0129	85,0937
[A 156] T6	Rekursive Synthese aus kleineren Multiplizierern	40	2,41	2,838	0,1212	2,0000	2,1468	13,8064	89,9109
[A 156] T7	Rekursive Synthese aus kleineren Multiplizierern	40	2,41	2,469	0,2202	2,0000	5,6490	32,2393	89,6164
[A 156] T8	Rekursive Synthese aus kleineren Multiplizierern	35	1,97	1,929	0,2339	2,0000	5,7753	32,5566	91,1346
[A 153] AM_II_4 [4.1]	Approximierte Kompressoren	48	N/A	4,33	0,014	N/A	0,19	N/A	N/A
[A 153] AM_I_4 [4.1]	Approximierte Kompressoren	55	N/A	4,44	0,0117	N/A	0,16	N/A	N/A
[A 148] mul8u_17A6	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	2	0,22	1,777	0,5914	1,0010	7,5142	25,0065	99,2172
[A 148] mul8u_17BE	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	10	0,75	2,202	0,2871	3,6641	2,2749	9,3217	99,2004
[A 148] mul8u_18G6	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	23	1,85	3,282	0,1191	3,0000	0,6646	2,9252	99,0326
[A 148] mul8u_176X	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	36	3,31	4,212	0,0929	15,0000	0,4652	2,0554	98,9304
[A 148] mul8u_QM8	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	47	3,74	3,560	0,0705	3,0000	0,3331	1,4008	98,6542
[A 148] mul8u_FE8	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	49	4,01	4,495	0,0448	4,0000	0,2325	1,2543	94,9814
[A 148] mul8u_GT4	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	58	4,68	4,218	0,0365	1,1250	0,1591	0,7340	96,2494
[A 148] mul8u_1G9M	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	61	6,47	4,285	0,4678	62,0000	0,0360	0,0946	98,4863
[A 148] mul8u_10K5	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	70	5,77	5,190	0,0091	0,4356	0,0668	0,4425	69,9371
[A 148] mul8u_5E8	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	75	8,17	5,345	0,0062	1,4000	0,0181	0,0656	84,9609
[A 148] mul8u_19PQ	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	76	6,42	4,418	0,0396	3,5000	0,1690	0,6592	98,3810
[A 148] mul8u_ABX	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	84	8,86	5,039	0,0008	0,4889	0,0041	0,0336	28,1250
[A 148] mul8u_50M	ML-gestützte Auswahl aus Bibliothek von ASIC-Designs	87	8,76	5,148	0,0003	1,0000	0,0008	0,0031	25,0000
Diese Arbeit	LUT-optimierte approximierte Kompressoren und statistisch optimierter Kompressionsbaum	53	4,66	3,197	0,0014	0,0816	0,0678	6,7384	6,7978

[4.1] Daten aus der Literatur; Zielplattform: Spartan-6 FPGA

4.6.2 Skalierbarkeit

Im Rahmen der vorliegenden Arbeit werden primär approximierte 8-Bit Multiplizierer behandelt, da diese Bitbreite in wichtigen Anwendungsbereichen von Approximate Computing wie der digitalen Bildverarbeitung und DNNs weit verbreitet ist. Allerdings ist die hier vorgestellte Methodik unabhängig von der konkreten Bitbreite. Insbesondere ist sie für größere Multiplizierer geeignet. Wie in Unterabschnitt 4.4.2 erläutert, weist die Partialproduktbildung ein konstantes Delay auf. Der Kompressionsbaum aus approximierten Kompressoren weist ein geringeres Delay auf als ein in gleicher Vorgehensweise konstruierter Kompressionsbaum aus exakten Kompressoren. Seine asymptotische Zeitkomplexität beträgt jedoch ebenfalls $O(log(n))$. Lediglich der abschließende Ripple-Carry Addierer (RCA) skaliert mit $O(n)$ und wird somit für große Bitbreiten das Zeitverhalten dominieren. Da RCAs jedoch durch die dedizierte Carry-Logik auf FPGAs vergleichsweise schnell sind, ist dennoch auch für größere Multiplizierer ein vergleichsweise geringes Delay zu erwarten. Um dies exemplarisch zu demonstrieren, wurde ein 16x16 Bit Multiplizierer nach der vorgeschlagenen Methodik implementiert. Das Design wurde dann mit der in Abschnitt 4.5 beschriebenen Evaluationsmethodik untersucht. Der Multiplizierer weist ein Delay von lediglich 4,309 ns auf und belegt 220 LUTs. Ein Anstieg des Delays um 35 % bei einer Verdopplung der Wortbreite liegt im Rahmen des aufgrund der asymptotischen Abschätzung Erwartbaren. Als Vergleich wurde die als Open Source verfügbare 16-Bit-Erweiterung von *[A 155] (Design Ca)* unter gleichen Bedingungen synthetisiert und implementiert. Diese benötigt 245 LUTs und verursacht ein Delay von 6,207 ns. Eine detaillierte Erörterung und ein umfassender Vergleich von approximierten 16-Bit-Multiplizierern liegt allerdings außerhalb des Rahmens der vorliegenden Arbeit.

Die im Bezug auf die meisten Gütemaße und Kostengrößen kompetitivsten Alternativen zu der im Rahmen dieser Arbeit entwickelten Methodik sind die Arbeiten von Guo et al. [A 156] und Ullah et al. [A 155]. Diese beruhen beide auf der rekursiven Synthese größerer Multiplizierer aus kleineren Basis-Multiplizierern. Mit diesem Verfahren lässt sich die Bitbreite jedoch nicht beliebig wählen, sondern kann nur verdoppelt oder halbiert werden. Sowohl in der Signalverarbeitung als auch im Bereich des ML wird jedoch häufig auch mit anderen Wortbreiten oder gar mit zwei unterschiedlich dimensionierten Inputs gearbeitet. Da die hier vorgeschlagene Methodik zur Konstruktion von approximierten Multiplizierern auf dem klassischen Prinzip eines Kompressionsbaumes beruht, kann die Bitbreite frei gewählt werden. Lediglich bei der Partialprodukterzeugung müssen bei bestimmten Bitbreiten verhältnismäßig mehr Partialproduktbits durch `AND`-Operationen erzeugt werden, was einen etwas erhöhten Ressourcenbedarf verursacht.

4.6.3 Anwendungsbeispiel: approximierter Faltungskern für digitale Bildverarbeitung

Die praktische Anwendbarkeit des Ansatzes wird hier anhand eines Faltungskerns für die digitale Bildverarbeitung demonstriert. Solche zweidimensionalen Faltungskerne werden in der Bildverarbeitung für viele wichtige Basis-Operationen wie Kantenerkennung, Weichzeichnung und viele weitere verwendet. Konkret wird ein Filterkern realisiert, der ein 3x3-Pixel-Fenster pro Taktzyklus verarbeitet. Die Resultate lassen sich jedoch gut auf größere Kerne übertragen. Als Basis für die Vergleiche wurde ein Filterkern mit Wallace-Tree-Multiplizierer implementiert. Als Akkumulatorstufe werden ein Carry-Save-Kompressionsbaum aus exakten 4:2-Kompressoren und ein abschließender RCA genutzt. Das hier vorgeschlagene Multiplizierer-Design wird eben-

Tabelle 4.8: Ergebnisse für die Anwendung verschiedener approximierter Multiplizierer in einem 3x3-Faltungskern [B 4]

Verwendete Multiplizierer und Akkumulatoren	LUTs	Relative Einsparung	Delay [ns]	Relative Beschleunigung
Exakter Wallace-Tree-Multiplizierer, exakte 4:2-Carry-Save-Kompressoren	872	—	6,046	—
Ullah et al. [A 155] *Design Cc*, RCAs	630	27,75 %	6,634	-9,73 %
Diese Arbeit, exakte 4:2-Carry-Save-Kompressoren	636	27,06 %	5,788	4,27 %
Diese Arbeit, approximierte 4:2-Carry-Save-Kompressoren	520	40,37 %	5,082	15,94 %

Abbildung 4.22: Ergebnisbilder: exakter Multiplizierer, hier vorgestellter approximierter Multiplizierer mit exakten 4:2-Carry-Save-Kompressoren, Ullah et al. [A 155] Design Cc (von links nach rechts) (Abbildung aus [B 4]; Original ©2021 IEEE)

falls mit einem Carry-Save-Kompressionsbaum mit abschließendem RCA kombiniert, wobei der abschließende RCA der Multiplizierer zugunsten einer durchgehenden Carry-Save-Darstellung entfällt. Von diesem Design wurden zwei Varianten abgeleitet. Diese unterscheiden sich durch die verwendeten 4:2-Kompressoren im Akkumulator des Filterkerns. Ein Design verwendet exakte 4:2-Kompressoren, während das andere auch im Kompressionsbaum des Filterkerns approximierte Kompressoren nutzt.

Zu Vergleichszwecken wurde auch ein Filterkern unter Verwendung des *Design Cc* von Ullah et al. [A 155] implementiert. Da der Ausgang der Multiplizierer das Produkt als nichtredundante Binärdarstellung ausgibt, wurde die Akkumulatorstufe aus RCAs realisiert. Die Designs wurden mit Vivado für einen Virtex-7 synthetisiert. Die Ergebnisse sind in Tabelle 4.8 aufgeführt. Zusätzlich wurde eine (Gauß-)Weichzeichnerfunktion mit den verschiedenen Filterkernimplementierungen realisiert. Abbildung 4.22 zeigt vergleichend die Ergebnis-Bilder. Sie zeigen deutlich, dass unser Ansatz für praktische Anwendungen geeignet ist. Bemerkenswert ist die enorme Ersparnis an Ressourcen von über 40 % während das Design um 16 % schneller arbeitet.

4.7 Zwischenfazit

In diesem Kapitel wurde eine Methodik zum Entwurf von approximierten Multiplizierern vorgestellt. Dabei wurden zunächst geeignete Zielgrößen von Approximationen diskutiert und festgestellt, dass diese sich an der Art der Fehlertoleranz der Zielanwendung ausrichten sollten. Beruht diese auf der Imperfektion menschlicher Sinneswahrnehmung, sollten die Gesetze der Psychophysik, insbesondere Webers Gesetz, berücksichtigt werden. In technischen Systemen ist häufig das SNR entscheidend. In beiden Fällen bilden absolutwertbasierte Metriken die Charakteristik der Fehlertoleranz nicht hinreichend ab, weshalb relative Fehlermaße, im Speziellen der MRE, als Zielgröße bei der Entwicklung und Bewertung gewählt werden sollten.

Hauptbeitrag ist eine Entwurfsmethodik für approximierte Multiplizierer, die auf das MRE als Zielgröße abstellt. Diese basiert auf drei Bausteinen:

- Approximierte Kompressoren, die effizient auf die Logikressourcen von FPGAs abgebildet werden können und nur anstelle des größten exakten Ausgangswertes einen abweichenden Wert ausgeben, wodurch die relative Abweichung gering gehalten wird
- Einer für FPGA optimierten Methodik zur Erzeugung und Teilkompression der Partialprodukte
- Der statistischen Optimierung der Zuordnung der teilkomprimierten Partialprodukte zu den Eingängen der approximierten Kompressoren

Zur Evaluation wurde ein nach der vorgeschlagenen Entwurfsmethodik erstellter 8-Bit-Multiplizierer mit dem Stand der Technik verglichen. In der Literatur wird jedoch mit verschiedenen Gütefunktionen gearbeitet und der ermittelte Ressourcenbedarf ist stark durch die angewandten Definitionen, EDA-Tools und Zielplattformen beeinflusst. Um eine quantitative Vergleichbarkeit der verschiedenen Ansätze herzustellen, wurde in dieser Arbeit eine Evaluationsmethodik entwickelt und automatisiert.

Die vorgeschlagene Entwurfsmethodik für approximierte Multiplizierer stellt sich bezüglich der gewählten Zielgröße MRE als Pareto-optimal und dem bisherigen Stand der Technik klar überlegen dar. Dies gilt sowohl bezüglich der Einsparung von Logikressourcen des FPGA und Senkung der Leistungsaufnahme als auch bezüglich des Delays. Auch bei drei der vier anderen untersuchten Gütemaße werden Pareto-optimale Ergebnisse erzielt. Der Hauptfokus der vorliegenden Dissertation gilt dabei der Reduktion der Leistungsaufnahme. Im Vergleich zum Stand der Technik bietet der hier vorgeschlagene approximierte Multiplizierer hinsichtlich der Leistungsaufnahme deutliche Vorteile. So benötigt *[156]-Design T1* 15 % mehr Energie, um einen vergleichbaren (jedoch leicht schlechteren) MRE zu erzielen. Im Vergleich mit dem von der Xilinx-Synthese erstellten Multiplizierer konnte eine Reduktion der dynamischen Verlustleistung von 26,5 % realisiert werden. Der dafür in Kauf genommene durchschnittliche relative Fehler beträgt lediglich 0,14 %. Der Bedarf an Logikressourcen konnte um 26,4 % gesenkt werden. Dies ermöglicht potentiell die Wahl eines kleineren FPGA und somit einerseits eine Reduktion der statischen Leistungsaufnahme und andererseits eine Kostenersparnis.

5 Zusammenfassung

FPGAs verbinden die Leistungsfähigkeit einer Hardwareimplementierung mit großer Flexibilität. Folglich werden sie in steigendem Maße in Rechenzentren, der Kommunikationsinfrastruktur und vielen weiteren Anwendungen eingesetzt. Im Vergleich zu ASICs benötigen sie jedoch erheblich mehr Energie und sind bezüglich der erreichbaren Taktraten unterlegen. Eine Steigerung der Energieeffizienz der Informations- und Kommunikationstechnologie ist sowohl aus finanzieller als auch aus gesellschaftlicher Sicht erforderlich. Da sich Verfahren zur Steigerung der Energieeffizienz von ASICs nicht unmittelbar auf FPGAs übertragen lassen, wurden im Rahmen dieser Arbeit dedizierte Techniken zur Steigerung der Energieeffizienz von FPGA-Anwendungen entwickelt.

5.1 Adaptive Voltage Scaling (AVS)

Das Logik-Delay integrierter Schaltungen unterliegt vielfältigen Einflüssen, wie etwa der Prozessvariation, der Temperatur, Schwankungen der Versorgungsspannung und der Alterung. Diese sind zur Design-Zeit nicht bekannt. In digitalen, synchron getakteten Very Large Scale Integration (VLSI)-Systemen wird das Timing deshalb traditionell für den Worst-Case berechnet, um sicherzustellen, dass das System zuverlässig funktioniert. Für die meisten produzierten Chips und ihre Anwendungsszenarien wird dadurch jedoch ein zu pessimistischer Spannungs-Taktfrequenz-Arbeitspunkt (VF-Arbeitspunkt) gewählt. Dadurch wird bezüglich der Energieeffizienz ein erheblicher Teil des Potentials einer konkreten Prozesstechnologie nicht ausgeschöpft. Adaptive Voltage Scaling (AVS) nutzt Laufzeitwissen, um die Versorgungsspannung in einer geschlossenen Regelschleife an die vorliegenden Bedingungen anzupassen. Die Anwendung von AVS auf FPGAs unterscheidet sich dabei von der auf ASICs. Der Grund dafür ist einerseits, dass das Anwendungsdesign – und damit die kritischen Pfade – zum Zeitpunkt der Produktion des FPGA prinzipbedingt nicht bekannt sind. Zudem ändert sich das Design über die Lebensdauer des FPGAs potentiell immer wieder. Andererseits sind eine hohe Flexibilität und eine kurze Time-to-Market wichtige Gründe für den Einsatz von FPGAs. Dadurch ist ein erhöhter Aufwand bei der Erstellung des Anwendungsdesigns als besonders kritisch zu werten.

Der Stand der Wissenschaft und Technik wurde kategorisiert und die Klasse der auf Monitoring-Flipflops basierenden AVS-Systeme einer theoretischen und simulativen Analyse unterzogen. Dabei wurde herausgearbeitet, dass diese Klasse von Ansätzen prinzipbedingt eine datenabhängige Fehleranfälligkeit aufweist. Mit umfangreichen Simulationen wurde diese Fehleranfälligkeit erstmals quantitativ bestimmt. Es wurden mehrere in der Literatur angegebene Lösungen zur Behebung dieser Fehleranfälligkeit untersucht und belegt, dass diese nicht geeignet sind, die Fehler hinreichend zu unterbinden. Es wurde gefolgert, dass Monitoring-Flipflop-basiertes AVS eher dem Voltage Underscaling im Sinne des Approximate Computing zuzurechnen ist.

Die vorliegende Arbeit konzentriert sich auf sensorbasiertes AVS. In der Literatur werden bereits vielfältige Sensordesigns für AVS auf FPGAs vorgeschlagen. In dieser Arbeit wurde zunächst analysiert, dass für ein AVS-System weitere essentielle Aspekte berücksichtigt werden müssen. Als drei solche Aspekte wurden die Kalibrierung der Sensoren, das Placement selbiger sowie die Bemessung eines residualen Guard Bands herausgearbeitet.

Im Rahmen dieser Arbeit wurde eine Offline-Sensorkalibrierung entwickelt. Diese basiert auf einer stufenweisen Absenkung der Versorgungsspannung, wobei auf jeder Stufe mittels geeigneter Path-Delay-Tests geprüft wird, ob das Design am jeweiligen VF-Arbeitspunkt zuverlässig arbeitet. Ein auftretender Timing-Fehler grenzt die kritische Versorgungsspannung auf einen Wert zwischen der aktuellen und der letzten als fehlerfrei getesteten Versorgungsspannung ein. Bei letzterer, also dem niedrigsten als fehlerfrei getesteten Spannungslevel, wird der jeweilige Sensor ausgelesen. Um eine valide Führungsgröße für die Regelung der Versorgungsspannung zu erhalten, wird ein Fast Fluctuation Guard Band (FFGB) genanntes residuales Guard Band angewendet, indem dieser Sensorwert mit einem entsprechenden Skalierungsfaktor multipliziert wird. Zur Bestimmung des FFGB werden für die Auswirkungen des Voltage Drops auf das Delay Werte aus der Literatur herangezogen. Diese basieren auf einem, in anderem Zusammenhang in der Literatur vorgeschlagenen, experimentellen Vorgehen zur Bestimmung des Voltage Drops auf FPGAs. Zusätzlich wird die relative Sensorabweichung berücksichtigt. Diese wurde bei, im Rahmen dieser Arbeit durchgeführten, Voruntersuchungen der eingesetzten Sensoren ermittelt.

Als Sensoren werden in dieser Arbeit Ringoszillatoren (ROs) verwendet, wie sie auch in der Literatur geläufig sind. Im Gegensatz zu der sonst üblichen Realisierung als Hard Macro mit festgelegtem Placement und Routing wird in dieser Arbeit jedoch eine lediglich örtlich eingegrenzte Platzierung vorgeschlagen. Diese vermeidet die Blockade ganz bestimmter Ressourcen durch die Sensoren und ermöglicht dadurch die Verwendung größerer Sensoren, von deren Delay ein relevanter Anteil durch Routing verursacht wird. Durch eine Reihe von Experimenten wurde gezeigt, dass diese Sensorplatzierung in Verbindung mit der vorgeschlagenen Sensorkalibrierung tatsächlich geeignet ist, um AVS-Sensoren zu platzieren. Mit der vorgeschlagenen Platzierungsmethode ergab sich sogar eine etwas geringere Worst-Case-Sensorabweichung vom Verhalten der untersuchten Logikpfade. Für diese Experimente musste die Temperatur des Chips manipuliert und Voltage Drop herbeigeführt werden. Beides lässt sich über Heater Cores realisieren. Da bestehende Lösungen nicht die gestellten Anforderungen erfüllen, wurde hierzu ein verbessertes Heater-Design vorgeschlagen und umgesetzt.

Das vorgeschlagene Konzept wurde prototypisch implementiert und für zwei verschiedene Anwendungsdesigns aus dem Bereich der digitalen Signalverarbeitung umgesetzt. Zum einen wurde ein relativ kleines Anwendungsdesign realisiert, bei dem die Leistungsaufnahme durch Leckströme das Verhalten dominiert – eine im Zuge des *Dark Silicon* typische Charakteristik. Zum anderen wurde ein großes Design mit hoher Schaltaktivität implementiert, dessen Leistungsaufnahme durch die dynamische Verlustleistung dominiert wird. Beide Designs wurden auf fünf baugleichen FPGA-Boards evaluiert, um den Einfluss der Prozessvariation und der Alterung zu berücksichtigen. Die Versorgungsspannung konnte bei konstanter Taktfrequenz um durchschnittlich 21 % niedriger gewählt werden. Dadurch sank der Energiebedarf für das von der dynamischen Leistungsaufnahme dominierte Design um 51 %. Für das von der Leakage Power dominierte Design ergab sich eine nochmals größere Energieersparnis von durchschnittlich 66 %.

5.2 Approximate Computing

Viele der Anwendungen, die entscheidend zum wachsenden Bedarf an Rechenleistung führen, sind einerseits für die Beschleunigung mittels FPGA geeignet und weisen andererseits ein Aus-

maß an Fehlertoleranz auf, das die Anwendung von Approximate Computing ermöglicht. Im Rahmen dieser Arbeit wurden approximierte Multiplizierer als eine mögliche Umsetzung des Approximate-Computing-Paradigmas erforscht.

Dafür wurden zunächst die relevanten Besonderheiten von FPGAs im Vergleich zu ASICs erörtert und auf dieser Grundlage herausgearbeitet, wie ein für FPGA optimierter approximierter Kompressor als Grundbaustein von Multiplizierer-Designs beschaffen sein muss. Da dieser Kompressor sich jedoch nicht gut mit der herkömmlichen Erzeugung der Partialprodukte mittels `AND`-Gatter vereinen lässt, wurde eine auf die Logikressourcen aktueller FPGAs abgestimmte Erzeugung der Partialprodukte erarbeitet. Diese basiert auf zwei verschiedenen sogenannten Partialprodukt-Blöcken (PPBs), die sich nach einem einfachen Schema zur Konstruktion von Multiplizierern beliebiger Größe kombinieren lassen.

Im Falle exakt arbeitender Multiplizierer ist ein Vertauschen von Signalen, die den gleichen Zahlenwert repräsentieren, für das Ergebnis irrelevant. Auf approximierte Multiplizierer – zumindest sofern sie aus approximierten Kompressoren aufgebaut sind – trifft diese Annahme jedoch im Allgemeinen nicht zu. Vielmehr ändert die Zuordnung der Signale zu bestimmten approximierten Kompressoren sowohl die Wahrscheinlichkeit, mit der eine Abweichung vom exakten Ergebnis auftritt, als auch deren Ausmaß. Im Rahmen dieser Arbeit wurde dies als mögliches Optimierungspotential erkannt und eine entsprechende statistische Optimierung der Signalzuordnungen im Kompressionsbaum vorgeschlagen und umgesetzt. Nach bestem Wissen des Autors wurde dies bisher in dieser Form nicht veröffentlicht.

Das vorgeschlagene Design wurde ausführlich evaluiert. Dazu wurde eine umfassende Evaluationsmethodik entwickelt, bei der die Kostengrößen *Logikressourcenverbrauch*, *Energiebedarf* und *Delay* sowie fünf verschiedene Fehlermetriken berücksichtigt werden. Die Fehlermetriken wurden diskutiert. Zudem wurde herausgearbeitet, dass den relativen Fehlermaßen ein hoher Stellenwert bei der Bewertung von Approximate-Computing-Designs eingeräumt werden sollte. Dies ist in der Literatur aber bisher nicht der Fall.

Das im Rahmen dieser Arbeit entwickelte Design erweist sich bei vier der fünf Fehlermetriken als paretooptimal bezüglich aller drei Kostengrößen. Der als entscheidende Metrik zur Optimierung und Bewertung herausgearbeitete Mean Relative Error (MRE) beträgt lediglich 0,14 %. Durch Inkaufnahme dieses vergleichsweise kleinen Fehlers konnte die dynamische Verlustleistung im Vergleich zu dem von der Vivado-Synthese erzeugten exakten Multiplizierer um 26,5 % reduziert werden. Auch das Delay ist um 23,1 % niedriger. Somit konnten erhebliche Verbesserungen bezüglich der beiden entscheidenden Schwachpunkte von FPGAs, nämlich des größeren Energiebedarfs und der langsameren Schaltgeschwindigkeit, erzielt werden.

5.3 Ausblick

Für die tatsächliche breite Anwendung von im Rahmen akademischer Arbeiten erforschten Verfahren in der Halbleiterindustrie sind häufig die *Non-recurring Engineering Costs* potentieller Anwender entscheidend. Für die im Rahmen dieser Arbeit entwickelten Beiträge zur Steigerung der Energieeffizienz von FPGA-Anwendungen bedeutet dies insbesondere, dass der Mehraufwand bei der Implementierung eines Anwendungsdesigns möglichst stark reduziert werden muss. Aus Sicht des Autors entscheidet sich der diesbezügliche Erfolg auf dem Gebiet der Elec-

tronic Design Automation (EDA). Dabei ist einerseits entscheidend, inwiefern sich die erforschten Verfahren automatisieren lassen. Dabei sollte von Seiten der Anwendungsentwickler:innen möglichst wenig spezifisches Wissen über die Verfahren nötig sein und auch der Aufwand der Implementierung nicht entscheidend steigen. Andererseits müssen sich die Verfahren möglichst problemlos in die etablierten EDA-Arbeitsabläufe integrieren lassen und mit der vorhandenen Software kompatibel sein. Obwohl eine spätere praktische Anwendung und Einbindung in EDA-Arbeitsabläufe während der Forschung für diese Arbeit insbesondere auf konzeptioneller Ebene bereits berücksichtigt wurden, sind auf diesem Gebiet weitergehende Aktivitäten notwendig.

Beispielsweise unterscheiden sich für Approximate Computing in Frage kommende Anwendungen, wie etwa Video- und Bildverarbeitung oder Deep Neural Networks (DNNs), sowohl hinsichtlich Art und Ausmaß der Fehlertoleranz als auch hinsichtlich der statistischen Eigenschaften der verarbeiteten Daten. Es stellt sich also die Frage nach den passenden Approximate-Computing-Komponenten für eine konkrete Anwendung. Aus Sicht des Autors ist der große diesbezügliche Aufwand ein erhebliches Hindernis für eine breitere Anwendung von Approximate-Computing-Techniken. Notwendig wäre eine Erweiterung der Forschung hinsichtlich der automatisierten Generierung von Approximate-Computing-Komponenten nach Maßgaben der Anwendung. Dies könnte beispielsweise durch Auswahl aus einer Bibliothek von entsprechend charakterisierten Designs umgesetzt werden. Entsprechende Arbeiten wurden vom Autor bereits begonnen.

Im Bereich des AVS für FPGAs wurde die einfache Integration in bestehende Arbeitsabläufe bereits in der vorliegenden Arbeit berücksichtigt. Als Beispiel sei hier das Sensor-Placement nach dem Erstellen der eigentlichen Anwendung nur in den verbleibenden freien Ressourcen genannt. Dieses Vorgehen lässt sich sehr gut in bestehende Arbeitsabläufe integrieren, da beispielsweise keine Placement-Konflikte mit vorgegebenen IP-Cores entstehen. Dennoch gibt es Bedarf an weiterer Arbeit bezüglich der Integration in EDA-Abläufe. Insbesondere besteht bei der vollautomatischen Generierung von optimalen Testpatterns sowie bei der Test-Injection noch Forschungsbedarf. Eine weitere äußerst interessante Forschungsfrage ist die nach dem eigentlichen Reglerdesign. So beeinflusst die Frage, wie schnell eine Störung ausgeregelt werden kann, die Größe des erforderlichen FFGB. Eine Regelabweichung im Sinne eines zu pessimistischen VF-Arbeitspunktes wiederum verursacht einen höheren Energiebedarf. Werden zusätzlich örtliche Gradienten – beispielsweise der Temperatur – berücksichtigt, ergibt sich ein äußerst interessantes Forschungsfeld, welches nach bestem Wissen des Autors bisher kaum beachtet wird.

A Literaturverzeichnis

[1] G. E. Moore, „Cramming More Components onto Integrated Circuits," *Electronics*, Jg. 38, Nr. 8, S. 114–117, Apr. 1965.

[2] T. N. Theis und H.-S. P. Wong, „The End of Moore's Law: A New Beginning for Information Technology," *Computing in Science Engineering*, Jg. 19, Nr. 2, S. 41–50, 2017. DOI: 10.1109/MCSE.2017.29.

[3] R. S. Williams, „What's Next? [The end of Moore's law]," *Computing in Science Engineering*, Jg. 19, Nr. 2, S. 7–13, 2017. DOI: 10.1109/MCSE.2017.31.

[4] M. M. Waldrop, „The chips are down for Moore's law," *Nature News*, Jg. 530, Nr. 7589, S. 144, 2016.

[5] A. M. Caulfield, E. S. Chung, A. Putnam, H. Angepat, J. Fowers, M. Haselman, S. Heil, M. Humphrey, P. Kaur, J.-Y. Kim, D. Lo, T. Massengill, K. Ovtcharov, M. Papamichael, L. Woods, S. Lanka, D. Chiou und D. Burger, „A cloud-scale acceleration architecture," in *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2016, S. 1–13. DOI: 10.1109/MICRO.2016.7783710.

[6] V. Chamola, S. Patra, N. Kumar und M. Guizani, „FPGA for 5G: Re-configurable Hardware for Next Generation Communication," *IEEE Wireless Communications*, Jg. 27, Nr. 3, S. 140–147, 2020. DOI: 10.1109/MWC.001.1900359.

[7] E. Masanet, A. Shehabi, N. Lei, S. Smith und J. Koomey, „Recalibrating global data center energy-use estimates," *Science*, Jg. 367, Nr. 6481, S. 984–986, 2020. DOI: 10.1126/science.aba3758. eprint: https://www.science.org/doi/pdf/10.1126/science.aba3758. Adresse: https://www.science.org/doi/abs/10.1126/science.aba3758.

[8] S. M. Trimberger, „Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology," *Proceedings of the IEEE*, Jg. 103, Nr. 3, S. 318–331, 2015. DOI: 10.1109/JPROC.2015.2392104.

[9] MarketsandMarkets. „FPGA Market with COVID-19 Impact Analysis by Configuration (Low-End FPGA, Mid-Range FPGA, High-End FPGA), Technology (SRAM, Flash, Antifuse), Node Size (≤16 nm, 22/28–90 nm, and >90 nm), Vertical, and Region - Global - Forecast to 2026." (2021), Adresse: https://www.marketsandmarkets.com/Market-Reports/fpga-market-194123367.html?gclid=EAIaIQobChMIw_bc76nN9QIVVI9oCR30JQ76EAAYASAAEgIWDPD_BwE (besucht am 25. 01. 2022).

[10] M. Shepovalov und V. Akella, „FPGA and GPU-based acceleration of ML workloads on Amazon cloud – A case study using gradient boosted decision tree library," *Integration*, Jg. 70, S. 1–9, 2020, ISSN: 0167-9260. DOI: https://doi.org/10.1016/j.vlsi.2019.09.007. Adresse: https://www.sciencedirect.com/science/article/pii/S0167926019300343.

[11] R. Skhiri, V. Fresse, J. P. Jamont, B. Suffran und J. Malek, „From FPGA to Support Cloud to Cloud of FPGA: State of the Art," *International Journal of Reconfigurable Computing*, Jg. 2019, 2019. DOI: 10.1155/2019/8085461. Adresse: https://doi.org/10.1155/2019/8085461.

[12] I. Kuon und J. Rose, „Measuring the Gap Between FPGAs and ASICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Jg. 26, Nr. 2, S. 203–215, 2007. DOI: 10.1109/TCAD.2006.884574.

[13] A. Greenberg, J. Hamilton, D. A. Maltz und P. Patel, „The Cost of a Cloud: Research Problems in Data Center Networks," *SIGCOMM Comput. Commun. Rev.*, Jg. 39, Nr. 1, S. 68–73, Dez. 2009, ISSN: 0146-4833. DOI: 10.1145/1496091.1496103. Adresse: https://doi.org/10.1145/1496091.1496103.

[14] S. Klingert und S. Szilvas, „Spinning gold from straw – evaluating the flexibility of data centres on power markets," *Energy Informatics*, Jg. 3, Nr. 1, Aug. 2020, ISSN: 2520-8942. DOI: 10.1186/s42162-020-00110-y. Adresse: https://doi.org/10.1186/s42162-020-00110-y.

[15] N. Jones, „How to stop data centres from gobbling up the world's electricity," *Nature*, Jg. 561, S. 163–166, Sep. 2018. DOI: 10.1038/d41586-018-06610-y. Adresse: https://www.nature.com/articles/d41586-018-06610-y.

[16] E. Gelenbe und Y. Caseau, „The Impact of Information Technology on Energy Consumption and Carbon Emissions," *Ubiquity*, Jg. 2015, Nr. June, Juni 2015. DOI: 10.1145/2755977. Adresse: https://doi.org/10.1145/2755977.

[17] Stromnetz Berlin GmbH. „Faktenblatt Stromnetz Berlin GmbH." (2021), Adresse: https://www.stromnetz.berlin/globalassets/dokumente/presse/faktenblatt-stromnetz-berlin.pdf (besucht am 27. 01. 2022).

[18] N. Pinckney, L. Shifren, B. Cline, S. Sinha, S. Jeloka, R. G. Dreslinski, T. Mudge, D. Sylvester und D. Blaauw, „Near-threshold computing in FinFET technologies: Opportunities for improved voltage scalability," in *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2016, S. 1–6. DOI: 10.1145/2897937.2898049.

[19] J. Rabaey, *Low Power Design Essentials*, 1. Aufl. Springer US, 2009, ISBN: 978-0-387-71712-8. DOI: 10.1007/978-0-387-71713-5.

[20] J. Monteiro, R. Patel und V. Tiwari, „Power Analysis and Optimization from Circuit to Register-Transfer Levels," in *Electronic Design Automation for IC Implementation, Circuit Design, and Process Technology*, L. Lavagno, I. L. Markov, G. Martin und L. K. Scheffer, Hrsg., 2. Aufl., CRC Press, by Taylor & Francis Group, LLC, 2016, S. 261–282, ISBN: 978-1-4822-5460-0.

[21] K. Sridhara, G. S. Biradar und R. Yanamshetti, „Subthreshold leakage power reduction in VLSI circuits: A survey," in *2016 International Conference on Communication and Signal Processing (ICCSP)*, 2016, S. 1120–1124. DOI: 10.1109/ICCSP.2016.7754326.

[22] P. Chaourani, I. Pappas, S. N. Nikolaidis und A. Rjoub, „Pass Transistor Operation Modeling for Nanoscale Technologies," in *Integrated Circuit and System Design. Power and Timing Modeling, Optimization, and Simulation (PATMOS)*, Springer Berlin Heidelberg, 2011, ISBN: 978-3-642-24154-3. Adresse: https://www.springerprofessional.de/pass-transistor-operation-modeling-for-nanoscale-technologies/3749356.

[23] W.-K. Chen, *The VLSI Handbook*, 2. Aufl. CRC Press, 2007, ISBN: 978-0-8493-4199-1.

[24] S. M. Nowick und M. Singh, „Asynchronous Design—Part 1: Overview and Recent Advances," *IEEE Design Test*, Jg. 32, Nr. 3, S. 5–18, 2015. DOI: 10.1109/MDAT.2015.2413759.

[25] S. M. Nowick und M. Singh, „Asynchronous Design—Part 2: Systems and Methodologies," *IEEE Design Test*, Jg. 32, Nr. 3, S. 19–28, 2015. DOI: 10.1109/MDAT.2015.2413757.

[26] J. Cortadella und S. S. Sapatnekar, „Static Timing Analysis," in *Electronic Design Automation for IC Implementation, Circuit Design, and Process Technology*, L. Lavagno, I. L. Markov, G. Martin und L. K. Scheffer, Hrsg., 2. Aufl., CRC Press, by Taylor & Francis Group, LLC, 2016, S. 133–154, ISBN: 978-1-4822-5460-0.

[27] J. Bhasker und R. Chadha, *Static Timing Analysis for Nanometer Designs: A Practical Approach*, 1st. Springer US, 2009, ISBN: 978-0-387-93819-6. DOI: 10.1007/978-0-387-93820-2.

[28] A. B. Kahng, „New game, new goal posts: A recent history of timing closure," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015, S. 1–6. DOI: 10.1145/2744769.2747937.

[29] K. Golshan, *The Art of Timing Closure, Advanced ASIC Design Implementation*, 1. Aufl. Springer, Cham, 2020, ISBN: 978-3-030-49636-4.

[30] A. B. Kahng, S. Kang, H. Lee, I. L. Markov und P. Thapar, „High-performance gate sizing with a signoff timer," in *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2013, S. 450–457. DOI: 10.1109/ICCAD.2013.6691156.

[31] V. Champac und J. G. Gervacio, *Timing Performance of Nanometer Digital Circuits Under Process Variations*, 1. Aufl. Springer International Publishing, 2018, ISBN: 978-3-319-75465-9. DOI: 10.1007/978-3-319-75465-9.

[32] S. Pidin, „Influence of Within-Die Transistor Characteristics Variation on FINFET Circuit Delay," *IEEE Transactions on Electron Devices*, Jg. 68, Nr. 7, S. 3276–3282, 2021. DOI: 10.1109/TED.2021.3082110.

[33] Z. Zhang, X. Jiang, R. Wang, S. Guo, Y. Wang und R. Huang, „Extraction of Process Variation Parameters in FinFET Technology Based on Compact Modeling and Characterization," *IEEE Transactions on Electron Devices*, Jg. 65, Nr. 3, S. 847–854, 2018. DOI: 10.1109/TED.2018.2790083.

[34] P. Jain und B. P. Das, „Within-Die Threshold Voltage Variability Estimation Using Reconfigurable Ring Oscillator," in *2017 30th International Conference on VLSI Design and 2017 16th International Conference on Embedded Systems (VLSID)*, 2017, S. 315–320. DOI: 10.1109/VLSID.2017.23.

[35] J. K. Lorenz, A. Asenov, E. Baer, S. Barraud, F. Kluepfel, C. Millar und M. Nedjalkov, „Process Variability for Devices at and beyond the 7 nm Node," *ECS Journal of Solid State Science and Technology*, Jg. 7, Nr. 11, P595–P601, 2018. DOI: 10.1149/2.0051811jss. Adresse: https://doi.org/10.1149/2.0051811jss.

[36] N. Weste und D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 4. Aufl. Pearson, 2011, ISBN: 9780321547743.

[37] T. Sakurai und A. Newton, „Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE Journal of Solid-State Circuits*, Jg. 25, Nr. 2, S. 584–594, 1990. DOI: 10.1109/4.52187.

[38] M. Anees, K. Rahul, S. A. Swarnkar und S. Yachareni, „Behaviour Shockley and Sakurai Models in 7nm FinFet," in *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, 2020, S. 1–4. DOI: 10.1109/IEMTRONICS51293.2020.9216418.

[39] D. Wolpert und P. Ampadu, „A Sensor to Detect Normal or Reverse Temperature Dependence in Nanoscale CMOS Circuits," in *2009 24th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, 2009, S. 193–201. DOI: 10.1109/DFT.2009.47.

[40] I. Keller und V. Kariat, „Noise in Digital ICs," in *Electronic Design Automation for IC Implementation, Circuit Design, and Process Technology*, L. Lavagno, I. L. Markov, G. Martin und L. K. Scheffer, Hrsg., 2. Aufl., CRC Press, by Taylor & Francis Group, LLC, 2016, S. 607–637, ISBN: 978-1-4822-5460-0.

[41] M. Cho, S. T. Kim, C. Tokunaga, C. Augustine, J. P. Kulkarni, K. Ravichandran, J. W. Tschanz, M. M. Khellah und V. De, „Postsilicon Voltage Guard-Band Reduction in a 22 nm Graphics Execution Core Using Adaptive Voltage Scaling and Dynamic Power Gating," *IEEE Journal of Solid-State Circuits*, Jg. 52, Nr. 1, S. 50–63, 2017. DOI: 10.1109/JSSC.2016.2601319.

[42] E. Karl, P. Singh, D. Blaauw und D. Sylvester, „Compact In-Situ Sensors for Monitoring Negative-Bias-Temperature-Instability Effect and Oxide Degradation," in *2008 IEEE International Solid-State Circuits Conference – Digest of Technical Papers*, 2008, S. 410–623. DOI: 10.1109/ISSCC.2008.4523231.

[43] S. Kiamehr, F. Firouzi und M. B. Tahoori, „Aging-aware timing analysis considering combined effects of NBTI and PBTI," in *International Symposium on Quality Electronic Design (ISQED)*, 2013, S. 53–59. DOI: 10.1109/ISQED.2013.6523590.

[44] G. Marti, W. H. Zisser, L. Arnaud und Y. Wouters, „Electromigration: Multiphysics model and experimental calibration," in *2016 IEEE International Reliability Physics Symposium (IRPS)*, 2016, IT-3-1-IT-3–4. DOI: 10.1109/IRPS.2016.7574616.

[45] A. Basavalingappa, J. M. Passage, M. Y. Shen und J. R. Lloyd, „Electromigration: Lognormal versus Weibull distribution," in *2017 IEEE International Integrated Reliability Workshop (IIRW)*, 2017, S. 1–4. DOI: 10.1109/IIRW.2017.8361224.

[46] H. Reisinger, O. Blank, W. Heinrigs, A. Muhlhoff, W. Gustin und C. Schlunder, „Analysis of NBTI Degradation- and Recovery-Behavior Based on Ultra Fast VT-Measurements," in *2006 IEEE International Reliability Physics Symposium Proceedings*, 2006, S. 448–453. DOI: 10.1109/RELPHY.2006.251260.

[47] M. Makabe, T. Kubota und T. Kitano, „Bias-temperature degradation of pMOSFETs: mechanism and suppression," in *2000 IEEE International Reliability Physics Symposium Proceedings. 38th Annual (Cat. No.00CH37059)*, 2000, S. 205–209. DOI: 10.1109/RELPHY.2000.843916.

[48] S. Khan und S. Hamdioui, „Temperature dependence of NBTI induced delay," in *2010 IEEE 16th International On-Line Testing Symposium*, 2010, S. 15–20. DOI: 10.1109/IOLTS.2010.5560238.

[49] D. James, „Intel Ivy Bridge unveiled — The first commercial tri-gate, high-k, metal-gate CPU," in *Proceedings of the IEEE 2012 Custom Integrated Circuits Conference*, 2012, S. 1–4. DOI: 10.1109/CICC.2012.6330644.

[50] A. Simevski, R. Kraemer und M. Krstic, „Low-complexity integrated circuit aging monitor," in *14th IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems*, 2011, S. 121–125. DOI: 10.1109/DDECS.2011.5783060.

[51] JEDEC Solid State Technology Association, *Failure Mechanisms and Models for Semiconductor Devices*, Document ID: JEP122G, 2011.

[52] S. Kiamehr, A. Amouri und M. B. Tahoori, „Investigation of NBTI and PBTI induced aging in different LUT implementations," in *2011 International Conference on Field-Programmable Technology*, 2011, S. 1–8. DOI: 10.1109/FPT.2011.6132704.

[53] J. Pachito, C. V. Martins, J. Semião, M. Santos, I. C. Teixeira und J. P. Teixeira, „The influence of clock-gating on NBTI-induced delay degradation," in *2012 IEEE 18th International On-Line Testing Symposium (IOLTS)*, 2012, S. 61–66. DOI: 10.1109/IOLTS.2012.6313842.

[54] J. H. Stathis und S. Zafar, „The negative bias temperature instability in MOS devices: A review," *Microelectronics Reliability*, Jg. 46, Nr. 2, S. 270–286, 2006, ISSN: 0026-2714. DOI: https://doi.org/10.1016/j.microrel.2005.08.001. Adresse: https://www.sciencedirect.com/science/article/pii/S0026271405003008.

[55] J. Albers, *Grundlagen integrierter Schaltungen*, 2. Aufl. Carl Hanser Verlag München, 2010, ISBN: 978-3-446-42232-2. DOI: 10.3139/9783446423688.fm. Adresse: https://www.hanser-elibrary.com/doi/abs/10.3139/9783446423688.fm.

[56] M. R. Guthaus, „Clock Design and Synthesis," in *mation for IC Implementation, Circuit Design, and Process Technology*, L. Lavagno, I. L. Markov, G. Martin und L. K. Scheffer, Hrsg., 2. Aufl., CRC Press, by Taylor & Francis Group, LLC, 2016, S. 261–282, ISBN: 978-1-4822-5460-0.

[57] M. Alioto, E. Consoli und G. Palumbo, *Flip-Flop Design in Nanometer CMOS, From High Speed to Low Energy*, 1. Aufl. Springer International, 2015, ISBN: 978-3-319-01997-0.

[58] S. Jain, L. Lin und M. Alioto, *Adaptive Digital Circuits for Power-Performance Range beyond Wide Voltage Scaling*, 1. Aufl. Springer Nature, 2020, ISBN: 978-3-030-38796-9. DOI: 10.1007/978-3-030-38796-9.

[59] E. Sicard, *Introducing 7-nm FinFET technology in Microwind*, 2017. Adresse: https://hal.archives-ouvertes.fr/hal-01558775.

[60] S. Jain, S. Khare, S. Yada, V. Ambili, P. Salihundam, S. Ramani, S. Muthukumar, M. Srinivasan, A. Kumar, S. K. Gb, R. Ramanarayanan, V. Erraguntla, J. Howard, S. Vangal, S. Dighe, G. Ruhl, P. Aseron, H. Wilson, N. Borkar, V. De und S. Borkar, „A 280mV-to-1.2V wide-operating-range IA-32 processor in 32nm CMOS," in *2012 IEEE International Solid-State Circuits Conference*, 2012, S. 66–68. DOI: 10.1109/ISSCC.2012.6176932.

[61] F. F. Khan und A. Ye, „An Evaluation on the Accuracy of the Minimum-Width Transistor Area Models in Ranking the Layout Area of FPGA Architectures," *ACM Trans. Reconfigurable Technol. Syst.*, Jg. 11, Nr. 1, Apr. 2018, ISSN: 1936-7406. DOI: 10.1145/3182394.

[62] „FPGA Architecture," Altera, Techn. Ber., 2006, White Paper. Adresse: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/wp/wp-01003.pdf.

[63] D. Lewis und J. Chromczak, „Process technology implications for FPGAs (Invited Paper)," in *2012 International Electron Devices Meeting*, 2012, S. 25.2.1–25.2.4. DOI: 10.1109/IEDM.2012.6479100.

[64] C. Chiasson und V. Betz, „Should FPGAS abandon the pass-gate?“ In *2013 23rd International Conference on Field programmable Logic and Applications*, 2013, S. 1–8. DOI: 10.1109/FPL.2013.6645511.

[65] M. B. Petersen, S. Nikolić und M. Stojilović, „NetCracker: A Peek into the Routing Architecture of Xilinx 7-Series FPGAs,“ in *The 2021 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, Ser. FPGA '21, Virtual Event, USA: Association for Computing Machinery, 2021, S. 11–22, ISBN: 9781450382182. DOI: 10.1145/3431920.3439285.

[66] S. P. Young, „Six-input multiplexer wtih two gate levels and three memory cells,“ US5744995A, US-Patent, Apr. 1998. Adresse: https://patents.google.com/patent/US5744995A/en.

[67] D. Lewis, G. Chiu, J. Chromczak, D. Galloway, B. Gamsa, V. Manohararajah, I. Milton, T. Vanderhoek und J. Van Dyken, „The Stratix™ 10 Highly Pipelined FPGA Architecture,“ in *Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, Ser. FPGA '16, Monterey, California, USA: Association for Computing Machinery, 2016, S. 159–168, ISBN: 9781450338561. DOI: 10.1145/2847263.2847267. Adresse: https://doi.org/10.1145/2847263.2847267.

[68] D. Lewis, V. Betz, D. Jefferson, A. Lee, C. Lane, P. Leventis, S. Marquardt, C. McClintock, B. Pedersen, G. Powell, S. Reddy, C. Wysocki, R. Cliff und J. Rose, „The Stratix™ Routing and Logic Architecture,“ in *Proceedings of the 2003 ACM/SIGDA Eleventh International Symposium on Field Programmable Gate Arrays*, Ser. FPGA '03, Monterey, California, USA: Association for Computing Machinery, 2003, S. 12–20, ISBN: 158113651X. DOI: 10.1145/611817.611821.

[69] C. Chiasson und V. Betz, „COFFE: Fully-automated transistor sizing for FPGAs,“ in *2013 International Conference on Field-Programmable Technology (FPT)*, 2013, S. 34–41. DOI: 10.1109/FPT.2013.6718327.

[70] Z. Li, Y. Xiao, Y. Zhang, Y. Pang, C. Hu, J. Wang und J. Lai, „An Automatic Transistor-Level Tool for GRM FPGA Interconnect Circuits Optimization,“ in *Proceedings of the 2019 on Great Lakes Symposium on VLSI*, Ser. GLSVLSI '19, Tysons Corner, VA, USA: Association for Computing Machinery, 2019, S. 93–98, ISBN: 9781450362528. DOI: 10.1145/3299874.3318003.

[71] M. Cho, S. Kim, C. Tokunaga, C. Augustine, J. Kulkarni, K. Ravichandran, J. Tschanz, M. Khellah und V. De, „Post-Silicon Voltage-Guard-Band Reduction in a 22nm Graphics Execution Core Using Adaptive Voltage Scaling and Dynamic Power Gating,“ in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, Bd. 59, 2016, S. 152–153. DOI: 10.1109/ISSCC.2016.7417952.

[72] L. Y.-Z. Lin und C. H.-P. Wen, „Speed binning with high-quality structural patterns from functional timing analysis (FTA),“ in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2016, S. 238–243. DOI: 10.1109/ASPDAC.2016.7428017.

[73] A. K. M. M. Islam, J. Shiomi, T. Ishihara und H. Onodera, „Wide-Supply-Range All-Digital Leakage Variation Sensor for On-Chip Process and Temperature Monitoring,“ *IEEE Journal of Solid-State Circuits*, Jg. 50, Nr. 11, S. 2475–2490, 2015. DOI: 10.1109/JSSC.2015.2461598.

[74] K. Maragos, G. Lentaris und D. Soudris, „In-the-Field Mitigation of Process Variability for Improved FPGA Performance," *IEEE Transactions on Computers*, Jg. 68, Nr. 7, S. 1049–1063, 2019. DOI: 10.1109/TC.2019.2898833.

[75] „Zynq UltraScale+ MPSoC Data Sheet: DC and AC Switching Characteristics," Xilinx, Techn. Ber. DS925 (v1.20), Jan. 2022. Adresse: https://www.xilinx.com/support/documentation/data_sheets/ds925-zynq-ultrascale-plus.pdf (besucht am 12. 03. 2022).

[76] K. Chapman und J. Hussein, „Lowering Power using the Voltage Identification Bit," Xilinx, Techn. Ber. XAPP555 (v1.1), Dezember 2012. Adresse: https://www.xilinx.com/support/documentation/application_notes/xapp555-Lowering-Power-Using-VID-Bit.pdf (besucht am 11. 03. 2022).

[77] *Intel® Stratix® 10 Power Management User Guide*, Technische Dokumentation, Intel, 2021. Adresse: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/stratix-10/ug-s10-pwr.pdf.

[78] J. Li und M. Seok, „Robust and In-Situ Self-Testing Technique for Monitoring Device Aging Effects in Pipeline Circuits," in *Proceedings of the 51st Annual Design Automation Conference*, Ser. DAC '14, San Francisco, CA, USA: Association for Computing Machinery, 2014, S. 1–6, ISBN: 9781450327305. DOI: 10.1145/2593069.2593205. Adresse: https://doi.org/10.1145/2593069.2593205.

[79] S. Zhao, I. Ahmed, C. Lamoureux, A. Lotfi, V. Betz und O. Trescases, „A universal self-calibrating Dynamic Voltage and Frequency Scaling (DVFS) scheme with thermal compensation for energy savings in FPGAs," in *2016 IEEE Applied Power Electronics Conference and Exposition (APEC)*, 2016, S. 1882–1887. DOI: 10.1109/APEC.2016.7468125.

[80] I. Ahmed, S. Zhao, O. Trescases und V. Betz, „Automatic Application-Specific Calibration to Enable Dynamic Voltage Scaling in FPGAs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Jg. 37, Nr. 12, S. 3095–3108, 2018. DOI: 10.1109/TCAD.2018.2801222.

[81] S. Zhao, I. Ahmed, C. Lamoureux, A. Lotfi, V. Betz und O. Trescases, „Robust Self-Calibrated Dynamic Voltage Scaling in FPGAs With Thermal and IR-Drop Compensation," *IEEE Transactions on Power Electronics*, Jg. 33, Nr. 10, S. 8500–8511, 2018. DOI: 10.1109/TPEL.2017.2775448.

[82] S. Zhao, I. Ahmed, A. Khakpour, V. Betz und O. Trescases, „A robust dynamic voltage scaling scheme for FPGAs with IR drop compensation," in *2017 IEEE Applied Power Electronics Conference and Exposition (APEC)*, 2017, S. 2939–2944. DOI: 10.1109/APEC.2017.7931114.

[83] C. T. Chow, L. S. M. Tsui, P. H. W. Leong, W. Luk und S. J. E. Wilton, „Dynamic Voltage Scaling for Commercial FPGAs," in *IEEE International Conference on Field-Programmable Technology*, 2005, S. 173–180. DOI: 10.1109/FPT.2005.1568543.

[84] M. Naouss und F. Marc, „Modelling delay degradation due to NBTI in FPGA Look-up tables," in *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, 2016, S. 1–4. DOI: 10.1109/FPL.2016.7577328.

[85] J. L. Nunez-Yanez, V. Chouliaras und J. Gaisler, „Dynamic Voltage Scaling in a FPGA-Based System-on-Chip," in *2007 International Conference on Field Programmable Logic and Applications*, 2007, S. 459–462. DOI: 10.1109/FPL.2007.4380689.

[86] K. Maragos, G. Lentaris und D. Soudris, „A PVT-Aware Voltage Scaling Method for Energy Efficient FPGAs," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2021, S. 1–5. DOI: 10.1109/ISCAS51556.2021.9401622.

[87] M. Agarwal, V. Balakrishnan, A. Bhuyan, K. Kim, B. C. Paul, W. Wang, B. Yang, Y. Cao und S. Mitra, „Optimized Circuit Failure Prediction for Aging: Practicality and Promise," in *2008 IEEE International Test Conference*, 2008, S. 1–10. DOI: 10.1109/TEST.2008.4700619.

[88] D. Ernst, Nam Sung Kim, S. Das, S. Pant, R. Rao, Toan Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner und T. Mudge, „Razor: a low-power pipeline based on circuit-level timing speculation," in *Proceedings. 36th Annual IEEE/ACM International Symposium on Microarchitecture, 2003. MICRO-36.*, 2003, S. 7–18. DOI: 10.1109/MICRO.2003.1253179.

[89] K. A. Bowman und J. W. Tschanz, „Resilient Circuits for Dynamic Variation Tolerance," in *Emerging Technologies and Circuits*, A. Amara, T. Ea und M. Belleville, Hrsg., 1. Aufl., Springer, Dordrecht, 2010, S. 141–162, ISBN: 978-90-481-9379-0.

[90] S. Das, C. Tokunaga, S. Pant, W.-H. Ma, S. Kalaiselvan, K. Lai, D. M. Bull und D. T. Blaauw, „RazorII: In Situ Error Detection and Correction for PVT and SER Tolerance," *IEEE Journal of Solid-State Circuits*, Jg. 44, Nr. 1, S. 32–48, 2009. DOI: 10.1109/JSSC.2008.2007145.

[91] R. Ragavan, C. Killian und O. Sentieys, „Adaptive Overclocking and Error Correction Based on Dynamic Speculation Window," in *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2016, S. 325–330. DOI: 10.1109/ISVLSI.2016.13.

[92] K. Minkovich und J. Cong, „Mapping for Better than Worst-Case Delays in LUT-Based FPGA Designs," in *Proceedings of the 16th International ACM/SIGDA Symposium on Field Programmable Gate Arrays*, Ser. FPGA '08, Monterey, California, USA: Association for Computing Machinery, 2008, S. 56–64, ISBN: 9781595939340. DOI: 10.1145/1344671.1344681. Adresse: https://doi.org/10.1145/1344671.1344681.

[93] A. Brant, A. Abdelhadi, D. H. Sim, S. L. Tang, M. X. Yue und G. G. Lemieux, „Safe Overclocking of Tightly Coupled CGRAs and Processor Arrays using Razor," in *2013 IEEE 21st Annual International Symposium on Field-Programmable Custom Computing Machines*, 2013, S. 37–44. DOI: 10.1109/FCCM.2013.63.

[94] J. L. Nunez-Yanez, „Adaptive Voltage Scaling with In-Situ Detectors in Commercial FPGAs," *IEEE Transactions on Computers*, Jg. 64, Nr. 1, S. 45–53, 2015. DOI: 10.1109/TC.2014.2365963.

[95] J. Luis Nunez-Yanez, M. Hosseinabady und A. Beldachi, „Energy Optimization in Commercial FPGAs with Voltage, Frequency and Logic Scaling," *IEEE Transactions on Computers*, Jg. 65, Nr. 5, S. 1484–1493, 2016. DOI: 10.1109/TC.2015.2435771.

[96] J. Nunez-Yanez, „Adaptive voltage scaling in a heterogeneous FPGA device with memory and logic in-situ detectors," *Microprocessors and Microsystems*, Jg. 51, S. 227–238, 2017, ISSN: 0141-9331. DOI: https://doi.org/10.1016/j.micpro.

2017.04.021. Adresse: https://www.sciencedirect.com/science/article/pii/S0141933117302223.

[97] J. Nunez-Yanez, „Energy Proportional Neural Network Inference with Adaptive Voltage and Frequency Scaling," *IEEE Transactions on Computers*, Jg. 68, Nr. 5, S. 676–687, 2019. DOI: 10.1109/TC.2018.2879333.

[98] J. M. Levine, E. Stott, G. A. Constantinides und P. Y. Cheung, „Online Measurement of Timing in Circuits: For Health Monitoring and Dynamic Voltage amp; Frequency Scaling," in *2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*, 2012, S. 109–116. DOI: 10.1109/FCCM.2012.27.

[99] J. M. Levine, E. Stott, G. A. Constantinides und P. Y. K. Cheung, „SMI: Slack Measurement Insertion for online timing monitoring in FPGAs," in *2013 23rd International Conference on Field programmable Logic and Applications*, 2013, S. 1–4. DOI: 10.1109/FPL.2013.6645598.

[100] J. M. Levine, E. Stott und P. Y. Cheung, „Dynamic Voltage & Frequency Scaling with Online Slack Measurement," in *Proceedings of the 2014 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, Ser. FPGA '14, Monterey, California, USA: Association for Computing Machinery, 2014, S. 65–74, ISBN: 9781450326711. DOI: 10.1145/2554688.2554784. Adresse: https://doi.org/10.1145/2554688.2554784.

[101] H. Giesen, B. Gojman, R. Rubin, J. Kim und A. Dehon, „Continuous Online Self-Monitoring Introspection Circuitry for Timing Repair by Incremental Partial-Reconfiguration (COSMIC TRIP)," *ACM Transactions on Reconfigurable Technology and Systems*, Jg. 11, Nr. 1, Jan. 2018, ISSN: 1936-7406. DOI: 10.1145/3158229. Adresse: https://doi.org/10.1145/3158229.

[102] A. M. Keller und M. J. Wirthlin, „Impact of Soft Errors on Large-Scale FPGA Cloud Computing," in *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, Ser. FPGA '19, Seaside, CA, USA: Association for Computing Machinery, 2019, S. 272–281, ISBN: 9781450361378. DOI: 10.1145/3289602.3293911.

[103] M. Agarwal, B. C. Paul, M. Zhang und S. Mitra, „Circuit Failure Prediction and Its Application to Transistor Aging," in *25th IEEE VLSI Test Symposium (VTS'07)*, 2007, S. 277–286. DOI: 10.1109/VTS.2007.22.

[104] M. D. Valdes-Peña, J. Fernández Freijedo, M. J. Moure Rodríguez, J. J. Rodríguez-Andina, J. Semião, I. M. C. Teixeira, J. P. C. Teixeira und F. Vargas, „Design and Validation of Configurable Online Aging Sensors in Nanometer-Scale FPGAs," *IEEE Transactions on Nanotechnology*, Jg. 12, Nr. 4, S. 508–517, 2013. DOI: 10.1109/TNANO.2013.2253795.

[105] Z. Ghaderi, M. Ebrahimi, Z. Navabi, E. Bozorgzadeh und N. Bagherzadeh, „SENSIBle: A Highly Scalable SENsor DeSIgn for Path-Based Age Monitoring in FPGAs," *IEEE Transactions on Computers*, Jg. 66, Nr. 5, S. 919–926, 2017. DOI: 10.1109/TC.2016.2622688.

[106] A. Grenat, S. Sundaram, S. Kosonocky, R. Rachala, S. Sambamurthy, S. Liepe, M. Rodriguez, T. Burd, A. Clark, M. Austin und S. Naffziger, „Increasing the Performance of a 28nm x86-64 Microprocessor Through System Power Management," in *IEEE Inter-*

national Solid-State Circuits Conference (ISSCC), 2016, S. 74–75. DOI: 10.1109/ISSCC.2016.7417913.

[107] S. Feng, S. Gupta, A. Ansari und S. Mahlke, „Maestro: Orchestrating Lifetime Reliability in Chip Multiprocessors," in *High Performance Embedded Architectures and Compilers*, Y. N. Patt, P. Foglia, E. Duesterwald, P. Faraboschi und X. Martorell, Hrsg., Springer Berlin Heidelberg, 2010, S. 186–200, ISBN: 978-3-642-11515-8. DOI: 10.1007/978-3-642-11515-8_15.

[108] S. Han, J. Choung, B.-S. Kim, B. H. Lee, H. Choi und J. Kim, „Statistical aging analysis with process variation consideration," in *2011 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2011, S. 412–419. DOI: 10.1109/ICCAD.2011.6105362.

[109] B. Salami, O. S. Unsal und A. Cristal Kestelman, „Comprehensive Evaluation of Supply Voltage Underscaling in FPGA on-Chip Memories," in *51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2018, S. 724–736. DOI: 10.1109/MICRO.2018.00064.

[110] I. Stratakos, K. Maragos und G. Lentaris, „Voltage Scaling and Guardband Customization of Multiple Constituent Components in SoC-FPGA," in *IEEE 29th International Symposium on Power and Timing Modeling, Optimization and Simulation, PATMOS 2019*, 2019, S. 75–80, ISBN: 9781728121031. DOI: 10.1109/PATMOS.2019.8862050.

[111] A. F. Gomez und V. Champac, „Critical path selection under NBTI/PBTI aging for adaptive frequency tuning," in *2016 IEEE East-West Design Test Symposium (EWDTS)*, 2016, S. 1–4. DOI: 10.1109/EWDTS.2016.7807652.

[112] S. Kostin, J. Raik, R. Ubar, M. Jenihhin, F. Vargas, L. M. B. Poehls und T. S. Copetti, „Hierarchical identification of NBTI-critical gates in nanoscale logic," in *2014 15th Latin American Test Workshop – LATW*, 2014, S. 1–6. DOI: 10.1109/LATW.2014.6841926.

[113] M. Ebrahimi und Z. Navabi, „Selecting Representative Critical Paths for Sensor Placement Provides Early FPGA Aging Information," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Jg. 39, Nr. 10, S. 2976–2989, 2020. DOI: 10.1109/TCAD.2019.2953174.

[114] M. Ebrahimi, Z. Ghaderi, E. Bozorgzadeh und Z. Navabi, „Path selection and sensor insertion flow for age monitoring in FPGAs," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2016, S. 792–797, ISBN: 978-3-9815-3707-9.

[115] P. Sedcole und P. Y. K. Cheung, „Within-die delay variability in 90nm FPGAs and beyond," in *2006 IEEE International Conference on Field Programmable Technology*, 2006, S. 97–104. DOI: 10.1109/FPT.2006.270300.

[116] K. M. Zick und J. P. Hayes, „On-Line Sensing for Healthier FPGA Systems," in *Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, Ser. FPGA '10, Monterey, California, USA: Association for Computing Machinery, 2010, S. 239–248, ISBN: 9781605589114. DOI: 10.1145/1723112.1723153.

[117] K. Maragos, G. Lentaris, D. Soudris, K. Siozios und V. F. Pavlidis, „Application performance improvement by exploiting process variability on FPGA devices," in *Design,*

Automation Test in Europe Conference Exhibition (DATE), 2017, 2017, S. 452–457. DOI: 10.23919/DATE.2017.7927032.

[118] T. Tuan, A. Lesea, C. Kingsley und S. Trimberger, „Analysis of within-die process variation in 65nm FPGAs," in *2011 12th International Symposium on Quality Electronic Design*, 2011, S. 1–5. DOI: 10.1109/ISQED.2011.5770808.

[119] M. Gag, T. Wegner, A. Waschki und D. Timmermann, „Temperature and On-chip Crosstalk Measurement Using Ring Oscillators in FPGA," in *Proceedings of the 2012 IEEE 15th International Symposium on Design and Diagnostics of Electronic Circuits and Systems, DDECS 2012*, 2012, S. 201–204, ISBN: 9781467311854. DOI: 10.1109/DDECS.2012.6219057.

[120] T. Wegner, *Modellierung und Steuerung der Temperaturverteilung Network-on-Chip-basierter Systeme*. 2014, Universität Rostock, Fak. für Informatik und Elektrotechnik, Dissertation, 2014. Adresse: https://doi.org/10.18453/rosdok_id00001371.

[121] L. Y. Foo, „Power Delivery Network Step Load Response and 1st Droop Formulation," in *2019 Electrical Design of Advanced Packaging and Systems (EDAPS)*, 2019, S. 1–6. DOI: 10.1109/EDAPS47854.2019.9011667.

[122] Texas Instruments, *Digital Point of Load System Controller datasheet (Rev. C)*, 2008. Adresse: http://www.ti.com/lit/ds/symlink/ucd9240.pdf.

[123] H. J. Zhang, „Basic Concepts of Linear Regulator and Switching Mode Power Supplies," Analog Devices, Techn. Ber. AN140, Oktober 2013. Adresse: https://www.analog.com/media/en/technical-documentation/app-notes/an140.pdf.

[124] B. Zimmer, Y. Lee, A. Puggelli, J. Kwak, R. Jevti, B. Keller, S. Bailey, M. Blagojevi, P.-F. Chiu, H.-p. Le, P.-h. Chen, N. Sutardja, R. Avizienis, A. Waterman, B. Richards, P. Flatresse, E. Alon, K. Asanovi und B. Nikolić, „A RISC-V Vector Processor With DC - DC Converters in 28 nm FDSOI," *IEEE Journal of Solid-State Circuits*, Jg. 51, Nr. 4, S. 930–942, 2016. DOI: 10.1109/JSSC.2016.2519386.

[125] D. R. E. Gnad, F. Oboril, S. Kiamehr und M. B. Tahoori, „An Experimental Evaluation and Analysis of Transient Voltage Fluctuations in FPGAs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Jg. 26, Nr. 10, S. 1817–1830, 2018. DOI: 10.1109/TVLSI.2018.2848460.

[126] J. Haj-Yihia, Y. B. Asher, E. Rotem, A. Yasin und R. Ginosar, „Compiler-Directed Power Management for Superscalars," *ACM Trans. Archit. Code Optim.*, Jg. 11, Nr. 4, Jan. 2015, ISSN: 1544-3566. DOI: 10.1145/2685393. Adresse: https://doi.org/10.1145/2685393.

[127] A. Aloisio, P. Branchini, R. Cicalese, R. Giordano, V. Izzo und S. Loffredo, „FPGA Implementation of a High-Resolution Time-to-Digital Converter," in *IEEE Nuclear Science Symposium Conference Record*, 2007, S. 504–507. DOI: 10.1109/NSSMIC.2007.4436379.

[128] K. M. Zick, M. Srivastav, W. Zhang und M. French, „Sensing Nanosecond-Scale Voltage Attacks and Natural Transients in FPGAs," in *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, Ser. FPGA '13, Monterey, California, USA: Association for Computing Machinery, 2013, S. 101–104, ISBN: 9781450318877. DOI: 10.1145/2435264.2435283.

[129] D. R. E. Gnad, F. Oboril, S. Kiamehr und M. B. Tahoori, „Analysis of Transient voltage Fluctuations in FPGAs," in *International Conference on Field-Programmable Technology (FPT)*, 2016, S. 12–19. DOI: 10.1109/FPT.2016.7929182.

[130] Xilinx, *Virtex-6 FPGA Data Sheet: DC and Switching Characteristics*, 2014. Adresse: https://www.xilinx.com/support/documentation/data%7B%5C_%7Dsheets/ds152.pdf.

[131] A. Amouri, J. Hepp und M. Tahoori, „Built-In Self-Heating Thermal Testing of FPGAs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Jg. 35, Nr. 9, S. 1546–1556, 2016. DOI: 10.1109/TCAD.2015.2512905.

[132] S. H. C. Hsieh und S. Chan, „Methods and circuits for measuring the thermal resistance of a packaged IC," 7 257 511, US-Patent, Aug. 2007. Adresse: https://patents.justia.com/patent/7257511.

[133] P. Weber, M. Zagrabski, P. Musz, K. Kępa, M. Nikodem und B. Wojciechowski, „Configurable heat generators for FPGAs," in *20th International Workshop on Thermal Investigations of ICs and Systems*, 2014, S. 1–4. DOI: 10.1109/THERMINIC.2014.6972506.

[134] A. Agne, H. Hangmann, M. Happe, M. Platzner und C. Plessl, „Seven Recipes for Setting Your FPGA on Fire – A Cookbook on Heat Generators," *Microprocessors and Microsystems*, Jg. 38, Nr. 8, S. 911–919, 2014. DOI: 10.1016/j.micpro.2013.12.001.

[135] M. Happe, H. Hangmann, A. Agne und C. Plessl, „Eight ways to put your FPGA on fire — A systematic study of heat generators," in *2012 International Conference on Reconfigurable Computing and FPGAs*, 2012, S. 1–6. DOI: 10.1109/ReConFig.2012.6416745.

[136] D. R. E. Gnad, F. Oboril und M. B. Tahoori, „Voltage drop-based fault attacks on FPGAs using valid bitstreams," in *2017 27th International Conference on Field Programmable Logic and Applications (FPL)*, 2017, S. 1–7. DOI: 10.23919/FPL.2017.8056840.

[137] J. Krautter, D. R. E. Gnad und M. B. Tahoori, „FPGAhammer: Remote Voltage Fault Attacks on Shared FPGAs, suitable for DFA on AES," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, Jg. 2018, Nr. 3, S. 44–68, Aug. 2018. DOI: 10.13154/tches.v2018.i3.44-68. Adresse: https://tches.iacr.org/index.php/TCHES/article/view/7268.

[138] G. Provelengios, D. Holcomb und R. Tessier, „Mitigating Voltage Attacks in Multi-Tenant FPGAs," *ACM Trans. Reconfigurable Technol. Syst.*, Jg. 14, Nr. 2, Juli 2021, ISSN: 1936-7406. DOI: 10.1145/3451236. Adresse: https://doi.org/10.1145/3451236.

[139] S. Chan und S. H. C. Hsieh, „Methods and apparatus for isolating critical paths on an IC device having a thermal energy generator," 6 895 566, US-Patent, Mai 2015. Adresse: https://patents.justia.com/patent/6895566.

[140] K. Matas, T. M. La, K. D. Pham und D. Koch, „Power-hammering through Glitch Amplification – Attacks and Mitigation," in *2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, 2020, S. 65–69. DOI: 10.1109/FCCM48280.2020.00018.

[141] G. Provelengios, D. Holcomb und R. Tessier, „Power Wasting Circuits for Cloud FPGA Attacks," in *2020 30th International Conference on Field-Programmable Logic and Applications (FPL)*, 2020, S. 231–235. DOI: 10.1109/FPL50879.2020.00046.

[142] „ML605 Hardware User Guide," Xilinx, Techn. Ber. UG534, 2019. Adresse: https://docs.xilinx.com/v/u/en-US/ug534 (besucht am 13. 07. 2022).

[143] S. Golson und L. Clark, „Language wars in the 21st century: verilog versus vhdl–revisited," *Synopsys Users Group (SNUG)*, 2016. Adresse: https://www.synopsys.com/community/snug/snug-silicon-valley/location-proceedings-2016.html.

[144] J. Reichardt und B. Schwarz, *VHDL-Synthese, Entwurf digitaler Schaltungen und Systeme*, 6. Aufl. De Gruyter Oldenbourg, 2013, ISBN: 9783486716788.

[145] D. Timmermann, *Selected Topics of VLSI Design*, Vorlesung an der Universität Rostock, 2022.

[146] W. Nogami, T. Ikegami, S.-i. O'uchi, R. Takano und T. Kudoh, „Optimizing Weight Value Quantization for CNN Inference," in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019, S. 1–8. DOI: 10.1109/IJCNN.2019.8852331.

[147] P. Gysel, J. Pimentel, M. Motamedi und S. Ghiasi, „Ristretto: A Framework for Empirical Study of Resource-Efficient Inference in Convolutional Neural Networks," *IEEE Transactions on Neural Networks and Learning Systems*, Jg. 29, Nr. 11, S. 5784–5789, 2018. DOI: 10.1109/TNNLS.2018.2808319.

[148] B. S. Prabakaran, V. Mrazek, Z. Vasicek, L. Sekanina und M. Shafique, „ApproxFPGAs: Embracing ASIC-Based Approximate Arithmetic Components for FPGA-Based Systems," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, S. 1–6. DOI: 10.1109/DAC18072.2020.9218533.

[149] M. Spering und T. Schmidt, *Allgemeine Psychologie kompakt : Wahrnehmung, Aufmerksamkeit, Denken, Sprache; mit Add-on*, 1. Aufl. Weinheim [u.a.]: Beltz, 2009, ISBN: 978-3-621-27868-3.

[150] C. Liu, J. Han und F. Lombardi, „A Low-Power, High-Performance Approximate Multiplier with Configurable Partial Error Recovery," in *2014 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2014, S. 1–4. DOI: 10.7873/DATE.2014.108.

[151] V. Mrazek, R. Hrbacek, Z. Vasicek und L. Sekanina, „EvoApprox8b: Library of Approximate Adders and Multipliers for Circuit Design and Benchmarking of Approximation Methods," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, 2017, S. 258–261, ISBN: 9783981537093. DOI: 10.23919/DATE.2017.7926993.

[152] J. Mody, R. Lawand, R. Priyanka, S. Sivanantham und K. Sivasankaran, „Study of Approximate Compressors for Multiplication Using FPGA," 2015. DOI: 10.1109/GET.2015.7453816.

[153] N. V. Toan und J. Lee, „Energy-Area-Efficient Approximate Multipliers for Error-Tolerant Applications on FPGAs," in *2019 32nd IEEE International System-on-Chip Conference (SOCC)*, 2019, S. 336–341. DOI: 10.1109/SOCC46988.2019.1570548202.

[154] M. Osta, A. Ibrahim, H. Chible und M. Valle, „Approximate Multipliers Based on Inexact Adders for Energy Efficient Data Processing," in *2017 New Generation of CAS (NGCAS)*, 2017, S. 125–128, ISBN: 9781509064472. DOI: 10.1109/NGCAS.2017.41.

[155] S. Ullah, S. Rehman, B. S. Prabakaran, F. Kriebel, M. A. Hanif, M. Shafique und A. Kumar, „Area-Optimized Low-Latency Approximate Multipliers for FPGA-Based Hardware Accelerators," in *Proceedings of the 55th Annual Design Automation Conference*, Ser. DAC '18, San Francisco, California: Association for Computing Machinery, 2018. DOI: 10.1145/3195970.3195996.

[156] Y. Guo, H. Sun und S. Kimura, „Small-Area and Low-Power FPGA-Based Multipliers using Approximate Elementary Modules," in *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2020, S. 599–604. DOI: 10.1109/ASP-DAC47756.2020.9045546.

[157] Xilinx, *7 Series FPGAs Configurable Logic Block – User Guide*, 2016. Adresse: https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf.

[158] Z. Yang, J. Han und F. Lombardi, „Approximate Compressors for Error-Resilient Multiplier Design," in *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*, 2015, S. 183–186. DOI: 10.1109/DFT.2015.7315159.

[159] „Vivado Design Suite User Guide – Getting Started," Xilinx, Techn. Ber. UG910 (v2022.1), 2022. Adresse: https://www.xilinx.com/content/dam/xilinx/support/documents/sw_manuals/xilinx2022_1/ug910-vivado-getting-started.pdf (besucht am 28. 08. 2022).

[160] „The Questa Verification Solution," Siemens, Techn. Ber., 2021. Adresse: https://static.sw.cdn.siemens.com/siemens-disw-assets/public/2VxKS6Q3bOyJlBmp3NzVpm/en-US/The-Questa-Verification-Solution-FS.pdf (besucht am 28. 08. 2022).

[161] J. Hussein, M. Klein und M. Hart, „Lowering Power at 28 nm with Xilinx 7 Series Devices," Xilinx, Techn. Ber. WP389, Jan. 2015, White Paper. Adresse: https://www.xilinx.com/support/documentation/white_papers/wp389_Lowering_Power_at_28nm.pdf (besucht am 09. 02. 2022).

[162] *Vivado Design Suite User Guide: Power Analysis and Optimization*, 2020. Adresse: https://www.xilinx.com/content/dam/xilinx/support/documentation/sw_manuals/xilinx2021_1/ug907-vivado-power-analysis-optimization.pdf (besucht am 11. 02. 2022).

[163] Y. Lin, Y. Li, T. Liu, T. Xiao, T. Liu und J. Zhu, „Towards Fully 8-Bit Integer Inference for the Transformer Model," Ser. IJCAI'20, Yokohama, Yokohama, Japan, 2021, ISBN: 9780999241165.

[164] Z. Li, L. Wang, S. Guo, Y. Deng, Q. Dou, H. Zhou und W. Lu, „Laius: An 8-Bit Fixed-Point CNN Hardware Inference Engine," in *2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)*, 2017, S. 143–150. DOI: 10.1109/ISPA/IUCC.2017.00030.

[165] J. Gong, H. Shen, G. Zhang, X. Liu, S. Li, G. Jin, N. Maheshwari, E. Fomenko und E. Segal, „Highly Efficient 8-Bit Low Precision Inference of Convolutional Neural Networks with IntelCaffe," Ser. ReQuEST '18, Williamsburg, VA, USA: Association for Computing Machinery, 2018, ISBN: 9781450359238. DOI: 10.1145/3229762.3229763. Adresse: https://doi.org/10.1145/3229762.3229763.

[166] M. Ha und S. Lee, „Multipliers With Approximate 4–2 Compressors and Error Recovery Modules,“ *IEEE Embedded Systems Letters*, Jg. 10, Nr. 1, S. 6–9, 2018. DOI: 10.1109/LES.2017.2746084.

[167] M. S. Ansari, H. Jiang, B. F. Cockburn und J. Han, „Low-Power Approximate Multipliers Using Encoded Partial Products and Approximate Compressors,“ *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, Jg. 8, Nr. 3, S. 404–416, 2018. DOI: 10.1109/JETCAS.2018.2832204.

[168] T. Yang, T. Ukezono und T. Sato, „Low-Power and High-Speed Approximate Multiplier Design with a Tree Compressor,“ in *2017 IEEE International Conference on Computer Design (ICCD)*, 2017, S. 89–96. DOI: 10.1109/ICCD.2017.22.

B Liste der Veröffentlichungen (peer-reviewed)

[1] C. Niemann, M. Ali, J. Heller und D. Timmermann, „A Calibration Procedure for Sensor Based Adaptive Voltage Scaling Approaches," in *2019 29th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 2019, S. 81–86. DOI: 10.1109/patmos.2019.8862075.

[2] C. Niemann, M. Ali, O. U. Shah, J. Heller und D. Timmermann, „Sensor based adaptive voltage scaling on FPGAs: Calibration and parametrization," *Integration – the VLSI Journal*, Jg. 75, S. 30–39, 2020, ISSN: 0167-9260. DOI: https://doi.org/10.1016/j.vlsi.2020.05.006.

[3] C. Niemann, T. Wegner, D. Timmermann und F. S. Torres, „Low overhead in situ aging monitoring and proactive aging management," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, S. 2799–2802. DOI: 10.1109/ISCAS.2016.7539174.

[4] C. Niemann, M. Rethfeldt und D. Timmermann, „Approximate Multipliers for Optimal Utilization of FPGA Resources," in *2021 24th International Symposium on Design and Diagnostics of Electronic Circuits Systems (DDECS)*, 2021, S. 23–28. DOI: 10.1109/DDECS52668.2021.9417027.

[5] E. Schweissguth, P. Danielis, C. Niemann und D. Timmermann, „Application-aware industrial ethernet based on an SDN-supported TDMA approach," in *2016 IEEE World Conference on Factory Communication Systems (WFCS)*, 2016, S. 1–8. DOI: 10.1109/WFCS.2016.7496496.

[6] H. Puttnies, C. Niemann, S. Rohde, D. Timmermann und J. Schacht, „Towards software performance estimation based on register-transfer level descriptions," in *2017 IEEE Nordic Circuits and Systems Conference (NORCAS): NORCHIP and International Symposium of System-on-Chip (SoC)*, 2017, S. 1–6. DOI: 10.1109/NORCHIP.2017.8124967.

[7] M. Zwar, D. Franz, M. Rohde, V. Neubert, M. Paap, S. Perl, C. Niemann, F. Plocksties, D. Timmermann, A. Richter und R. Köhling, „Modulation of cortico-striatal GABAergic inhibition by short-term deep brain stimulation in a phenotypic animal model of dystonia," in *98th Meeting of the German Physiological Society*, 2019. DOI: 10.1111/apha.13385.

[8] M. Paap, S. Perl, A. Lüttig, F. Plocksties, C. Niemann, D. Timmermann, C. Bahls, U. van Rienen, D. Franz, M. Zwar, M. Rohde, R. Köhling und A. Richter, „Deep brain stimulation as a therapeutic option in dystonia: effects of different frequencies in a phenotypic animal model by using an optimized stimulator," *Journal Naunyn-Schmiedeberg's Archives of Pharmacology, Abstracts of the 86th Annual Meeting of the German Society for Experimental and Clinical Pharmacology and Toxicology (DGPT)*, Jg. 393, S. 72, Feb. 2020. DOI: 10.1007/s00210-020-01828-y.

[9] C. Niemann, C. Ewert, H. Puttnies, M. Rethfeldt, D. Timmermann und P. Danielis, „Modeling Energy Consumption for Task-Offloading Decisions on Mobile and Embedded Devices," in *2020 IEEE 2nd Global Conference on Life Sciences and Technologies (LifeTech)*, 2020, S. 400–404. DOI: 10.1109/LifeTech48969.2020.1570618809.

[10] F. Plocksties, C. Niemann, R. Bader, F. Möws, H. Seitz, P. W. Kämmerer, M. Dau, V. Kamp, J. Heller und D. Timmermann, „Requirements and Design of an Implant for

Electrical Stimulation for Bone Regeneration in Animals," in *2020 IEEE 2nd Global Conference on Life Sciences and Technologies (LifeTech)*, 2020, S. 157–160. DOI: 10.1109/LifeTech48969.2020.1570619159.

[11] J. Heller, C. Niemann, F. Plocksties, C. Haubelt und D. Timmermann, „Towards Virtual Prototyping of Electrically Active Implants Using SystemC-AMS," in *Workshop Methods and Description Languages for Modelling and Verification of Circuits and Systems (MBMV)*, 2020, ISBN: 978-3-8007-5221-8.

[12] M. Paap, S. Perl, A. Lüttig, F. Plocksties, C. Niemann, D. Timmermann, C. Bahls, U. van Rienen, D. Franz, M. Zwar, M. Rohde, R. Köhling und A. Richter, „Deep brain stimulation by optimized stimulators in a phenotypic model of dystonia: Effects of different frequencies," *Neurobiology of Disease*, Jg. 147, S. 105 163, 2021, ISSN: 0969-9961. DOI: https://doi.org/10.1016/j.nbd.2020.105163.

[13] M. Heerdegen, M. Zwar, D. Franz, J. Hörnschemeyer, V. Neubert, F. Plocksties, C. Niemann, D. Timmermann, C. Bahls, U. van Rienen, M. Paap, S. Perl, A. Lüttig, A. Richter und R. Köhling, „Mechanisms of pallidal deep brain stimulation: Alteration of corticostriatal synaptic communication in a dystonia animal model," *Neurobiology of Disease*, Jg. 154, S. 105 341, 2021, ISSN: 0969-9961. DOI: 10.1016/j.nbd.2021.105341. Adresse: https://www.sciencedirect.com/science/article/pii/S0969996121000905.

[14] F. Plocksties, M. Kober, C. Niemann, J. Heller, M. Fauser, M. Nüssel, F. Uster, D. Franz, M. Zwar, A. Lüttig, J. Kröger, J. Harloff, A. Schulz, A. Richter, R. Köhling, D. Timmermann und A. Storch, „The software defined implantable modular platform (STELLA) for preclinical deep brain stimulation research in rodents," *Journal of Neural Engineering*, Jg. 18, Nr. 5, S. 056 032, Sep. 2021. DOI: 10.1088/1741-2552/ac23e1.

C Liste weiterer Veröffentlichungen und Konferenzbeiträge

[1] C. Niemann, F. Plocksties, J. Heller und D. Timmermann, „Towards an energy autonomous platform for electrically active implants," in *Invited Session on 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, 2019.

[2] C. Niemann, J. Heller, F. Plocksties, E. Janchivnyambuu und D. Timmermann, „Reliability Enhancement of Signal Processing Hardware Accelerators for Closed Loop Deep Brain Stimulation," in *Workshop des Arbeitskreises „Zuverlässigkeit von Implantaten und Biostrukturen" des Deutschen Verbandes für Materialforschung und -prüfung (DVM)*, DVM, 2019.

[3] V. Neubert, M. Zwar, D. Franz, M. Heerdegen, A. Lüttig, S. Perl, J. Heller, C. Niemann, F. Plockstieß, F. Krüger, S. Spors, D. Timmermann, A. Richter und R. Köhling, „Identification of Biomarkers for Progression and Amelioration of Disease in a Hamster Model of Dystonia," in *Progress in Electrically Active Implants - Tissue and Functional Regeneration*, Rostock, Deutschland, Sep. 2020.

[4] M. Paap, S. Perl, A. Lüttig, F. Plockstieß, C. Niemann, D. Timmermann, C. Bahls, U. van Rienen, D. Franz, M. Zwar, M. Rhode, R. Köhling und A. Richter, „Deep brain stimulation as a therapeutic option in dystonia: effects of different frequencies in a phenotypic animal model by using an optimized stimulator," in *Progress in Electrically Active Implants - Tissue and Functional Regeneration*, Rostock, Deutschland, Sep. 2020.

[5] J. Heller, P. Wilsdorf, C. Niemann, F. Plocksties, A. M. Uhrmacher, C. Haubelt und D. Timmermann, „Assisting Early Design Decision For Implantable Neurostimulators Through Virtual Prototyping," in *Progress in Electrically Active Implants - Tissue and Functional Regeneration*, Rostock, Deutschland, Sep. 2020.

[6] M. Zwar, M. Heerdegen, D. Franz, V. Neubert, F. Plocksties, C. Niemann, D. Timmermann, C. Bahls, U. van Rienen, M. Paap, S. Perl, A. Lüttig, A. Richter und R. Köhling, „Increase of striatal inhibitory tone by pallidal deep brain stimulation in awake dystonic hamsters," in *Progress in Electrically Active Implants - Tissue and Functional Regeneration*, Rostock, Deutschland, Sep. 2020.

[7] F. Plocksties, C. Niemann, J. Heller, C. Haubelt und D. Timmermann, „Towards an Energy Autonomous Implant for Closed-loop Neurostimulation," in *Progress in Electrically Active Implants - Tissue and Functional Regeneration*, Rostock, Deutschland, Sep. 2020.

D Liste der betreuten studentischen Arbeiten

[1] H. Amjad, „Testing Different Online Sensors and Development of a User Interface based on C# for an Adaptive voltage scaling project," Specialization Module, Universität Rostock, 2018.

[2] M. Ali, „Adoption of an Adaptive Voltage Scaling Algorithm for a More Complex IP Core," Specialization Module, Universität Rostock, 2018.

[3] M. Ali, „Development of Electronic Design Automation (EDA) flow for the Application of Adaptive Voltage Scaling in FPGAs," Masterarbeit, Universität Rostock, 2019.

[4] O. U. Shah, „Extension of an Adaptive Voltage Scaling Algorithm," Specialization Module, Universität Rostock, 2018.

[5] O. U. Shah, „Development and Verification of an Adaptive Voltage Scaling Control Algorithm," Masterarbeit, Universität Rostock, 2019.

[6] S. Rohde, „Betrachtung der Hardware-Software-Partitionierung im Rahmen der Entwicklung eines Hardware-Software-Co-Designs," Masterarbeit, Universität Rostock, 2016.

[7] C. Ewert, „Modellierung des Energieverbrauchs für Mobile Cloud Computing," Bachelorarbeit, Universität Rostock, 2018.

[8] S. M. A. Hashmi, „Evaluation of a MICS-Band Transceiver and Concept and Prototypical Development of Interoperability Capabilities for Implants," Specialization Module, Universität Rostock, 2018.

[9] F. Segatz, „Energieanforderung einer FPGA-Implementierung einer Online-PCA für multivariate ECoG-Datensätze," Bachelorarbeit, Universität Rostock, 2019.

[10] H. Amjad, „Development of a Testbed for Approximate Computing Units," Masterarbeit, Universität Rostock, 2019.