

Muhammad Hassan
Daniel Große
Rolf Drechsler

Erweiterte virtuelle Prototypen für heterogene Systeme

Erweiterte virtuelle Prototypen für heterogene Systeme

Muhammad Hassan · Daniel Große ·
Rolf Drechsler

Erweiterte virtuelle Prototypen für heterogene Systeme

 Springer Vieweg

Muhammad Hassan
DFKI GmbH
Bremen, Deutschland

Daniel Große
Johannes Kepler University of Linz
Linz, Österreich

Rolf Drechsler
University of Bremen and DFKI GmbH
Bremen, Deutschland

ISBN 978-3-031-53151-4 ISBN 978-3-031-53152-1 (eBook)
<https://doi.org/10.1007/978-3-031-53152-1>

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <https://portal.dnb.de> abrufbar.

Übersetzung der englischen Ausgabe: „Enhanced Virtual Prototyping for Heterogeneous Systems“ von Muhammad Hassan et al., © The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2023. Veröffentlicht durch Springer International Publishing. Alle Rechte vorbehalten.

Dieses Buch ist eine Übersetzung des Originals in Englisch „Enhanced Virtual Prototyping for Heterogeneous Systems“ von Muhammad Hassan, publiziert durch Springer Nature Switzerland AG im Jahr 2023. Die Übersetzung erfolgte mit Hilfe von künstlicher Intelligenz (maschinelle Übersetzung). Eine anschließende Überarbeitung im Satzbetrieb erfolgte vor allem in inhaltlicher Hinsicht, so dass sich das Buch stilistisch anders lesen wird als eine herkömmliche Übersetzung. Springer Nature arbeitet kontinuierlich an der Weiterentwicklung von Werkzeugen für die Produktion von Büchern und an den damit verbundenen Technologien zur Unterstützung der Autoren.

© Der/die Herausgeber bzw. der/die Autor(en), exklusiv lizenziert an Springer Nature Switzerland AG 2024

Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsgesetz zugelassen ist, bedarf der vorherigen Zustimmung des Verlags. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

Die Wiedergabe von allgemein beschreibenden Bezeichnungen, Marken, Unternehmensnamen etc. in diesem Werk bedeutet nicht, dass diese frei durch jedermann benutzt werden dürfen. Die Berechtigung zur Benutzung unterliegt, auch ohne gesonderten Hinweis hierzu, den Regeln des Markenrechts. Die Rechte des jeweiligen Zeicheninhabers sind zu beachten.

Der Verlag, die Autoren und die Herausgeber gehen davon aus, dass die Angaben und Informationen in diesem Werk zum Zeitpunkt der Veröffentlichung vollständig und korrekt sind. Weder der Verlag noch die Autoren oder die Herausgeber übernehmen, ausdrücklich oder implizit, Gewähr für den Inhalt des Werkes, etwaige Fehler oder Äußerungen. Der Verlag bleibt im Hinblick auf geografische Zuordnungen und Gebietsbezeichnungen in veröffentlichten Karten und Institutionsadressen neutral.

Planung/Lektorat: Charles Glaser

Springer Vieweg ist ein Imprint der eingetragenen Gesellschaft Springer Nature Switzerland AG und ist ein Teil von Springer Nature.

Die Anschrift der Gesellschaft ist: Gewerbestrasse 11, 6330 Cham, Switzerland

Das Papier dieses Produkts ist recycelbar.

*An Khushboo und Maryam,
Marie,
und
Anna.*

Vorwort

Die erfolgreiche Co-Design und Verifizierung von sicheren multidisziplinären heterogenen *Systemen-auf-Chips* (SOCs) mit engen Interaktionen zwischen *Hardware/Software* (HW/SW) Systemen und ihrer analogen physischen Umgebung ist eine zunehmend entmutigende Aufgabe. In dieser Hinsicht hat das Aufkommen von *virtuellen Prototypen* (VPs) auf der Abstraktionsebene des *elektronischen Systemlevels* (ESL) das Design und die Verifizierung von heterogenen SOC's modernisiert. Ein VP ist im Wesentlichen ein ausführbares abstraktes Modell der gesamten HW-Plattform und wird vorwiegend in der C++-basierten Systemmodellierungssprache SystemC zusammen mit *Transaktionslevel-Modellierung* (TLM) Techniken und seiner Mixed-Signal-Erweiterung SystemC AMS erstellt. Die deutlich frühere Verfügbarkeit sowie die signifikant schnellere Simulation im Vergleich zu den *Register Transfer Level* (RTL) Modellen und SPICE-Level-Modellen gehören zu den Hauptvorteilen von VPs. Daher ermöglicht das virtuelle Prototyping das HW/SW Co-Design und die Verifizierung sehr früh im Designprozess. Als Referenz für die (frühe) eingebettete SW-Entwicklung und HW-Verifizierung ist die funktionale Korrektheit und Sicherheitsvalidierung von VPs sehr wichtig. Daher wird ein VP einer rigorosen funktionalen Verifizierung unterzogen. Allerdings hat der moderne VP-basierte Verifizierungsfluss immer noch Schwächen, insbesondere aufgrund des Fehlens von Methoden zur Erfassung der komplexen Interaktionen zwischen digitalen und analogen Designs sowie der Nichtverfügbarkeit von Sicherheitsvalidierungstechniken. Dieses Buch schlägt mehrere neuartige Ansätze vor, die verschiedene Verifizierungsaspekte abdecken, um den modernen VP-basierten Verifizierungsfluss stark zu verbessern. Die Kapitel des Buches sind im Wesentlichen in vier Teile gegliedert: Der erste Teil führt eine neue Verifizierungsperspektive für VPs ein, indem er *metamorphes Testing* (MT) verwendet, da keine Referenzmodelle/Werte für die Verifizierung benötigt werden, im Gegensatz zum modernen VP-basierten Verifizierungsfluss. Der zweite Teil verbessert die Code-Coverage-Closure-Methoden im modernen VP-basierten Verifizierungsfluss durch Berücksichtigung von *Mutationsanalyse* und stärkeren Coverage-Metriken wie *Datenfluss-Coverage*. Der dritte Teil

behandelt eine Reihe von neuartigen, systematischen und leichtgewichtigen funktionalen Coverage-getriebenen Verifizierungsmethoden zur Verbesserung der Coverage-Closure. Der vierte und letzte Teil des Buches präsentiert neuartige Ansätze zur frühzeitigen Sicherheitsvalidierung von VPs. Alle Ansätze werden detailliert vorgestellt und umfangreich mit mehreren Experimenten bewertet, die ihre Wirksamkeit bei der starken Verbesserung des modernen VP-basierten Verifizierungsflusses deutlich demonstrieren.

Bremen, Deutschland
Linz, Österreich
Bremen, Germany
November 2021

Muhammad Hassan
Daniel Große
Rolf Drechsler

Danksagungen

Zunächst möchten wir den Mitgliedern der Forschungsgruppe für *Computerarchitektur* (AGRA) an der Universität Bremen sowie den Mitgliedern der Forschungsabteilung für *Cyber-Physische Systeme* (CPS) am *Deutschen Forschungszentrum für Künstliche Intelligenz* (DFKI) in Bremen danken. Wir schätzen die großartige Atmosphäre und anregende Umgebung. Darüber hinaus möchten wir allen Mitautoren der Arbeiten danken, die den Ausgangspunkt für dieses Buch bildeten: Hoang M. Le, Vladimir Herdt, Mingsong Chen und Mehran Goli. Wir danken insbesondere Karsten Einwich und Thilo Vörtler von COSEDA Technologies GmbH für viele interessante Diskussionen und erfolgreiche Zusammenarbeiten.

Inhaltsverzeichnis

- 1 Einleitung** 1
 - 1.1 Design und Verifizierung auf elektronischer Systemebene 4
 - 1.2 Buchbeitrag 8
 - 1.2.1 Beitragsbereich 1: AMS Metamorphic Testing Umgebung 10
 - 1.2.2 Beitrag Bereich 2: Verbesserte AMS Code Coverage Verifizierungsumgebung 11
 - 1.2.3 Beitrag Bereich 3: Verbesserte AMS Funktionsabdeckungs-Verifizierungsumgebung 12
 - 1.2.4 Beitrag Bereich 4: Frühe digitale Sicherheitsvalidierung ... 13
 - 1.2.5 Zusammenfassung der Beiträge 14
 - 1.3 Buchorganisation 14
- 2 Vorarbeiten** 15
 - 2.1 SystemC 15
 - 2.1.1 Grundlagen 16
 - 2.1.2 Transaktions-Level-Modellierung (TLM) 18
 - 2.2 SystemC AMS 20
 - 2.2.1 Modelle der Berechnung (MOC) 20
 - 2.2.2 Zeitgesteuerte Datenfluss (TDF) 20
- 3 AMS Metamorphic Testing Umgebung** 23
 - 3.1 MT-basierter System-Level-Verifizierungsansatz 24
 - 3.1.1 Überblick 25
 - 3.1.2 Testfallgenerator 26
 - 3.1.3 Metamorphe Beziehungen 26
 - 3.1.4 Kerneigenschaften 26
 - 3.2 Metamorphes Testen für RF-Verstärker 27
 - 3.2.1 MT-Prinzip für RF-Verstärker 27
 - 3.2.2 Identifizierung von Metamorphen Beziehungen 28
 - 3.2.3 Experimentelle Bewertung 31

3.3	Metamorphes Testen für PLLs	37
3.3.1	Phase-Locked Loop	38
3.3.2	MT-Prinzip für gemischte Signalinteraktionen	40
3.3.3	Identifizierung von MRs für PLLs	40
3.3.4	Experimente	42
3.4	Zusammenfassung	45
4	AMS verbesserte Code-Coverage-Verifizierungsumgebung	47
4.1	Software Driven Verification für IP-Integration	48
4.1.1	SW-Test-Qualifikationsmethodik	50
4.1.2	Beispiel zur Demonstration der Konsistenz	56
4.1.3	Experimentelle Ergebnisse	61
4.2	Datenflusstest für digitale virtuelle Prototypen	68
4.2.1	SystemC Laufendes Beispiel	69
4.2.2	Def-Use-Verknüpfung und Datenflusstest	70
	Datenflusstest für SystemC	71
4.2.4	Implementierungsdetails	78
4.2.5	Experimentelle Ergebnisse	80
4.3	Datenflusstest für SystemC AMS Virtuelle Prototypen	81
4.3.1	SystemC AMS Motivierendes Beispiel	81
4.3.2	Datenflusstest für SystemC-AMS TDF-Modelle	83
4.3.3	Implementierungsdetails	90
4.3.4	Experimentelle Ergebnisse	91
4.4	Zusammenfassung	93
5	AMS verbesserte funktionale	
	Abdeckungsverifizierungsumgebung	95
5.1	Grundlagen	98
5.1.1	Funktionale Abdeckung	98
5.1.2	AMS VP Verifizierungsumgebung und Mängel	99
5.2	Erweiterte Einrichtung der funktionalen	
	Abdeckungsverifizierungsumgebung	100
5.2.1	Laufendes Beispiel: LNA	100
5.2.2	Umgebungseinrichtung	101
5.3	AMS Funktionaler Abdeckungsgetriebener	
	Verifizierungsansatz	107
5.3.1	Abdeckungsanalyse	107
5.3.2	Industriestudie	110
5.4	Leichtgewichtige Erzeugung von Abdeckungsgerichteten	
	Stimuli	113
5.4.1	Überarbeitung der Definition und Sammlung	
	der Ausgabedeckung	114
5.4.2	Leichtgewichtige Deckungsanalyse	116
5.4.3	Experimentelle Bewertung	123
5.5	Zusammenfassung	132

6	Digitale frühzeitige Sicherheitsvalidierung	135
6.1	Statische Informationsflussanalyse	137
6.1.1	Überblick über den Ansatz	137
6.1.2	Datenflussgesteuerte Informationsflussanalyse	141
6.1.3	Experimentelle Ergebnisse	149
6.2	Dynamische Informationsflussanalyse	152
6.2.1	Motivierendes Beispiel und Bedrohungsmodelle	153
6.2.2	Dynamische IFT-Methodik	157
6.2.3	Experimentelle Bewertung	166
6.3	Zusammenfassung	170
7	Schlussfolgerung	173
7.1	Zukünftige Richtungen	175
	Bibliography	177

Liste der Algorithmen

Algorithmus 1	Vorgeschlagener LCDG-Ansatz zur Erfassung von linearen, nicht-linearen und instabilen Verhaltensweisen	109
Algorithmus 2	Direkte Sicherheitseigenschaftserzeugung	161
Algorithmus 3	Indirekte Sicherheitseigenschaftserzeugung	162
Algorithmus 4	Erkennung von Sicherheitsverletzungen	165

Abbildungsverzeichnis

Abb. 1.1	Frühe SW-Entwicklung unter Nutzung des Shift-Left-Konzepts	2
Abb. 1.2	Platzierung von VP im vollständigen SOC-Designfluss	4
Abb. 1.3	Allgemeine AMS VP Verifizierungsumgebung.	6
Abb. 1.4	Vier Stufen des modernen simulationsbasierten VP-Verifizierungsablaufs	7
Abb. 1.5	Vorgeschlagene Verbesserungen im VP-Verifizierungsfluss	9
Abb. 1.6	Beiträge des Buches: Verbesserter VP-Verifizierungsfluss	10
Abb. 2.1	SystemC-Architektur mit TLM- und AMS-Erweiterungen [1, 71]	16
Abb. 2.2	TLM: Initiator, Ziel, Socket und Verbindung	18
Abb. 2.3	Simulationsgeschwindigkeit von Modellierungssprachen im Vergleich zu SPICE [9]	19
Abb. 2.4	Tiefpassfilter zweiter Ordnung	21
Abb. 3.1	Aufkommende Verifizierungstechniken	24
Abb. 3.2	Beispiel für metamorphes Testen – SUM-Funktion	25
Abb. 3.3	Grafische Darstellung der MR für Verstärker Ausgang = 7 × Eingang. (a) Basistest-Stimulus: $x(t) = \sin(2\pi 5000t)$. (b) Verstärkerausgang beim Basistest-Stimulus. (c) Der Nachfolgetest-Stimulus: $3 \times x(t) = 3 \sin(2\pi 5000t)$. (d) Verstärkerausgang beim Nachfolgetest-Stimulus	29
Abb. 3.4	I_LNA Ausgangsleistung vs. Eingangsleistung (dBm). Überschwingen der Ausgangsleistung aufgrund von Nichtlinearitätsapproximation.	33
Abb. 3.5	Fehlererkennungsqualität der MT-basierten Verifikation	35
Abb. 3.6	Verstärkungsausgabe von FLNA.	35
Abb. 3.7	Leistungsausgabe von FLNA	36
Abb. 3.8	PLL Top-Level-Diagramm	38
Abb. 3.9	Grafische Darstellung der MR für die Ladungspumpe (CP)	39
Abb. 3.10	Fehlerhaftes Verhalten der PLL – Totbereich-Effekt aufgedeckt durch MR4	42

Abb. 3.11	FFT des fehlerhaften Verhaltens der PLL	43
Abb. 3.12	Phasenfrequenzdetektor (PFD) ohne Verzögerungselement	43
Abb. 3.13	Verhalten der PLL nach Hinzufügung eines Verzögerungselements im PFD	44
Abb. 4.1	Erweiterte strukturelle Abdeckungsverifizierung	48
Abb. 4.2	SW-Qualifikationsmethodik	55
Abb. 4.3	SC_THREAD find_max mit den ursprünglichen Abdeckungsergebnissen für Tests 1 und 2	58
Abb. 4.4	Konsistenzbeispiel: Abdeckungsergebnisse für C2-Beispiel	59
Abb. 4.5	Konsistenzbeispiel: Abdeckungsergebnisse für C4-Beispiel	60
Abb. 4.6	Eine Übersicht über unseren Datenflusstestansatz für SystemC	71
Abb. 4.7	Sensorsystem – T: Temperatur, H: Feuchtigkeit, XS: X Sensor (X=T,H), Z^{-1} = Verzögerung, AMUX: Analogmux, ADC: Analog-Digital-Wandler, X_LED: Leuchtdiode, G: Verstärkung	82
Abb. 4.8	Vorgeschlagener Überblick über die Datenflusstestmethodik	84
Abb. 5.1	Verbesserte funktionale Abdeckungsverifizierung	96
Abb. 5.2	Erweiterte funktionale Abdeckungsverifizierungsumgebung für AMS – LNA: Low Noise Amplifier, BPF: Band-Pass Filter, res: Auflösung, ired: Intervallauflösung	99
Abb. 5.3	Grafische Darstellung des Auflösungs-Parameters. (a) Definition der Stimuli-Parameterauflösung mit 0,2 V Amplitudendifferenz. (b) Definition der Stimuli- Parameterauflösung mit 0,2 Hz Frequenzdifferenz.	102
Abb. 5.4	Überblick über den AMS funktionalen abdeckungsgetriebenen Ansatz – res: Auflösung, ired: Intervallauflösung	108
Abb. 5.5	Verstärkung (G_1) gegen Eingangsleistung	110
Abb. 5.6	Fallstudie: RF-Sender- und Empfängermodell	111
Abb. 5.7	Fallstudie – Abdeckungsschluss in Bezug auf die Auflösung.	112
Abb. 5.8	Überblick über den LCDG-Ansatz – res: Auflösung, ired: Intervallauflösung, L/NL: lineare/nichtlineare Abdeckungsziele, O/U: Überschwingen/Unterschwingen Abdeckungsziele, SCG: sekundäre Erweiterungsabdeckungsziele	114
Abb. 5.9	Verstärkungskurve (Ausgangsleistung vs. Eingangsleistung) – Überschwingverhalten aufgrund der Taylor-Reihen-Approximation	131
Abb. 6.1	Frühe Sicherheitsvalidierung.	136
Abb. 6.2	Motivierendes Beispiel	139
Abb. 6.3	Überblick über die Informationsflussanalyse	140
Abb. 6.4	Ein Zugriffspfad zum Beispiel in Auflistung 6.1 gezeigt	145
Abb. 6.5	Die Architektur des Motivationsbeispiels RISC-32 SOC	155
Abb. 6.6	Die Architektur der vorgeschlagenen Methodik	157
Abb. 6.7	Datenextraktion. (a) Teil der generierten Debug-Symbole und (b) Teil der generierten GDB-Befehlsdatei des RISC-32 SOC-Modells	159

Tabellenverzeichnis

Tab. 4.1	Zusammenfassung der SystemC-spezifischen Mutationoperatoren.	52
Tab. 4.2	Konsistenzanalyseergebnisse für eine Mutation.	53
Tab. 4.3	Konsistenzbeispiel: Abdeckungslegende	58
Tab. 4.4	Ergebnisse der IRQMP SW-Test-Qualifikation	65
Tab. 4.5	GPTimer SW Test Qualifikationsergebnisse	67
Tab. 4.6	Datenfluss-Assoziationen für das SystemC-Beispiel in Auflistung 4.6 sortiert nach Klassifizierung.	75
Tab. 4.7	SystemC-AMS TDF-Modelle-spezifische Datenfluss-Assoziationen – Referenz Auflistung 4.7.	88
Tab. 4.8	Fallstudie: Datenflussassoziationen des Autofensterheber- Systems und des Buck-Boost-Konverters	91
Tab. 5.1	LNA: Parameter A (Amplitude) Abdeckungsbericht	104
Tab. 5.2	LNA: Parameter F (Frequenz) Abdeckungsbericht	104
Tab. 5.3	LNA: Eingangsleistung Abdeckungsbericht	104
Tab. 5.4	LNA Verstärkung (G_1) Ausgangsabdeckungsbericht	106
Tab. 5.7	Cross-Coverage fehlerfreies DUV (Auszug) – Prüfer gegen Eingangsleistung gegen Verstärkung (G_1)	111
Tab. 5.5	LNA-Verstärkung (G_1) Abdeckungsbericht. Fall C2: Hinzufügen von Bins im Verstärkungs-Coverpoint zwischen 18,8 dB und 19,9 dB.	109
Tab. 5.6	LNA-Verstärkung (G_1) Abdeckungsbericht. Fall C1: Statische Parameterverfeinerung bei Eingangsreizen	111
Tab. 5.8	LNA-Gewinn (G_2) Ausgabedeckungsbericht – Iteration 1	115
Tab. 5.9	LNA-Gewinn (G_2) – Überschreiten/Unterschreiten Deckungsziele	115
Tab. 5.10	LNA-Gewinn (G_2) – Sekundärerweiterungsziele.	116
Tab. 5.11	Auszug aus der LNA-Gewinn (G_2) Ausgabespur nach Iteration 1.	116
Tab. 5.12	LNA-Verstärkung (G_2) Ausgabeabdeckungsbericht – Iteration 2.	122

Tab. 5.13	LNA-Verstärkung (G_2) Ausgabeabdeckungsbericht – Iteration 4	122
Tab. 5.14	LNA-Verstärkung (G_2) Ausgabeabdeckungsbericht – Iteration 6	122
Tab. 5.15	LNA-Verstärkung (G_2) – Sekundärerweiterungsziele	124
Tab. 5.16	Cross-Coverage (Auszug) – Prüfer vs. Eingangsleistung vs. Verstärkung	124
Tab. 5.17	Cross-Coverage (Auszug) – Frequenz vs. Eingangsleistung vs. Verstärkung	124
Tab. 5.18	Spezifikationen von sieben industriellen LNAs	126
Tab. 5.19	LNAs Fallstudie – Verstärkung (G) Fortschritt über mehrere Iterationen.	127
Tab. 5.20	LNAs Fallstudie – 1 dB Kompressionspunkt Fortschritt über mehrere Iterationen	128
Tab. 5.21	LNAs Fallstudie – Eingangs-Intercept-Punkt dritter Ordnung (IIP3) Fortschritt über mehrere Iterationen.	129
Tab. 5.22	LNAs Fallstudie – Eingangs-Zweiter-Ordnung- Intercept-Punkt (IIP2) Fortschritt über mehrere Iterationen	130
Tab. 6.1	Experimentelle Ergebnisse für alle Fallstudien	163

Kapitel 1

Einleitung



Internet der Dinge (IOT) und *5G* (fünfte Generation der Technologiestandards für Breitband-Mobilfunknetze) haben eine Fülle von Möglichkeiten ermöglicht, die einst undenkbar waren. Während *5G* die Hochgeschwindigkeitsverbindung und die allgegenwärtige Abdeckung bietet, sind es die intelligenten IOT-Geräte, die die Daten sammeln und transportieren, die das Versprechen und das Potenzial von IOT antreiben. Diese intelligenten Geräte sind ein hervorragendes Beispiel für heterogene *System-On-Chips* (SOCs), die aus zwei Teilen bestehen: (1) *Mixed-Signal Hardware* (HW), wo die analoge Welt auf die digitale Welt trifft, (2) und *Software* (SW), die unsichtbare Schicht, die uns mit der physischen Realität verbindet. Heterogene SOC's gehören zu den am schnellsten wachsenden Marktsegmenten in der Elektronik- und Halbleiterindustrie. Getrieben durch Wachstumschancen in verschiedenen Anwendungsbereichen, passen viele Halbleiterhersteller ihre Strategie an und verlagern ihren Fokus von separaten *Integrierten Schaltkreisen* (ICs), die eine Funktion ausführen, hin zu einer stärker integrierten Lösung von *Radiofrequenz* (RF) und leistungsstarken *Analog/Mixed-Signal* (AMS) Designs. Aufgrund dieser Branchenverschiebung enthalten die meisten SOC's heute heterogene Elemente wie analoge Sensoren, Mixed-Signal-Wandler, digitale Prozessoren, die SW ausführen, und RF-Transceiver, die eng auf einem einzigen Die integriert sind. Während diese Verschiebung zu leistungsstarken, effizienten und platzsparenden Geräten geführt hat, z. B. Apple M1 SOC [6], hat sie den Aufwand für die Entwicklung und Verifizierung dieser hochkomplexen Geräte erheblich erhöht und gleichzeitig die erforderliche *Time-To-Market* (TTM) erreicht.

Die erste Herausforderung in diesem Zusammenhang ist die Abhängigkeit von HW und SW. Traditionell wurden HW und SW isoliert entwickelt und trafen erst in den späten Integrations- und Testphasen aufeinander. Infolgedessen bestand immer eine sequenzielle Abhängigkeit zwischen den Entwicklungsphasen von HW und SW, wie in Abb. 1.1a dargestellt. Daher konnte die SW erst ordnungsgemäß getestet werden, wenn die ersten Siliziumprototypen des SOC verfügbar waren. Insbesondere HW-abhängige SW wie Gerätetreiber und Low-Level-Kernel-Code

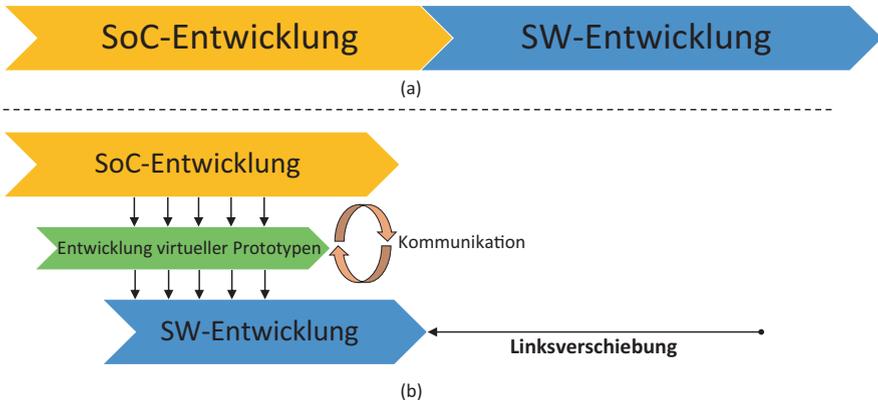


Abb. 1.1 Frühe SW-Entwicklung unter Nutzung des Shift-Left-Konzepts

konnten erst geschrieben werden, nachdem das Siliziumdesign abgeschlossen war. Eine Lösung zur Verringerung der TTM, die von der Industrie weitgehend übernommen wurde, besteht darin, von der vollständigen internen HW-Entwicklung abzurücken und stattdessen größere Mengen an vorverifiziertem Drittanbieter-*Geistigem Eigentum* (IP) zu verwenden. Dies ermöglicht es ihnen, sich stärker auf das *Alleinstellungsmerkmal* (USP) ihrer HW und SW zu konzentrieren. Mit der zunehmenden SW-Funktionalität und den komplexen Interaktionen zwischen verschiedenen HW-Komponenten hat die Designkomplexität um ein Vielfaches zugenommen, was die Designverifizierung von heterogenen SOC zu einer zunehmend entmutigenden Aufgabe macht.

Die zweite Herausforderung bei der Verifizierung von heterogenen SOC ist die langsame gemeinsame Simulation von *Register Transfer Level* (RTL) und SPICE (Simulation Program with Integrated Circuit Emphasis) Modellen für den digitalen und analogen/RF-Teil des SOC [9]. Traditionell war die Verifizierungsmethodik für analoge/RF-IPs ad hoc und diese IPs wurden immer von separaten Teams verifiziert. Sie wurde durch gerichtete Tests über Sweeps, Ecken und Monte-Carlo-Analysen gesteuert. Leider hat sich dies bis heute nicht viel geändert und schafft einen Engpass im Design- und Verifizierungsprozess. Auf der anderen Seite hatten digitale IPs formalisierte Verifizierungsmethoden, die von separaten Teams verwendet wurden. Dazu gehörten ausführbare Verifizierungspläne, erzeugung von beschränkt-zufälligen Stimuli [119], Testbench-Automatisierung, Assertions und Abdeckungsmetriken. Diese vorverifizierten analogen, RF- und digitalen IPs wurden später in einem größtenteils digitalen SOC-Design integriert und SW wurde darauf ausgeführt, um zu testen, ob alles wie erwartet funktioniert. Aufgrund der multifunktionalen Natur der heterogenen SOC sind jedoch insbesondere die analogen und RF-IPs sehr komplex geworden und enthalten erhebliche digitale Steuerlogik. Darüber hinaus hat die Interaktion von analogen/RF- und digitalen IPs, insbesondere wenn man die darauf laufende SW berücksichtigt,

erheblich zugenommen [77]. Daher sind die traditionellen Verifizierungsansätze nicht mehr ausreichend. Die gemeinsame Simulation, obwohl langsam, gilt immer noch als Goldstandard aufgrund ihrer hohen Genauigkeit und kann nicht ignoriert werden. Sie ist jedoch zu langsam für Chip-Level-Simulationen, es sei denn, sie wird äußerst selektiv eingesetzt.

Drittens ist die Designraum- und Architekturerkundung eingeschränkt. Angesichts der Systemanforderungen ist die Suche nach der optimalen Systemkonfiguration eine mühsame Aufgabe. Jede Parameteränderung, egal wie gering, kann den Designer ein paar Tage in der Designphase kosten. Daher ist die Erkundung sehr selektiv.

Schließlich hat die weit verbreitete Praxis der Integration von Drittanbieter-IPs die modernen SOCs berüchtigt unsicher gemacht, da sie als Fahrzeug für Boshaftigkeit genutzt werden können. Zum Beispiel der jüngste Sicherheitskompromiss von SOCs, die Intels Mikroprozessor-IPs und Actel ProASIC3-IPs verwenden. Die ersteren wurden durch die „*Meltdown und Spectre*“ Schwachstellen [79, 89] ausgenutzt und die letzteren durch die JTAG-Schwachstelle [108]. Dies unterstreicht die Tatsache, dass legitime kommerzielle Off-the-Shelf-IPs Benutzerinformationen auf eine Weise manipulieren oder bei der Manipulation helfen können, die ihre Benutzer weder erwarten noch schätzen. Darüber hinaus kann ein SOC nach der Fertigung nicht gepatcht werden und erfordert daher ein neues Design.

Nach einer Schätzung kann die traditionelle funktionale Verifizierung von heterogenen SOCs bis zu 70 % der Projektzeit in Anspruch nehmen. Dennoch resultiert sie in vielen Neudrehungen, die hauptsächlich aus Mixed-Signal-Verifizierungsproblemen resultieren [77]. Dies geschieht hauptsächlich aufgrund schwächerer Verifizierungsmethoden, Tools oder Bibliotheken, die nicht in der Lage sind, *vermeidbare* HW- und SW-Fehler frühzeitig in der Designphase zu erkennen. Daher sind Tools, Bibliotheken und komplexe AMS-Verifizierungsmethoden erforderlich, die (1) die *HW/SW-serielle Abhängigkeit* im Design und der Verifizierung von heterogenen SOCs entfernen und stattdessen HW/SW-Co-Design ermöglichen (2) die *Simulationgeschwindigkeit erhöhen* und gleichzeitig eine gute Genauigkeit bieten (3) eine *schnelle Designraum- und Architekturerkundung* ermöglichen (4) eine *frühe Sicherheitsvalidierung* bieten

Zusammenfassend lässt sich sagen, dass das erfolgreiche Co-Design von sicheren multidisziplinären heterogenen SOCs, die enge Interaktionen zwischen HW/SW-Systemen und ihrer analogen physischen Umgebung aufweisen, eine Herausforderung darstellt. Da die TTM kürzer wird, wird die Fähigkeit, komplexe heterogene SOCs zu modellieren und zu simulieren, bei denen digitale HW/SW funktional mit AMS-IPs verflochten ist, d. h. RF-Schnittstellen, Leistungselektronik, Sensoren und Aktuatoren, immer wichtiger. Wenn solche Gesamtsystem- und Architekturebenenmodelle so früh wie möglich im Designzyklus verfügbar sind, werden die Architekturerkundung und Designprobleme sowie Sicherheitslücken drastisch reduziert.