RESEARCH

Christoph Brandau

Modellierung und Transformation digitaler Schaltungen mittels Digital Circuit Petri Nets

Springer Vieweg

Modellierung und Transformation digitaler Schaltungen mittels Digital Circuit Petri Nets

Christoph Brandau

Modellierung und Transformation digitaler Schaltungen mittels Digital Circuit Petri Nets

Mit einem Geleitwort von Prof. Dr.-Ing. Dietmar Tutsch

Springer Vieweg

Christoph Brandau
Fakultät für Elektrotechnik
Informationstechnik und Medientechnik
Bergische Universität Wuppertal
Wuppertal, Deutschland

Zugl.: Dissertation, Bergische Universität Wuppertal, 2018

ISBN 978-3-658-25243-4 ISBN 978-3-658-25244-1 (eBook)
https://doi.org/10.1007/978-3-658-25244-1

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Springer Vieweg

Springer Vieweg ist ein Imprint der eingetragenen Gesellschaft Springer Fachmedien Wiesbaden GmbH und ist ein Teil von Springer Nature
Die Anschrift der Gesellschaft ist: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

für Jenny

Geleitwort

Seit der Vorstellung der sogenannten Petri-Netze durch Carl Adam Petri zur Modellierung von Nebenläufigkeit im Jahr 1962 wurde diese Art der Stellen-Transitions-Netze kontinuierlich zur Steigerung des Modellierkomforts aber auch der Modelliermächtigkeit erweitert. Neben der Einführung von zeitbehafteten und stochastischen Petri-Netzen waren dies auch High-Level-Petri-Netze mit ihren teils objektorientierten Strukturen. Waren zu Beginn hauptsächlich die Abläufe bei technischen Prozessen und Computersystemen im Fokus der Modellierung, so dehnte sich der Anwendungsbereich später auch auf davon weiter entfernte Themen wie Prozesse in der Geschäftswelt aus. In Konsequenz wurde ein auf Petri-Netze basierendes Konzept auch für die Unified Modeling Language (UML) übernommen und im Rahmen des Aktivitätsdiagramms als eines der Verhaltensdiagramme realisiert.

Der Erfolg der Petri-Netze ist insbesondere durch die übersichtliche graphische Beschreibung von Nebenläufigkeit begründet. Insofern lag die Idee von Herrn Brandau nahe, eine solche Modellierungsart auch für digitale Schaltungen zu verwenden, in denen der Stromfluss und damit der Logikfluss auch nebenläufig stattfindet. Bisherige Modellierungsarten wie beispielsweise die Hardware-Beschreibungssprachen VHDL oder Verilog HDL erfordern fundierte Kenntnisse von deren Syntax. Zieht man diese Sprachen zum Entwurf von digitalen Schaltungen heran, so ist zusätzlich eine weitreichende Erfahrung im Umgang mit deren Sprachkonstrukten unabdingbar um synthesefähigen Code zu erzeugen. Die in diesem Buch vorliegende Arbeit von Herrn Brandau schaltet mit den Petri-Netzen eine Stufe der Modellierung vor diese Hardware-Beschreibungssprachen, so dass der Entwickler sich nicht um die Eigenheiten dieser Sprachen kümmern muss. Mit den hier vorgestellten Erweiterungen der Petri-Netz-Nomenklatur können digitale Schaltungen direkt auf der graphischen Petri-Netz-Ebene beschrieben werden. Dies betrifft nicht nur Schaltnetze, auch Schaltwerke mit ihrem speichernden Verhalten sind auf diese Weise komfortabel zu entwerfen. Damit ist der Weg eingeschlagen, hin zu einer reinen Verhaltensbeschreibung von digitalen Schaltungen mittels Petri-Netzen, aus denen dann die eigentliche Schaltung generiert werden kann.

Die große Leistung dieses Buches liegt in der hohen Qualität der mathematischen Definition des neuen Petri-Netz-Typs, den Herr Brandau mit Digital Circuit Petri Nets (DCPN) bezeichnet hat. Durch diese umfassende formale Definition werden zusätzliche Erweiterungen in Richtung Verhaltensbeschreibung auf einfache Weise ermöglicht, um das beschriebene Ziel zu erreichen. Zusätzlich wird eine Implementierung der Definitionen in einem Tool namens Logical PetriNet vorgestellt, mit dem entsprechende Schaltungen bereits beschrieben und synthetisiert werden können. Beispielschaltungen und Erläuterungen runden die Arbeit ab. Das Buch ist sehr zu empfehlen für Entwickler digitaler Schaltungen.

Prof. Dr.-Ing. Dietmar Tutsch
Lehrstuhl für Automatisierungstechnik/Informatik
Bergische Universität Wuppertal

Vorwort

Der Ansatz für meine Dissertation ist aus der Verbindung mehrerer meiner Studieninteressen entstanden. Auf der einen Seite haben mich Modellierungen von Systemen immer beschäftigt, wodurch in dieser Arbeit die Petri-Netze eine große Rolle spielen. Zum anderen haben Hardware-Beschreibungssprachen mein Interesse geweckt und ich habe mich intensiv mit diesen beschäftigt. Daraus ist die Frage entstanden, ob beide Themen nicht miteinander verknüpft werden können und wenn ja, in welcher Art und Weise dies geschehen kann. Heraus gekommen ist das vorliegende Buch, indem aus neu definierten Petri-Netzen, durch einen formalisierten Transformationsprozess, Hardware-Beschreibung generiert werden. Die Umsetzung der Forschungsergebnisse in Software zur Verifikation ist ein weiteres Interessengebiet und hat die Arbeit abgerundet und deutlich verbessert.

Ich habe die Arbeit an der Bergischen Universität Wuppertal am Lehrstuhl für Automatisierungstechnik/Informatik geschrieben. Für seine Unterstützung und ein immer offenes Ohr möchte ich meinem Doktorvater Prof. Dr.-Ing. Dietmar Tutsch danken. Ebenfalls hatte mein Zweitgutachter Prof. Dr.-Ing. habil. Carsten Gremzow immer gute Ratschläge und Hinweise, die bei der Entstehung viel beigetragen haben. Ebenfalls möchte ich Prof. Dr.-Ing. habil. Armin Zimmermann für seine Hinweise zur Verbesserung dieser Arbeit danken.

Ich Danke meinen Kollegen am Lehrstuhl dafür, dass Sie meine Launen ausgehalten und mit Ratschlägen zur Verbesserung der Arbeit beigetragen haben. Ebenso möchte ich mich bei meinen Freunden dafür bedanken, dass sie auch nach einem Jahr wenig Kontakt noch immer uneingeschränkt an meiner Seite stehen und jederzeit für mich da sind. Gleiches gilt für meine Eltern und Schwester, die mit mir viel durchmachen mussten, aber trotzdem immer vorbehaltlos zu mir gestanden haben.

Ich widme meiner Freundin, Seelenverwandten und großen Liebe Jenny diese Dissertation, ohne Sie wäre das vorliegende Werk niemals entstanden. Sie hatte immer aufmunternde Worte und eine Schulter zum Anlehnen, egal wie schwierig die Situation auch war.

Wuppertal Christoph Brandau

Inhaltsverzeichnis

Symbolverzeichnis

Symbol	Beschreibung
A	Menge der Kanten der Erreichbarkeitsgraphen
B	Menge der Bedingungen der Erreichbarkeitsgraphen
E	Menge der Knoten der Erreichbarkeitsgraphen
F	Flussrelation der Kanten des DCPN
H	Matrix der hemmenden Kanten
I	Matrix der Eingangskanten
IRG	Eingabeorientierter Erreichbarkeitsgraph
K	Menge aller Konflikte
M	Markierung des Petri-Netzes
N	Menge der Netzverbinder
O	Matrix der Ausgangskanten
P	Menge der Stellen
P_{sub}	Menge der Substellen
PN	Petri-Netz
$\mathcal{P}$	Prioritäten
RG	Erreichbarkeitsgraph
RRG	Reduzierter Erreichbarkeitsgraph
S	Schaltvektor
SF	Schaltfolge
SW	Schaltzeiten
T	Menge der Transitionen
T_{sub}	Menge der Subtransitionen
W	Wahrheitstabelle
Z	Menge der Zyklen in einem Petri-Netz

Eigenschaften

Strategien

Abbildungsverzeichnis

Tabellenverzeichnis

Listingsverzeichnis

Algorithmenverzeichnis

Kurzfassung

In dieser Arbeit wird der Entwurf und die Verifikation von digitalen Schaltungen mittels Petri-Netzen behandelt. Dabei wird die Definition der Bedingungs-Ereignis-Netze erweitert, woraus die *Digital Circuit Petri Nets* entstanden sind. Im ersten Teil der Arbeit sind die Eigenschaften dieses Netztyps aufgezeigt, welche bei der späteren Transformation in digitale Schaltungen zum Einsatz kommen.

Der Hauptteil der Arbeit beruht auf einer Petri-Netz-basierten Entwurfsmethodik zum Erzeugen von digitalen Schaltungen. Der hier vorgestellte Transformationsprozess besteht aus den Teilschritten Modellierung des Netzes, Validierung des Netzes, Netzoptimierung, Strukturanalyse, Logiksynthese, Überführung in eine Hardwarebeschreibungssprache und einer abschließenden Verifikation der erzeugten Schaltung. Schwerpunkte der Transformation sind hierbei die Synthese des Netzes und die vorherige Strukturanalyse, da aus dieser der Schaltungstyp abgeleitet wird. Hierbei werden die Schaltungen in kombinatorische und sequentielle Schaltungen aufgesplittet, um für den jeweiligen Schaltungstyp optimierte Algorithmen zur Synthese anwenden zu können.

Es werden Richtlinien zur Modellierung von Petri-Netzen für digitale Schaltungen gegeben, um die Erstellung von kombinatorischer, sowie getakteter Logik zu vereinheitlichen. Bei getakteter Logik wird zwischen synchronen und asynchronen Schaltungen unterschieden. Die Erzeugung erfolgt über einen eingabeorientierten Erreichbarkeitsgraphen, der in einen Zustandsautomaten überführt wird. Zur Verifikation des erarbeiteten Prozesses der Transformation ist das Tool *Logical PetriNet* entwickelt worden, welches dem praktischen Nachweis der vorgeschlagenen Methodik dient. Die erarbeiteten Ergebnisse der Transformation werden anhand einiger komplexer Schaltungen verifiziert.

1 Einleitung

Zum Verständnis der Wichtigkeit neue Methoden und Werkzeuge für den Entwurf digitaler Schaltungen zu geben, beginnt diese Arbeit mit dem zeitlichen Verlauf des digitalen Schaltungsentwurfs. Ein besonderes Augenmerk wird auf den Transistor gelegt, da er das grundlegende Element dieser Schaltung ist. Im Jahre 1947 wurde in den *Bell Laboratories* der erste funktionierende Transistor entwickelt. Etwa zehn Jahre später entwickelte *Texas Instruments* die Technik, um mehrere Transistoren als integrierte Schaltung in einem Silizium-Chip zu realisieren. Kurz darauf hat Jack Kilby das erste integrierte Flip-Flop, bestehend aus zwei Transistoren, erstellt [BL10].

1963 entwickelt Frank Wanlass [WS63] die ersten Logikgatter basierend auf Metall-Oxid-Halbleiter-Feldeffekttransistoren (*metal-oxide-semiconductor field-effect transistor*, *MOSFET*). Hierbei kommen *nMOS*- und *pMOS*-Transistoren zum Einsatz, wobei immer ein n- und ein p-Transistor zu einem Element zusammen geschaltet werden [Bea68]. Die Verwendung von beiden Transistortypen auf einem Chip wird auch als sich ergänzender Metall-Oxid-Halbleiter (*complementary metal-oxide-semiconductor*, *CMOS*) bezeichnet.

Transistoren sind zunächst in analogen Schaltungen, wie beispielsweise in Transistor-Radios, verbaut worden, bis sie für digitale Schaltungen, wie Prozessoren, zum Einsatz kommen [WH10], [MC79]. Der Transistor ist heutzutage das am häufigsten produzierte elektrische Bauteil der Welt. Die Transistoranzahl pro Chip wurde von Gordon Moore [Moo06] untersucht und er bestätigt eine Verdopplung alle 18 Monate. Daraus ist das *Mooresche Gesetz* abgeleitet, welches bis heute seine Gültigkeit behält, wobei eine Verdopplung etwa alle 26 Monate stattfindet. Gründe hierfür sind immer kleiner werdende Transistoren und teilweise größer werdende Chipflächen. Die Integration von Chips lässt sich in die Klassen *Small-Scale Integration* (*SSI*) mit ca. 10 Gattern auf einem Chip, *Medium-Scale Integration* (*MSI*) mit bis zu 1.000 Gattern, *Large-Scale Integration* (*LSI*) mit bis zu 10.000 Gattern und *Very-Large-Scale Integration* (*VLSI*) mit mehr als 10.000 Gattern pro Chip aufteilen.

C. Brandau, *Modellierung und Transformation digitaler Schaltungen mittels Digital Circuit Petri Nets*, https://doi.org/10.1007/978-3-658-25244-1_1

1.1 Motivation

Seit 1981 können die Entwurfsverfahren zur Erzeugung digitaler Schaltungen immer weniger mit den technischen Möglichkeiten der Halbleiterfertigung Schritt halten. Während aus technischer Sicht die mögliche Anzahl der Transistoren pro Chip jährlich um 58% ansteigt, liegt der Anstieg der verwendeten Transistoren auf Entwurfsebene bei 21% pro Jahr. Die aufgrund dessen stetig wachsende Lücke zwischen diesen beiden Aspekten wird als *Design Productivity Gap* bezeichnet (siehe Abbildung 1.1). Um der technischen Entwicklung folgen zu können, müssen immer abstraktere Methoden zum Entwurf digitaler Schaltungen entwickelt werden. Je nach Modellierungsvorhaben kann die Beschreibung eines Systems auf verschiedenen Ebenen vonstatten gehen. So kann beispielsweise auf der Systemschicht oder auf der Register-Transfer-Ebene modelliert werden.

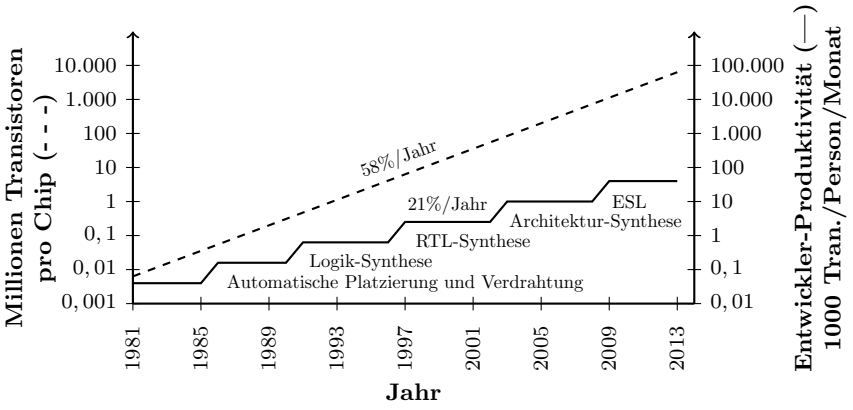

Abbildung 1.1: Entwicklung der Transistoranzahl pro Chip (gestrichelte Linie) im Verhältnis zur Entwicklerproduktivität (durchgezogene Linie) nach [RU01] und [Rei13]

Ziel dieser und folgender Arbeiten ist es, das Verhalten und die Struktur digitaler Schaltungen mittels einer grafischen Repräsentation zu modellieren und die daraus resultierenden Vorteile bezüglich der Verifikation und Validierung zu nutzen. Aus diesem Grund werden in der vorliegenden Arbeit Möglichkeiten zur Beschreibung von Strukturen digitaler Schaltungen unter Zuhilfenahme von Petri-Netzen vorgestellt. Petri-Netze sind formale

Beschreibungen von Systemen, die aus den vier Elementen Stellen, Marken, Transitionen und gerichteten Kanten bestehen. Für diese Netze existieren viele Methoden zur Validierung und Analyse, wie beispielsweise die Markov-Ketten [Mar13]. Zur Verwendung von Petri-Netzen werden in dieser Arbeit zusätzliche Elemente und Eigenschaften eingeführt, die eine Vereinheitlichung der Transformation eines Netzes in eine Schaltung ermöglichen.

Petri-Netze sind dabei für den Modellierer greifbarer als die Entwicklung von digitalen Schaltungen in VHDL oder anderen Hardware-Beschreibungssprachen, da bei den Petri-Netzen mittels grafischer Methoden modelliert werden kann. Die Motivation ist ein Verfahren zu entwickeln, bei dem die Abstraktionsebene frei gewählt werden kann. Mittels Petri-Netzen soll also eine hierarchische Anordnung der Elemente des Netzes erfolgen. Die erste Sicht auf das modellierte System kann eine sehr abstrakte Sicht sein, indem das Gesamtsystem mit einzelnen Komponenten abgebildet wird. Durch diese Art der Modellierung wird eine übersichtliche Form der Beschreibung des Systems ermöglicht. Hierzu soll in einem zu Implementierenden Werkzeug die Option bestehen, die Elemente von Petri-Netzen mittels Algorithmen optimiert zu platzieren. Hierzu können Ansätze aus der automatischen Verteilung von Graphen eingesetzt werden. Bisher bieten die Petri-Netze vor allem im Bereich der Steuerwerke und besonders bei asynchronen Schaltungen schon jetzt Analyse-Methoden, die im Bereich des reinen Schaltungsentwurf derzeit nicht verfügbar sind. Hier sind die Methoden der Petri-Netze viel tiefgehender und eignen sich zur Validierung der modellierten Schaltungen [Lie05].

Damit die Durchführung der Transformation von Petri-Netzen in digitale Schaltungen weitestgehend automatisiert ablaufen kann, muss zunächst ein Formalismus aufgestellt werden. Zur Lösung nicht-trivialer Probleme und zur Unterstützung des Anwenders sind entsprechende Methoden zur Transformation in eine geeignete Software zu integrieren. Durch den aufgestellten Formalismus soll es möglich sein sequentielle und nebenläufige Ereignisse zu modellieren. Weiterhin muss die Systemstruktur und das Systemverhalten mit dem Modell wiedergegeben werden. Durch eine Hierarchiebildung ist es möglich das System auf verschiedenen Abstraktionsebenen zu betrachten. Darüber hinaus muss die Möglichkeit bestehen das System funktional und zeitbehaftet zu simulieren. Durch die formale Petri-Netz-Spezifikation ist eine Analyse auf vielfältige Arten durchführbar, da die schon bestehenden Methoden mit Anpassungen für den neuen Petri-Netz-Typ verwendet werden können.

Die meisten der in dieser Arbeit vorgestellten Verfahren lassen sich auch auf die Rücktransformation von digitalen Schaltungen in Petri-Netze an-

wenden, was ein nützliches Nebenprodukt der Arbeit darstellt. Eine solche Rücktransformation bietet die Möglichkeit, die in dieser Arbeit beschriebenen Analysemethoden von Petri-Netzen verwenden zu können, um digitale Schaltungen zu untersuchen. Daraus folgt eine formale Validierung des Netzes und Verifikation der modellierten Schaltungen. Der noch zu erledigende Arbeitsschritt zur Rücktransformation von Schaltungen in Petri-Netze ist das Anlegen von Netzen für alle benötigten Schaltelemente.

Es stellt sich die Frage warum ausgerechnet Petri-Netze aus den bekannten und verbreiteten grafischen Sprachen zur Modellierung von Systemen zum Einsatz kommen sollten. Für die Modellierung könnten auch Zustandsautomaten eingesetzt werden, bei denen aber vor allem bei komplexeren Systemen bei der Analyse Zustandsraumexplosionen auftreten. Beim Aufteilen des Systems in mehrere kleinere Automaten tritt das Problem der Synchronisierung und das Problem einer adäquaten Abbildung der Parallelität auf [YK98]. Die Vorteile von Petri-Netzen sind die gute Verifizierbarkeit der erzeugten Netze und die einfache Erstellung von Netzen über Werkzeuge zur Modellierung, die vor allem grafischer Natur sind. Mit Petri-Netzen können asynchrone Systeme beschrieben werden, wobei die synchronen einen Spezialfall dieser Systeme darstellen, bei denen der Takt ein zusätzliches Signal zur Generierung eines Events ist. Nach Möglichkeit sollte jedoch zwischen den beiden Typen unterschieden werden, da so bessere Optimierungsmethoden für jeden Systemtyp angewandt werden können. Als zusätzlicher Netztyp existiert noch reine Kombinatorik, die ebenfalls abgebildet werden sollte.

Wünschenswert sind die folgenden Charakteristiken bei der Modellierung mittels Petri-Netzen: Zum einen soll eine präzise Syntax und Semantik des Netzes vorhanden sein. Zum anderen ist eine lesbare Modellierung, am besten mit gleichzeitiger Verifikation des Modells, und die Abbildung von Parallelität und Synchronisierung interessant. Ebenfalls wünschenswert wäre eine der Modellierungen *top-down* oder *bottom-up*. Bei Petri-Netzen bietet sich durch die Bildung von Hierarchien vor allem die top-down-Methode an, da so das Netz in der obersten Ebene betrachtet werden kann und durch weitere hierarchische Elemente des Netzes mehr Details angezeigt werden können. Dabei wird auch von einer Verfeinerung des Systems gesprochen.

1.2 Ziel der Arbeit

Das Ziel dieser Arbeit ist ein Verfahren zu erarbeiten und zu formalisieren, welches die Transformation von Petri-Netzen in digitale Schaltungen ermöglicht. Dabei ist ein modularer Aufbau anzustreben, um spätere Er-

weiterungen und Optimierungen mit möglichst wenig Aufwand implementieren und hinzufügen zu können. Bisherige Ansätze und Definitionen von Petri-Netzen sind auf ihre sinnvolle Anwendung im genannten Umfeld zu überprüfen, wobei der Aspekt der Modellierung ohne Vorkenntnisse von digitalen Schaltungen zu berücksichtigen ist.

Die Transformation soll, soweit dies möglich ist, auf bestehende Hardwarebeschreibungssprachen zurückgreifen, um die vorhandenen Tools im Bereich der Hardwarebeschreibung zur Optimierung und Platzierung zu verwendbar zu machen. Eine erneute Implementierung dieser Werkzeuge in einem eigenen Programm übersteigt den Rahmen dieser Arbeit, da die entwickelten Werkzeuge teilweise schon mehrere Jahrzehnte in Entwicklung von namhaften Herstellern wie Xilinx, Altera oder Mentor Graphics sind. Diese Werkzeuge haben sich bewährt und eine Neuimplementierung ist daher nicht notwendig.

Die Modellierung von Petri-Netzen soll für den späteren Entwickler von digitalen Schaltungen möglich sein, ohne dass dieser Kenntnisse vom Entwurf von digitalen Schaltungen besitzen muss. Um dies zu bewerkstelligen, ist ein Tool zu implementieren, welches zum einen eine handliche Modellierung eines Petri-Netzes zur Verfügung stellt und auf der anderen Seite den Transformationsprozess abbilden kann und zur Verifizierung der erarbeiteten Methoden und Strategien dient.

Weiterhin soll der hier erarbeitete Transformationsprozess, anders als in bisherigen Forschungsarbeiten üblich, das gesamte Netz analysieren und die klassische und bisher übliche Transformation von Struktur in Struktur (auch Komponentenweise genannt) aufbrechen. Im Abschnitt zum aktuellen Stand der Technik werden existierende Ansätze zur Transformation von Petri-Netzen in digitale Schaltungen vorgestellt und analysiert, welche alle auf dem Ansatz der Strukturüberführung basieren. Vorteile dieser Verfahren sind nach Möglichkeit in die zu entwickelnden Methoden und Strategien aufzunehmen.

Eine hierarchische Darstellung innerhalb der Petri-Netze wäre wünschenswert, um die Übersichtlichkeit der modellierten Netze zu gewährleisten. Hierzu sind vorhandene Methoden zur Hierarchiebildung zu untersuchen beziehungsweise eigene Subnetze zu erarbeiten und im entstehenden Programm zu integrieren. Weiterhin sollen die Subnetze dazu dienen, um schon modellierte Petri-Netze in neu zu erstellenden Netzen als Bibliotheks-Elemente zu verwenden.

Eine Verifizierung der beschriebenen Methoden anhand von Test-Netzen und deren Transformation soll die Arbeit abschließen.

1.3 Aufbau der Arbeit

Die Arbeit beginnt mit dem Ziel und der Motivation in diesem Kapitel, woraufhin im zweiten Kapitel eine Einführung in die Theorie der Petri-Netze und eine Beschreibung der Einsatzgebiete dieser Netze folgt. Daran anknüpfend folgt die Vorstellung verschiedener Analysemethoden, die bei der Untersuchung von Petri-Netzen zur Anwendung kommen können. Nachfolgend wird eine grundlegende Einführung in den Bereich der digitalen Schaltungen und die unterschiedlichen Strukturen von Schaltungen gegeben. Daran anschließend erfolgt die Erläuterung der beiden Hardwarebeschreibungssprachen Verilog und VHDL. Der aktuelle Stand der Forschung rundet dieses Kapitel ab.

Im dritten Kapitel erfolgt die Erweiterung der im vorhergehenden Kapitel eingeführten Petri-Netze zur eigenen Klasse der *Digital Circuit Petri Nets* (DCPN) mit den Erweiterungen um Eingangs- und Ausgangsstellen und Subnetzen in unterschiedlichen Ausprägungen. Die Definition der DCPN vollendet die Erweiterungen des neuen Netztyps. Abgeschlossen wird das Kapitel mit den Eigenschaften der DCPN, die für eine Transformation in eine Hardwarebeschreibungssprache benötigt werden.

In Kapitel Vier wird die Transformation von Petri-Netzen in VHDL beschrieben. Zunächst werden verschiedene Elemente der Digitaltechnik sowie ihre Abbildung als Petri-Netz vorgestellt. Dadurch stehen Netze mit bekannten Eingangs- und Ausgangskombinationen zur Verfügung. Anschließend werden verschiedene Beschreibungsmöglichkeiten eines Taktes innerhalb eines Petri-Netzes diskutiert. Darauf folgt der Entwurfsprozess, der die einzelnen Schritte erläutert um von einem Petri-Netz zu einer Hardwarebeschreibung zu gelangen. Dabei wird in einigen Schritten der Transformation zwischen zwei möglichen Verfahren unterschieden, je nachdem ob es sich um kombinatorische oder sequentielle Logik handelt.

Eine mögliche Implementierung der hier vorgestellten Verfahren erfolgt in der Software *Logical PetriNet*, die im fünften Kapitel vorgestellt wird. Einige nicht triviale Beispiele zeigen die Methoden des Tools und verifizieren die Ergebnisse der vorhergehenden Kapitel. Ebenfalls in diesem Kapitel werden hilfreiche Empfehlungen zur Modellierung von Petri-Netzen gegeben, um diese in digitale Schaltungen überführen zu können.

Im folgenden Kapitel wird ein Fazit dieser Arbeit und eine Abgrenzung zu den Veröffentlichungen aus dem Stand der Forschung gegeben und zeigt Unterschiede zu diesen auf. Abgeschlossen wird die Arbeit mit einer Zusammenfassung und dem Ausblick auf mögliche weitere Forschungsarbeiten im Bereich der Transformation von Petri-Netzen in digitale Schaltungen.