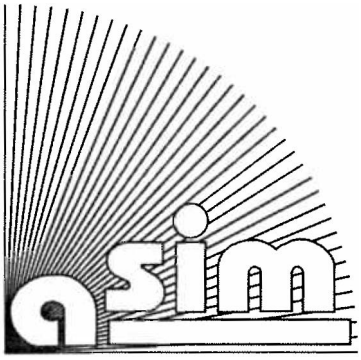
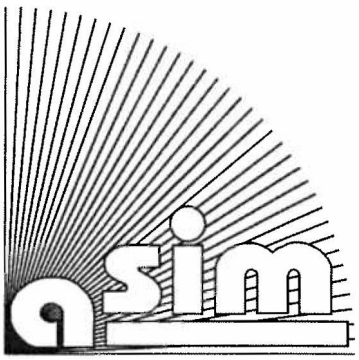


Praxisforum & Poster

13. ASIM – Symposium Simulationstechnik Weimar, September 1999

Georg Hohmann (Hrsg.)

ARGESIM Report AR 14 ASIM
Mitteilung AM 66B
ISBN print 978-3-901608-14-8
ISBN ebook 978-3-901608-86-5
DOI 10.11128/arep.14



Titlel: Praxisforum & Poster - 13. ASIM-Symposium Simulationstechnik
ASIM SST 1999, Weimar, September 1999

Herausgeber: Georg Hohmann

Reihe „ARGESIM Reports"
Reihenherausgeber: F. Breitenecker

ARGESIM Report AR 14

ASIM Mitteilung AM 66B

ISBN print 978-3-901608-14-8, 1999

ISBN ebook 978-3-901608-86-5, 2021

DOI 10.11128/arep.14

ARGE Simualtion News (ARGESIM)
Technische Universität Wien
Wiedner Hauptstraße 8-10
A-1040 Wien, Österreich
Tel.: +43-1-58801-11452
Fax: +43-1-58801-11499
Email: info@argesim.org
WWW: http://www.argesim.org

Inhaltsverzeichnis

Praxisforum „Elektroniksimulation“ I

Praxisforum „Elektroniksimulation“ II

Postersession I: Anwendungen

Postersession II: Simulationsmethoden

Nachtrag zum Tagungsband

Modellbildung – Datenmodelle/Internet

Dipl.-Ing. Birgit Knorr, Dipl.-Ing. Falk Nerger, Dr.-Ing. Lutz Zacharias, SIMEC GmbH & Co KG:

Effektive Modellierung und Simulation von leistungselektronischen Schaltungen

Die Entwicklung von leistungselektronischen Komponenten ist zunehmend durch verschiedene Einflußfaktoren gekennzeichnet, die einen Einsatz von Simulationswerkzeugen unumgänglich machen. Dabei geht es nicht nur um die klassische Simulation der Schaltung die in einem Stromversorgungsgerät steckt, sondern um wesentlich komplexere Zusammenhänge. Bedingt durch immer höhere Schaltfrequenzen, neue Bauelemente und Steuerverfahren, verschärfte EMV-Richtlinien, zunehmende ISO Zertifizierung etc. sind die Entwickler immer mehr gezwungen, komplexere Zusammenhänge bei gleichzeitigem tieferem Eindringen in Detailprobleme zu bewältigen. Hinzu kommt ein enormer Kostendruck, verursacht u. a. durch eine Internationalisierung des Wettbewerbs. Es wird an allen Ecken und Enden gespart, in vielen Unternehmen ist eine drastische Reduzierung des Personals zu verzeichnen, die selbst vor Entwicklungsabteilungen nicht Halt macht. Somit ruht auch hier ein bedeutendes Potential für den Einsatz moderner Entwicklungswerkzeuge. Ziel dieses Beitrages ist es, Möglichkeiten zum rechnergestützten Leistungselektronik-Design aufzuzeigen und einen Einblick in die Bandbreite möglicher Simulationsverfahren zu vermitteln.

Ausgangspunkt Schaltungssimulation

In vielen Unternehmen werden bereits konventionelle Schaltungssimulatoren eingesetzt. Die meisten dieser Werkzeuge beruhen auf dem an der Berkley University entwickelten Simulator SPICE. Sie sind leistungsfähige Werkzeuge für die Untersuchung analoger (integrierter) Schaltungen und sind speziell hierfür ausgelegt. Diese Ausrichtung birgt jedoch gerade für hochfrequent schaltende Systeme die Gefahr numerischer Instabilitäten und von Konvergenzproblemen in sich. Ein weiterer Nachteil entsteht aus der Betrachtung zu untersuchenden Systemstruktur. Läßt man Konvergenz und Numerik aus der Betrachtung heraus, stellt diese Klasse von Simulatoren lediglich eine Beschreibungssprache - elektrische Ersatzschaltbilder - zur Verfügung und deckt damit a priori nur die eigentliche Schaltungstopologie ab. Alle anderen Modellkomponenten müssen als Schaltung nachgebildet werden. Ein analoger PID-Regler könnte beispielweise mit Hilfe eines beschalteten Operationsverstärkers modelliert werden, bei mechanischen Komponenten (Motor mit mechanischer Belastung) ist die Ersatzschaltung schon wesentlich schwieriger zu finden. Diese Umwandlungen und Analogiebeziehungen bergen eine ganze Reihe von Fehlerquellen in sich und verursachen einen sehr großen Modellierungsaufwand.
Ein weiterer wesentlicher Punkt in Zusammenhang mit der Schaltungssimulation ist die Verwendung geeigneter Bauteilmodelle, insbesondere von Halbleiterbauelementen. Diese Elemente entscheiden oft über eine hinreichende Genauigkeit der Simulation. Allerdings sollten auch in diesem Zusammenhang einer Simulation Betrachtungen grundlegender Natur vorausgehen. Bei genauerem Hinsehen kann derselbe Grundsatz angewandt werden, wie in der Meßtechnik. Die Simulation sollte so genau wie nötig, nicht so genau wie möglich erfolgen. Letztendlich muß jeder Anwender selber entscheiden, welche Ziele er mit der Simulation erreichen will und die dazu passende Modellebene auswählen. Gerade bei der Entwicklung von Stromversorgungsgeräten läßt sich mittlerweile ein Trend zur Anwendung verschiedener Abstraktionsebenen erkennen. Das bedeutet, ein Netzteil wird natürlich mit seiner Schaltung aber auch unter Berücksichtigung umgebender Modellkomponenten betrachtet. Die Erfahrung hat gezeigt, daß zur Systemsimulation klassische Modellansätze, die das detaillierte dynamische Verhalten eines Halbleiterschalters wiedergeben nur bedingt einsetzbar sind. Sie erfordern sehr kleine Rechenschritte, um die internen dynamischen Vorgänge abzubilden. Demgegenüber steht jedoch ein verhältnismäßig großer Gesamtsimulationszeitraum, in dem Lastschwankungen, Veränderungen der Versorgungsspannung, Batterieladungsänderungen, regelungstechnisches Verhalten bis hin zu Power-Managementstrategien untersucht werden sollen.
Deshalb ist es überlegenswert, andere Modellierungsansätze für Halbleitermodelle zu verwenden. In SIMPLORER, einem für komplexe Systemuntersuchungen entwickelten Simulator wurde eine Abb. 1 entsprechende 3-Ebenen Modellierungsphilosophie implementiert. Der Nutzer kann je nach Simulationsziel entscheiden, ob er ideale Bauelemente (ideale Schalter mit unendlich großem oder kleinem Wider-

stand), statische Bauelemente (dem realen Verhalten entsprechende R_{ON} und R_{OFF} plus eine dem statischen Übertragungsverhalten nachgebildete Kommutierungsfunktion) oder dynamische Halbleiterbauelemente einsetzt. Ideale Schalter und statische Halbleitermodelle werden dabei einfach durch ein logisches Signal angesteuert, eine aufwendige Modellierung der Ansteuerschaltung entfällt.
Es ist leicht nachzuvollziehen, daß die Verwendung idealer oder statischer Modelle wesentliche Geschwindigkeitsvorteile bei der Untersuchung des Systemverhaltens mit sich bringt. Die Rechenschrittweiten werden im wesentlichen durch die Schaltfrequenzen bestimmt. Die Erfahrung hat gezeigt, daß hier eine Steigerung der Rechengeschwindigkeit von bis zu Faktor 10 erreicht werden kann, weil einfach weniger Schritte berechnet werden müssen und das Simulationsmodell weitaus weniger Nichtlinearitäten beinhaltet.

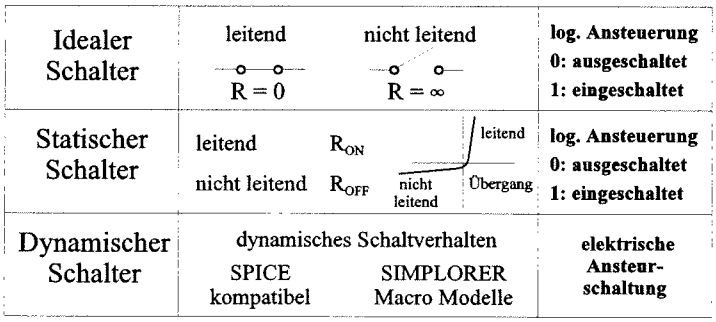


Abb. 1 3-Ebenen - Modellierungsphilosophie für Halbleiterschalter

Ein PWM-Baustein elegant nachgebildet

Neben der schaltungstechnischen Beschreibung ist auch die Nachbildung der Ansteuerschaltung oder gar eines kompletten Steuergerätes von Interesse. Typische Vertreter sind hier PWM-Bausteine oder auch Hysteresecontroller. Weiterhin sind Einflüsse von Nutzerinteraktionen zu untersuchen. All diesen Systemkomponenten ist gemeinsam, daß sie diskontinuierlich arbeiten. Auch hier ergibt sich wieder die Möglichkeit, Schaltungen und Steuergeräte auf Basis kombiniert analoger und digitaler Schaltungen nachzubilden, allerdings dürfte hier der Modellierungsaufwand für den reinen Anwender solcher Bausteine in jedem Falle zu hoch sein. Selbst bei Kenntnis des internen Aufbaus eines solchen diskontinuierlich arbeitenden Systems ist es in diesem Zusammenhang wenig sinnvoll, auf dieser Ebene zu modellieren. Man kann, wie auch im Bereich der Regelungen davon ausgehen, daß die angebotenen Bausteine Ihre Funktionalität bereitstellen, also lediglich die Nachbildungen ihres Verhaltens erforderlich ist.
Somit stellt sich als erstes die Frage nach einer geeigneten Beschreibungsform für diskontinuierliche Prozesse. Betrachtet man den Bereich der Steuerungstechnik, ist festzustellen, daß dort diskontinuierliche Prozesse seit langen mit Zustandsgraphen, Ablaufplänen oder Petri-Netzten beschrieben werden. Das heißt, man zerlegt einen realen Prozeß in signifikante Teilzustände. In diesen wird so lange verharrt, bis eine Übergangsbedingung zum Erreichen des nächsten Zustandes eintritt. Ausgehend von diesem Prinzip lassen sich diskontinuierliche Prozesse sehr elegant nachbilden. Solange ein Zustand aktiv ist (man spricht auch von „markiert"), gelten die für diesen Zustand definierten Eigenschaften (Aktivitäten). Je nach Implementation stehen verschiedene Aktionstypen zur Verfügung, vom einfachen Setzen einer Variable bis hin zu mathematischen und logischen Ausdrücken und die Definition von Timern. Der Übergang von einem Zustand zum anderen wird durch eine Transition (Übergangsbedingung) gesteuert. Diese wird als logischer Ausdruck interpretiert und überträgt die Aktivität auf den/die nachfolgenden Zustände, wenn das Resultat logisch wahr ist. Beim Übergang werden alle Vorgängerzustände deaktiviert. Eine charakteristische Eigenschaft der Zustandsgraphen besteht im verzögerungsfreien Übergang von einem Zustand zum anderen. Zeitliche Übergangsbedingungen (Timer) müssen also explizit modelliert werden. Mit Hilfe von Zustandsgraphen lassen sich, wie mit einer Programmiersprache, Abläufe graphisch darstellen, ohne eine spezielle Sprache und deren Syntax erlernen zu müssen. Ein weiters Charakteristikum der Zustandsgraphen ist beispielsweise, daß der Übergang nur dann erfolgen kann, wenn alle vorgelagerten Zustände aktiv *und* die Übergangsbedingung logisch *wahr* sind. Wenn einer der Eingangszustände noch nicht aktiviert (markiert) ist, kann auch bei erfüllter Übergangsbedin-

gung der Übergang nicht erfolgen. Mit dieser Definition kann man parallele Prozesse synchronisieren. Aber auch konventionelle Programmierbefehle lassen sich mit Zustandsgraphen nachbilden, wie zum Beispiel Schleifen, Ketten und Alternativen (Abb. 2).

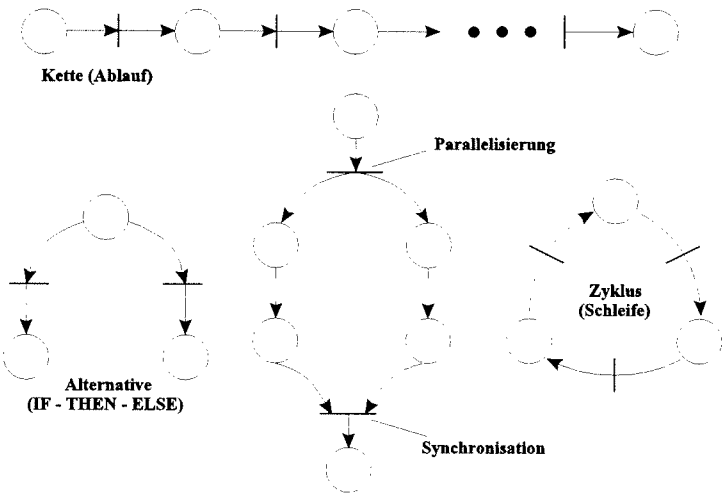


Abb. 2 Realisierbare Programmierstrukturen mit Hilfe von Zustandsgraphen

Zur Pulsweitenmodulation werden eine ganze Reihe von Bausteinen angeboten, die im wesentlichen die gleiche Funktion haben. Ein Eingangssignal wird mit einem hochfrequenten Referenzsignal (Dreieck- oder Sägezahnfunktion) verglichen und stellt ein entsprechendes Ausgangssignal mit variabler Pulslänge bereit. Dieses Verhalten kann man natürlich auf der Ebene der im Baustein realisierten Schaltung modellieren, allerdings wird man dabei auf verschiedene Probleme stoßen. Einerseits ist die innere Struktur nicht immer bekannt, und wenn, so fehlen doch mit hoher Wahrscheinlichkeit die entsprechenden Bauteilparameter. Aber selbst wenn diese Daten bekannt sind, ist am Ende der Modellierungsaufwand sehr hoch, um das Schaltungsverhalten wirklichkeitsnah widerzuspiegeln. Schließlich wird die Rechenzeit inakzeptabel hoch, insbesondere im Zusammenhang mit Systemsimulationen, die einen längeren Zeithorizont abdecken.
Deshalb ist es sehr viel einfacher und auch effizienter, wenn man die Funktionalität des Bausteins mit Hilfe von Zustandsgraphen darstellt. Zerlegt man das System in signifikante Zustände, erhält man einen simplen Graphen mit zwei Zustandselementen und zwei Übergangsbedingungen. In den Zuständen sind die jeweiligen Werte für das Ausgangssignal bestimmt, im einfachsten Falle ein logisches Signal mit 0 oder 1 Pegel. Die Übergangsbedingungen können einfach mit Hilfe eines logischen Vergleiches beschrieben werden (siehe Abb. 3).

Zustandsgraphen als Prozeßsteuerungsmodelle

Als ein einfaches Beispiel sei die Simulation eines Einphasenwechselrichters betrachtet. Ziel der Simulation ist es, mit Hilfe eines PWM-Bausteines die Halbleiterschalter des Wechselrichters so zu steuern, daß sich in der Lastinduktivität ein sinusförmiger Strom einstellt. Die Simulation wurde mit SIMPLORER durchgeführt, weil in diesem Programmpaket sowohl ein Schaltungssimulator als auch alle Möglichkeiten der Modellierung mit Hilfe von Zustandsgraphen simultan vorhanden sind.

Abb. 3 zeigt die Modellierung der Pulsweitenmodulation (Zustandsgraf). In den Zuständen werden die Ansteuersignale für die Halbleiterschalter der Brücke gesetzt, der Laststrom wird an den Übergangsbedingungen ausgewertet. Da im vorliegenden Fall die Systemsimulation im Vordergrund steht, wurden die Halbleiterbauelemente einfach als ideale Schalter modelliert. In Abb. 3 sind ebenfalls modellierte Schaltung und Ergebnis der Simulation gleichzeitig dargestellt. Dies kann bei SIMPLORER direkt in der Modellbeschreibung (on sheet) visualisiert werden und wird immanenter Bestandteil des Modells. Damit lassen sich auch später aus der zugrunde liegenden Datenbank Simulationsergebnisse reproduzieren, ohne eine Simulation wiederholen zu müssen.

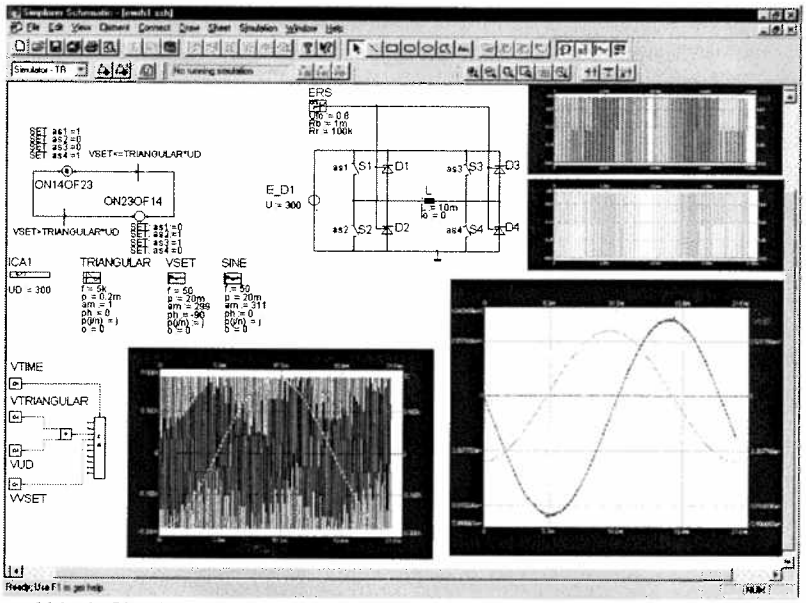


Abb. 3 Simulationsbeispiel Einphasenwechselrichter mit Ergebnisdarstellung

Zustandsgraphen als Meßsystem und Simulationssteuereinrichtung

Die Anwendungsbreite von Zustandsgraphen ist nicht nur auf die reine Prozeßdatenverarbeitung und Prozeßbeeinflussung beschränkt. Darüber hinaus können mit Zustandsgraphen noch während des Simulationsablaufes vom Nutzer definierte Kennwerte ermittelt werden. Ziel dieser Anwendung ist es, unter Ausnutzung der Tatsache, daß die Berechnung von Kennwerten im Vergleich zur Berechnung des Systemmodells mit numerischen Verfahren kaum Rechenzeit beansprucht, parallel zur Simulation Kennwerte und integrale Größen eines Modells zu berechnen. Die Übergangsbedingung dient dabei zur Auswertung bzw. Erfassung der transienten Größen der Simulation, in den Zuständen werden dann mathematische Operationen ausgeführt, die zur Bildung des Kennwertes erforderlich sind. Der Nutzen liegt auf der Hand: Kennwerte sind bei Simulationsende sofort verfügbar und müssen nicht erst aufwendig im Postprozessing errechnet werden.
Wenn man den Gedanke der Steuerung mit Zustandsgraphen weiter führt und die Simulation selbst als Prozeß begreift, kommt man zu einem weiteren Anwendungsgebiet der Zustandsgraphen. Der Simulationsprozeß kann, wenn dies im Simulator vorgesehen ist, genau wie jeder andere Prozeß überwacht und beeinflußt werden. So ist es denkbar, einen Prozeß bis zum eingeschwungenen Zustand mit relativ großer Schrittweite zu berechnen, um dann später im eingeschwungen Zustand eine entsprechend kleinere Schrittweite zu wählen (besonders interessant bei der Simulation von Schaltnetzteilen). Eine andere Anwendung ist das Beenden der Simulation in Abhängigkeit verschiedener erreichter Zustände, beispielweise bei Erreichen eines bestimmten Schwellwertes. Darüber hinaus können zustandsgesteuert Informationen gespeichert und anschließend einer Weiterverarbeitung zugeführt werden.

Über den Tellerrand der Schaltung hinausgeblickt - die Regelung

Das gedankliche Konzept für beliebige regelungstechnische Aufgaben wird im allgemeinen mit Hilfe von Blockdiagrammen erstellt. Dabei bleibt die hardwareseitige Realisierung des Regelgerätes zunächst im Hintergrund. Häufig wird heute auf eine Vielzahl fertiger Reglerbausteine zurückgegriffen, z. B. Schaltregler-IC. Bei diesen Bausteinen kann man davon ausgehen, daß deren Funktionsweise gesichert ist. Deshalb ist eine Konzentration auf das für die Untersuchung des Gesamtsystems maßgebende systemtechnische Verhalten ausreichend.
Natürlich besteht auch bei konventionellen Schaltungssimulatoren die Möglichkeit, diese Funktionsblöcke beispielsweise mit Hilfe von beschalteten Operationsverstärkern oder anderen Ersatzschaltungen nachzubilden. Die Problematik ist dann jedoch dieselbe wie bei der Untersuchung des Schaltverhaltens

der Halbleiterschalter oder der Untersuchung von Steueralgorithmen mit detaillierten Schaltungsmodellen. Die exakte elektrische Nachbildung erfordert die Berücksichtigung vieler Modellkomponenten, die mit kleinen Zeitkonstanten ablaufen und somit eine entsprechend kleine Simulationsschrittweite erfordern.
Deshalb wurden in SIMPLORER außer der Schaltungssimulation und zusätzlich zur Zustandsgraphenbeschreibung die Blockdiagramme implementiert. Sie beruhen auf der allgemein üblichen Beschreibungsform aus dem Bereich der Regelungstechnik. Blockdiagramme gestatten eine elegante Modellierung von systemtheoretischen Zusammenhängen. Darüber hinaus erlauben sie bei geeigneter Implementation die Analyse digitaler und gemischt analog/digitaler Regelungen, weil die Blöcke mit verschiedenen (jeweils konstanten) Abtastzeiten berechnet werden können. Die Kombination der regelungs- und schaltungstechnischen Beschreibungen in Verbindung mit den verschiedenen Abstraktionsebenen bei der Halbleitermodellierung machen es möglich, den Einfluß von Struktur- und Parameteränderungen im Regler zu untersuchen und vor allem die Wechselwirkungen zwischen den einzelnen Systemkomponenten schnell und effizient zu analysieren. Dabei ist es sogar möglich, diese Parametervariationen durch einen automatisierten Optimierungsalgorithmus zu erzeugen. Mit einem solchen Werkzeug lassen sich mittlerweile nahezu beliebig komplexe Systeme optimieren, was eine weitere beträchtliche Einsparung an Experimentalaufwand impliziert.

Als Beispiel soll eine Schaltung für eine unterbrechungsfreie Stromversorgung untersucht werden. Eines der wesentlichen Ziele bei der Entwicklung moderner Stromversorgungsgeräte ist die Vermeidung von Oberschwingungen im Netzstrom. Hierfür werden häufig Powerfactor Regler eingesetzt (Abb. 4).

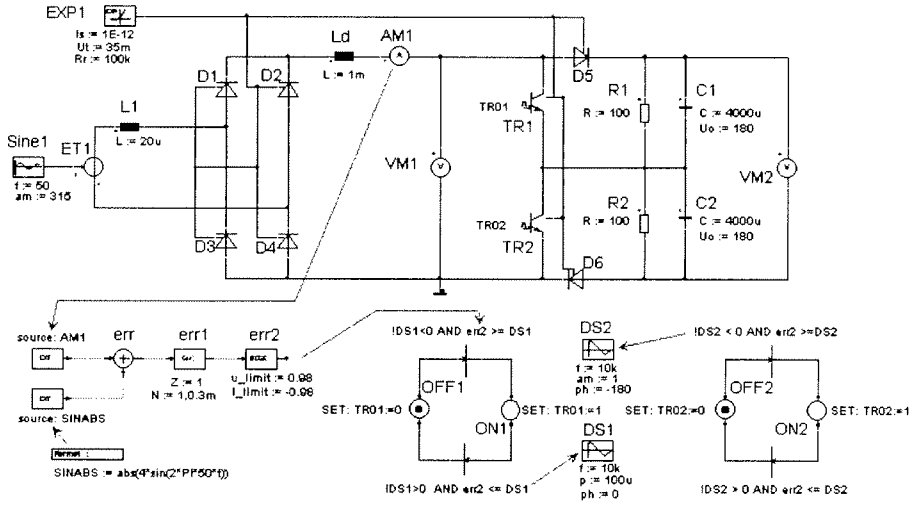


Abb. 4 USV-Schaltung - Simulation mit verschiedenen Beschreibungsebenen

Das zu untersuchende System besteht aus Komponenten aller drei SIMPLORER-Beschreibungsformen. Im Schaltungsteil werden die Transistoren als statische Halbleiterschalter modelliert, um Rechenzeit bei der Untersuchung des im Vordergrund stehenden regelungstechnischen Verhaltens zu sparen. Beide Transistoren werden 180° phasenverschoben angesteuert. Der aktuelle Strom am Ausgang der Gleichrichterbrücke wird mit Hilfe eines Amperemeters gemessen und an die Regeleinrichtung als Istwert übertragen. Der Sollwert wird als gleichgerichtete Sinusfunktion mit Hilfe eines Formelelementes definiert. Der erforderliche Regler wird mittels Laplace-Übertragungsfunktion inklusive Stellgrößenbegrenzung modelliert. Das Resultat des Soll/Istwertvergleiches steht als Vergleichssignal für die beiden PWM-Bausteine zur Verfügung. Beide PWM-Bausteine liefern dann das jeweilige Ansteuersignal für die Transistoren. Somit wird das gesamte System mit geschlossener Regelschleife modelliert, nur daß der Prozeß nicht, wie sonst in der Regelungstechnik üblich, als Übertragungsfunktion G(s) beschrieben wird, sondern als reale Schaltung.

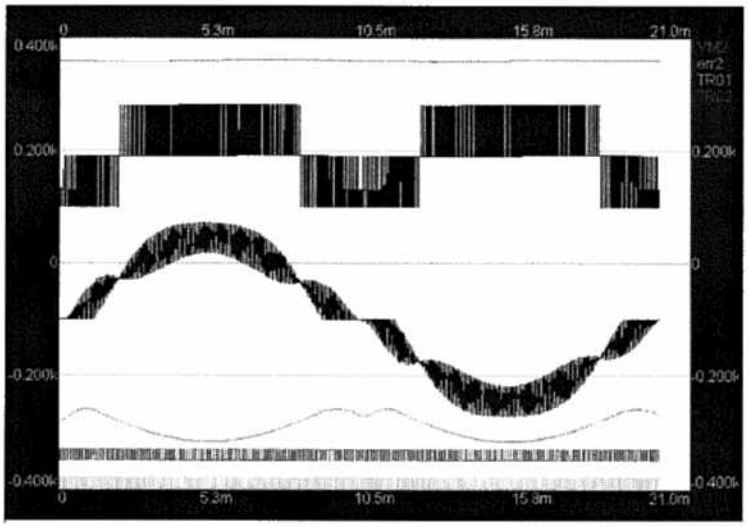


Abb. 5 USV-Schaltung - Simulationsergebnis mit sinusförmigem Netzstrom

Mit der Beschreibung der Regler auf der Ebene der Blockdiagramme ist die Anwendung dieser Beschreibungssprache jedoch noch nicht beendet. Eine C/C++Programmierschnittstelle läßt die Simulation auch auf die Untersuchung µ-Controller basierter Steuer- und Regeleinrichtungen ausdehnen. Eine solche Schnittstelle erlaubt die Einbindung von nutzerspezifischen Routinen.

Neue Anforderungen erfordern weitergehende Untersuchungen

Die Untersuchung von leistungselektronischen Komponenten ist aufgrund der gewachsenen Komplexität solcher Anordnungen nicht mehr auf die reine Schaltungssimulation reduzierbar. Die Schaltung kann nur noch in Zusammenhang mit ihrer Umgebung betrachtet werden. Dabei sind eine ganze Reihe von Wechselwirkungen zu berücksichtigen. Deshalb ist eine Erweiterung der klassischen Schaltungssimulation um weitere Verfahren notwendig. Hier helfen ingenieurgemäße Beschreibungsformen auf abstrakterer Ebene, um wesentlich schneller und komfortabler an das gewünschte Simulationsziel zu gelangen, als es eine Modellierung ausschließlich auf Schaltungsebene erlaubt. Mit zunehmenden Schaltfrequenzen ist jedoch auch eine detailliertere Untersuchung des Stromversorgungssystems in seiner Gesamtheit erforderlich. Insbesondere hinsichtlich der Untersuchung von EMV und eines optimal kostensparenden Komponentendesigns sind weitere Simulationsverfahren notwendig. Immer mehr in den Blickpunkt rücken dabei Softwarepakete zur Bestimmung parasitärer Effekte, die durch Leiterführung auf Leiterplatten oder durch Verbindungen zwischen diskreten Bauelementen entstehen. Hier sind mittlerweile leistungsfähige Produkte verfügbar, die aus einer gegebenen Geometrie- und Materialdefinition die Ersatzschaltung der parasitären Induktivitäten, Widerstände und Gegeninduktivitäten berechnen können. Somit wird die Gesamtsystemsimulation genauer und es besteht die Möglichkeit, entsprechende Ableitungen hinsichtlich zu erwartender Oberwellen und Schaltspitzen zu treffen. Das gleiche zunehmende Interesse gilt adäquaten FEM (Finite-Elemente-Methode) Programmen. Einerseits sind hier Untersuchungen in Richtung mechanischer Beanspruchungen und thermischer Effekte möglich, aber auch die Auslegung elektromagnetischer Wandler ist auf dieser Modellierungsebene denkbar. Im Bereich der Stromversorgungstechnik betrifft das vor allem Übertrager oder Transformatoren.
Zusammenfassend kann festgestellt werden, daß der Stand der Technik eine Vielzahl von Simulationsverfahren bereitstellt, die dem Ingenieur das Design moderner und marktfähiger Produkte bedeutend erleichtern. Sie helfen, das Produktdesign besser zu durchdringen, optimale Lösungen zu finden und in vielen Bereichen Kosten zu senken, insbesondere durch die Einsparung von Experimentalaufwand. Leistungsfähige Personalcomputer ermöglichen es dabei, diese Simulationsprogramme direkt an den Arbeitsplatz des Ingenieurs zu bringen und somit als ständig präsentes Werkzeug bereitzuhalten.

Einsatz von *Mathematica* beim Entwurf von Schaltnetzteilen

Horst Edel

Jeder Entwurf eines Schaltnetzteiles beinhaltet die Dimensionierung einer Regelschleife, die nicht nur dafür sorgen soll, dass eine Ausgangsgröße konstant bleibt, sondern auch gewisse Dynamikeigenschaften sicherstellen muss. Beispielsweise soll eine niederfrequente Störung am Eingang mit einer festgelegten Dämpfung am Ausgang erscheinen. Bei dynamischen Laständerungen muss ein definiertes Lastsprungverhalten eingehalten werden und schließlich soll das System im gesamten Last- und Eingangsspannungsbereich stabil arbeiten. Mit Hilfe von Linearmodellen und Mathematikprogrammen mit symbolischen Berechnungsmöglichkeiten lässt sich schon vor dem ersten Brettaufbau ein Schaltnetzteil so dimensionieren, dass es die Dynamikforderungen erfüllt.
Wenn die Ausgangsgröße die Ausgangsspannung u_a ist, könnte die Struktur des gesamten Schaltnetzteiles folgendes Aussehen haben

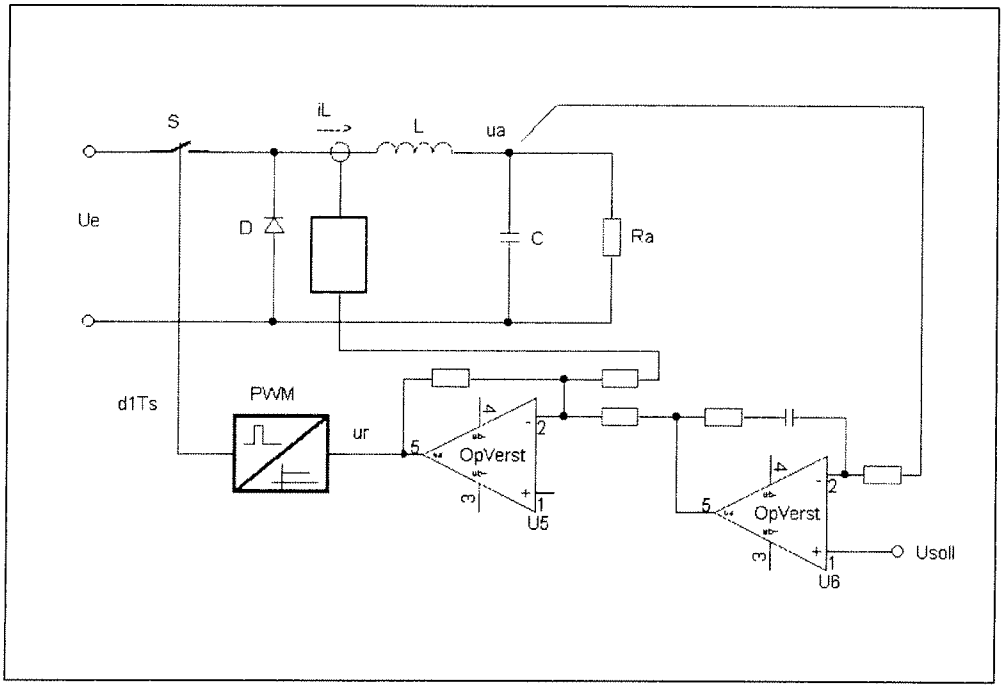


Bild 1

Bild 1 zeigt einen Abwärtswandler, der von einer Eingangsspannung U_e gespeist wird. Das Leistungteil besteht aus dem Schalter S, der Diode D und dem LC Ausgangsfilter. Der Widerstand R_a stellt die zu versorgende Last dar. Zur Regelung soll ein zweiläufiger Regelkreis verwendet werden, der die beiden Zustandsgrößen Drosselstrom i_L und Kondensatorspannung u_a rückkoppelt. Der Pulsweitenmodulator (PWM) erzeugt aus einer analogen Regelgröße u_r eine pulsweitengesteuerte Impulsfolge $d_1 T_s$, mit welcher der Schalter S angesteuert wird.

Die Linearisierung der Strecke führt zu folgender Zustandsdarstellung im kontinuierlichen, also nichtlückenden Betrieb:

$$\frac{dI_L}{dt} = -\frac{1}{L} u_a + \frac{d_1}{L} u_e$$
$$\frac{du_C}{dt} = \frac{1}{C} i_L - \frac{1}{R_a C} u_a$$

Hierbei sind L, C und R_a die konstanten Streckenelemente. Die Größen i_L, u_a, u_e und d_1 sind Mischgrößen, die sich aus Gleich- und Wechselanteilen zusammensetzen:

$i_L = I_L + \hat{i_L}$, $u_a = U_a + \hat{u_a}$, $u_e = U_e + \hat{u_e}$ und $d_1 = D_1 + \hat{d_1}$

Aus Bild 1 lässt sich die Reglerstruktur ablesen. Der Drosselstrom i_L wird proportional rückgekoppelt. Die Rückführung der Ausgangsgröße u_a geschieht über einen PI Regler, das heißt sie hat einen proportionalen und einen integrierenden Anteil. Damit ergibt sich folgendes Strukturbild der Regelschleife:

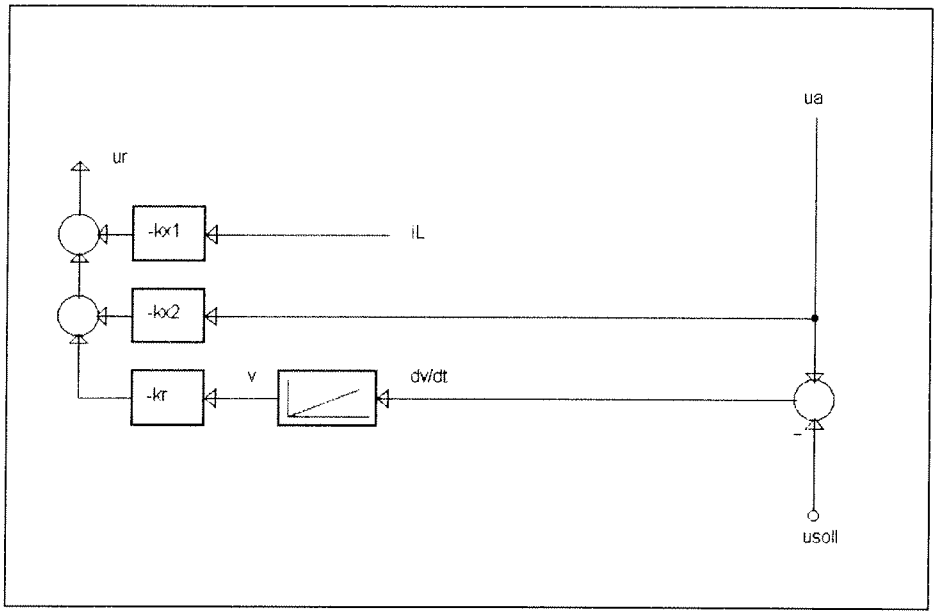


Bild 2

Aus Bild 2 kann man unmittelbar ablesen: $u_r = -k_{x1}\, i_L - k_{x2}\, u_a - k_r\, v$

Nimmt man die Ausgangsgröße v des Integrierers als zusätzliche Zustandsvariable, so ergibt sich folgendes Differentialgleichungssystem:

$$f1 = \frac{d\, i_L}{dt} = -\frac{1}{L} u_a + \frac{d_1}{L} u_e$$
$$f2 = \frac{d\, u_a}{dt} = \frac{1}{C} i_L - \frac{1}{R_a C} u_a + \frac{1}{C} i_z$$
$$f3 = \frac{d\, v}{dt} = u_a - u_{soll}$$

Dieses Differentialgleichungssystem kann nun von *Mathematica* bearbeitet werden. Um die Zustandsdarstellung des Gesamtsystems zu erhalten, müssen die drei Gleichungen f1, f2 und f3 partiell nach den Variablen $\hat{i}_L$, $\hat{u}_a$, $\hat{u}_e$, $\hat{i}_z$, $\hat{u}_{soll}$, $\hat{v}$ und $\hat{d}_1$ differenziert werden:

```
Aa = {{∂IL f1, ∂Ua f1, ∂V f1}, {∂IL f2, ∂Ua f2, ∂V f2}, {∂IL f3, ∂Ua f3, ∂V f3}}
b = {{∂Ue f1}, {∂Ue f2}, {∂Ue f3}}
c = {{∂D1 f1}, {∂D1 f2}, {∂D1 f3}}
d = {{∂Iz f1}, {∂Iz f2}, {∂Iz f3}}
ea = {{∂Usoll f1}, {∂Usoll f2}, {∂Usoll f3}}
d3 = {{∂IL Dd3, ∂Ua Dd3, ∂V Dd3}}
```

Zusätzlich muss zur Matrix Aa der Rückkoppelvektor kx addiert werden:

```
kx = {{kx1, kx2, kr}};          A = Aa - c . kx
```

Damit ergibt sich die Zustandsdarstellung des Systems:

$\frac{d\hat{\underline{x}}}{dt} = \underline{A}\,\hat{\underline{x}} + \underline{b}\,\hat{u}_e + \underline{c}\,\hat{u}_f + \underline{d}\,\hat{i}_z + \underline{e}\,\hat{u}_{soll}$; mit Zustandsvektor $\hat{\underline{x}} = \{\hat{i}_L, \hat{u}_a\}$

$$\underline{A} = \left\{\left\{-\frac{kx1\,Ue}{L}, -\frac{1 + kx2\,Ue}{L}, -\frac{kr\,Ue}{L}\right\}, \left\{\frac{1}{C}, -\frac{1}{C\,Ra}, 0\right\}, \{0, 1, 0\}\right\}$$

$$\underline{b} = \left\{\left\{\frac{D1}{L}\right\}, \{0\}, \{0\}\right\}$$

$$\underline{c} = \left\{\left\{\frac{Ue}{L}\right\}, \{0\}, \{0\}\right\}$$

$$\underline{d} = \left\{\{0\}, \left\{\frac{1}{C}\right\}, \{0\}\right\}$$

$$\underline{e} = \left\{\left\{\frac{ks\,Ue}{L}\right\}, \{0\}, \{-1\}\right\}$$

Die statischen Endwerte erhält man, wenn die beiden Gleichungen f1 und f2 gleich Null gesetzt werden und nach Ua bzw. IL aufgelöst wird:

```
Solve[f1 == 0, Ua]      Solve[f2 == 0, IL]
```

$$\{\{Ua \to D1\,Ue\}\} \qquad \left\{\left\{IL \to -\frac{Iz\,Ra - Ua}{Ra}\right\}\right\}$$

Damit ergibt sich folgendes Strukturbild des Gesamtsystems:

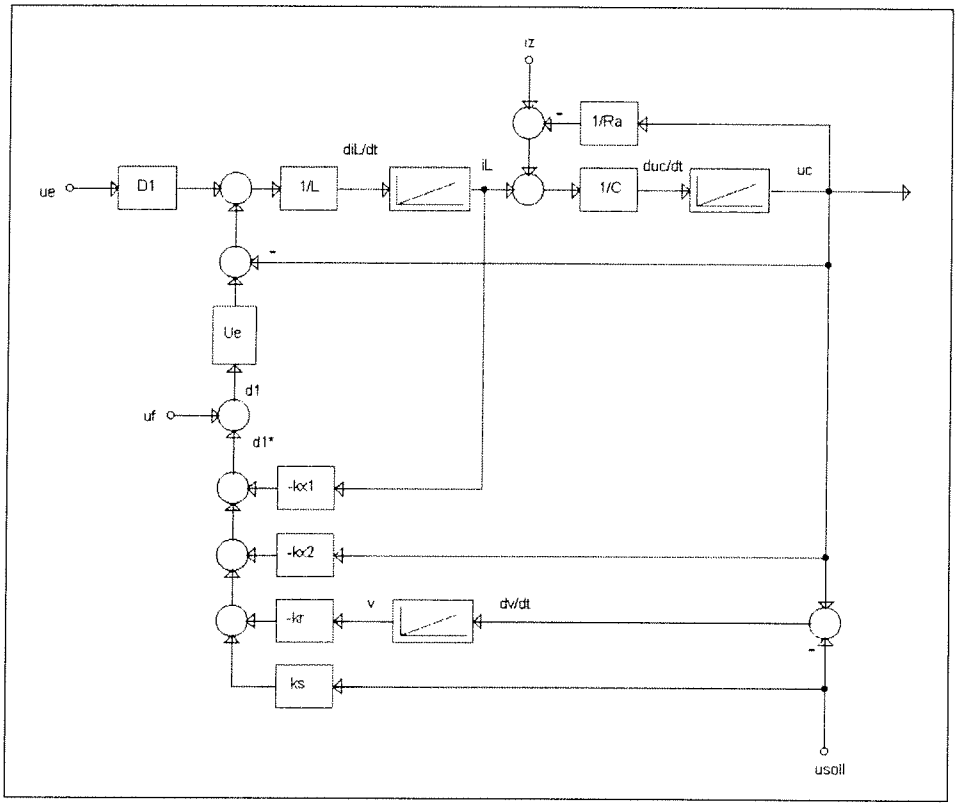


Bild 3

Die Größen $\hat{i}_z$ und $\hat{u}_f$ sind Hilfsgrößen, die zur Berechnung der Ausgangsimpedanz und der offenen Schleife nötig sind.

Nach dem Übergang in den Frequenzbereich erhält man aus der Gleichung

$$\frac{d\hat{x}}{dt} = \underline{A}\,\hat{x} + \underline{b}\,\hat{u}_e + \underline{c}\,\hat{u}_f + \underline{d}\,\hat{i}_z + \underline{e}\,\hat{u}_{soll}$$

alle relevanten Frequenzgänge mit der kompexen Frequenz $p = \sigma + j\omega$:

Störfrequenzgang F_s ($\hat{u}_f = \hat{i}_z = \hat{u}_{soll} = 0$): $\quad \hat{x} = (p\,\underline{E} - \underline{A})^{-1}\,\underline{b}\,\hat{u}_e$

Ausgangsimpedanz ($\hat{u}_e = \hat{u}_f = \hat{u}_{soll} = 0$): $\quad Z_a = \dfrac{(p\,\underline{E} - \underline{A})^{-1}\,\underline{d}}{1 - 1/R_a\,(p\,\underline{E} - \underline{A})^{-1}\,\underline{d}}$

Führungsfrequenzgang F_w ($\hat{u}_f = \hat{i}_z = \hat{u}_e = 0$): $\quad \hat{x} = (p\,\underline{E} - \underline{A})^{-1}\,\underline{b}\,\hat{u}_{soll}$

Offene Schleife F_o ($\hat{u}_e = \hat{i}_z = \hat{u}_{soll} = 0$): $\quad F_o = \dfrac{d_1^*}{d_1} = \dfrac{d_1^*}{\hat{u}_f + d_1^*}$

----> $\quad F_o = \dfrac{-\underline{k}_x\,(p\,\underline{E} - \underline{A})^{-1}\,\underline{c}}{1 - \underline{k}_x\,(p\,\underline{E} - \underline{A})^{-1}\,\underline{c}}$

Berechnung der Frequenzgänge

Mit Mathematica können diese Frequenzgänge berechnet werden:

```
(*  Einheitsmatix e: *)  e = IdentityMatrix[Dimensions[A][[1]]];
ψ = Inverse[p e - A]; Simplify[ψ]; Fs1 = ψ . b; Fs = Simplify[Fs1[[2, 1]]]
Za1 = ψ . d; Za2 = Simplify[Za1[[2, 1]]]; Za = Simplify[ Za2 / (1 - 1 / Ra Za2) ]
Fw1 = ψ . e; Fw = Simplify[Fw1[[2, 1]]]
Fo1 = ψ . c; Fo2 = kx . Fo1; Fo3 = -Fo2[[1]] / (1 - Fo2[[1]]); Fo = Simplify[Fo3][[1]]
```

Störfrequenzgang $Fs = (D1\, p\, Ra) / (L\, p^2\, (1 + C\, p\, Ra) + kr\, Ra\, Ue + C\, kx1\, p^2\, Ra\, Ue + p\, (Ra + kx1\, Ue + kx2\, Ra\, Ue))$

Ausgangsimpedanz $Za = \dfrac{p\, (L\, p + kx1\, Ue)}{p + C\, L\, p^3 + kr\, Ue + kx2\, p\, Ue + C\, kx1\, p^2\, Ue}$

Führungsfrequenzgang $Fw =$
$((kr + ks\, p)\, Ra\, Ue) / (L\, p^2\, (1 + C\, p\, Ra) + kr\, Ra\, Ue + C\, kx1\, p^2\, Ra\, Ue + p\, (Ra + kx1\, Ue + kx2\, Ra\, Ue))$

Offene Schleife $Fo = -\dfrac{((kr + kx2\, p)\, Ra + kx1\, p\, (1 + C\, p\, Ra))\, Ue}{p\, (Ra + L\, p\, (1 + C\, p\, Ra))}$

Formt man den Zähler der Offenen Schleife um, so erhält man

$$Z_{Fo} = 1 + \left(\frac{kx2}{kr} + \frac{kx1}{kr\,Ra}\right) p + \frac{kx1}{kr}\, C p^2$$

mit dem Ausdruck für Z_{Fo} lassen sich die Ausdrücke für die Frequenzgänge vereinfachen:

Störfrequenzgang: $$F_s = \frac{D1\, p}{kr\, Ue} \frac{1}{Z_{Fo}\left(1 + \frac{L}{kx1\, Ue}\, p\right)} \approx \frac{D1\, p}{kr\, Ue} \frac{1}{Z_{Fo}}$$

Ausgangsimpedanz: $$Z_a = \frac{kx1\, p}{kr} \frac{1}{1 + \frac{kx2}{kr}\, p + \frac{kx1}{kr}\, C p^2} \approx \frac{kx1\, p}{kr} \frac{1}{Z_{Fo}}$$

Führungsfrequenzgang: $$F_w = \frac{1 + \frac{ks}{kr}\, p}{Z_{Fo}\left(1 + \frac{L}{kx1\, Ue}\, p\right)} \approx \left(1 + \frac{ks}{kr}\, p\right) \frac{1}{Z_{Fo}}$$

Offene Schleife: $$F_o = -\frac{kr\, Ue}{p} \frac{Z_{Fo}}{1 + \frac{L}{Ra}\, p + L\, C p^2}$$

Mit der Annahme, daß im interessierenden Frequenzbereich $1 >> \frac{L}{kx1\, Ue}\, p$ ist, wird Z_{Fo} zum charakterisierenden Polynom des Systems. Für dieses Polynom sollen nun die Dämpfung D und die Zeitkonstante T festgelegt werden. Dazu dient ein Koeffizientenvergleich mit einem idealen Tiefpass 2. Ordnung

$$N(p) = 1 + 2\, D_z\, T_z\, p + T_z^2\, p^2$$

mit $T_z = L_z C$ folgt

$L_z = \frac{kx1}{kr}$ (L_z ist die mit dem Zustandsregler gebildete Induktivität)

und

$D_z = \frac{1}{2}\frac{kx2+\frac{kx1}{Ra}}{\sqrt{kr\,kx1\,C}}$ (D_z ist die mit dem Zustandsregler erreichte Dämpfung)

Bestimmung der k-Faktoren

Störverhalten:

Um bei 100Hz eine gewisse Dämpfung von A dB zu erhalten, ergibt sich aus dem Störfrequenzgang

$$20 \log \frac{2\pi\, 100\, D1}{kr\, Ue} = -A \qquad \text{---->} \qquad kr = 10^{\frac{A}{20}} \frac{200\pi\, D1}{Ue}$$

Ausgangsimpedanz:

Die Ausgangsimpedanz soll bei der Frequenz fzamax den größten Wert Zamax besitzen.

Mit $L_z = \frac{kx1}{kr} = \frac{Zamax}{\pi\, fzamax}$ -----> $kx1 = kr\,\frac{Zamax}{\pi\, fzamax}$

und aus $D_z = \frac{1}{2}\frac{kx2+\frac{kx1}{Ra}}{\sqrt{kr\,kx1\,C}}$ ----> $kx2 = 2\,D_z\sqrt{kr\,kx1\,C} - \frac{kx1}{Ra}$

Mit den Daten der Strecke lässt sich die Stördämpfung, die Dämpfung des Regelkreises und das Lastsprungverhalten vorgeben:

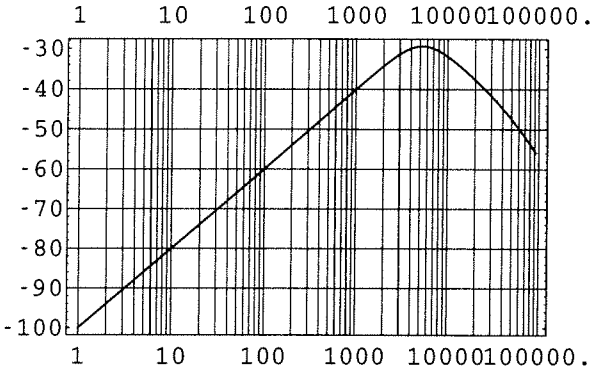
```
(*  Induktivität L  *)                  L0 = 1.3 10^-6;
(*  Lastwiderstand Ra  *)               Ra0 = 0.33;
(*  Taktperiode Ts  *)                  Ts0 = 10 10^-6;
(*  Tastverhältnis D1  *)               D10 = 0.34;
(*  Eingangsspannung Ue *)              Ue0 = 42;
(*  Proportionalfaktor des Sollwertes *)  ks0 = 0;

(*  Gewünschte 100 Hz - Dämpfung [dB]: *)     A0 = 60;
(*  Mit ZuR zu realisierende Dämpfung:  *)    Dz0 = 0.7;
(*  Einschwingzeit für Lastsprünge [s]: *)    Δt0 = 0.1 10^-3;
(*  Max. Über- bzw. Unterschwinger  [V]:*)    Δua0 = -1.5;
(*  Lastwiderstand vor Lastsprung  [Ω  *)     Rav0 = 33.3;
(*  Lastwiderstand nach Lastsprung  [Ω]:*)    Ran0 = 0.33;

(*  Max. Ausgangsimp.[Ω]:*)  Zamax0 = Δua0 Ran0 Rav0 / ((Ran0 - Rav0) Ua0 - Rav0 Δua0);

fzamax = 1 / (2 Δt);  Cz = 1 / (4 π fzamax Zamax);  Lz = Zamax / (π fzamax);

kg = LinearSolve[K, Ko];
```

Mathematica berechnet hieraus den benötigten Ausgangskondensator C_z und die Rückkoppelfaktoren k_x, die für dieses Dynamikverhalten eingestellt werden müssen:

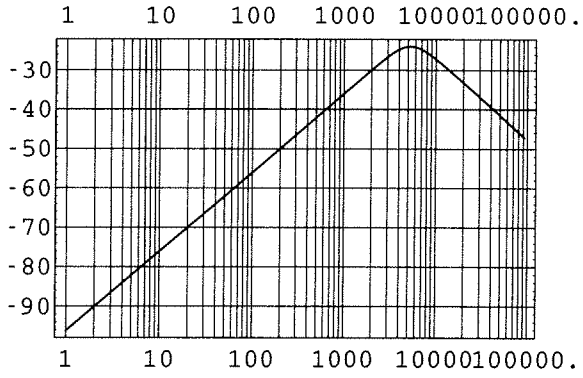
```
fzamax= 5000.;     Cz= 0.000406359;     Lz= 2.49339×10^-6 ;
Zustandsvektor kx1,kx2,kr = {{0.0126824}, {0.188235}, {5086.39}}
Max. Ausgangsimpedanz Zamax= 0.0391661
```

Darstellung der Frequenzgänge

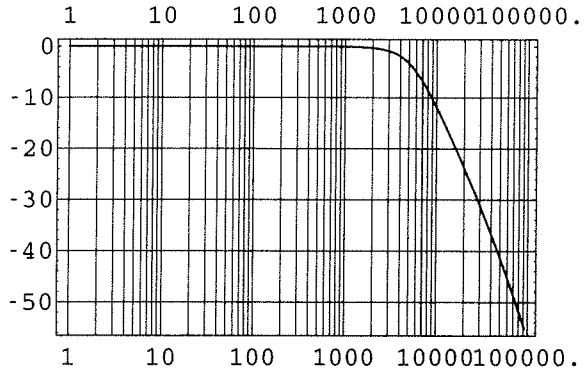
Störfrequenzgang Fs:



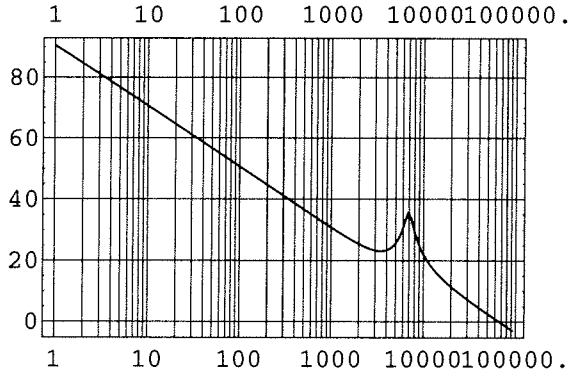
Ausgangsimpedanz Za:

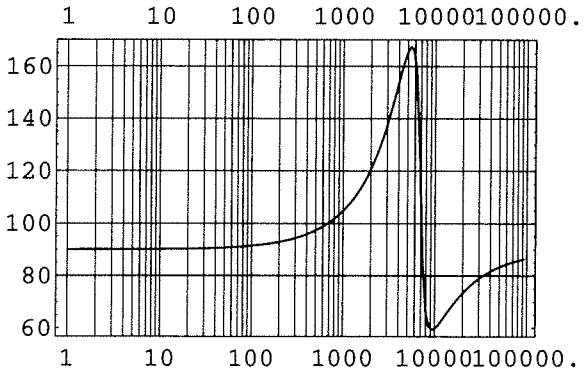


Führungsfrequenzgang Fw:



Offene Schleife Fo:





Gute Dynamikeigenschaften eines Schaltnetzteils lassen sich in der Regel nur erreichen, wenn der Regler **und** die Strecke entsprechend ausgelegt sind. Deshalb sollten diese Dimensionierungen schon vor dem ersten Brettaufbau der Schaltung erfolgt sein. Das ist mit Linearmodellen und leistungsstarken Mathematikprogrammen sehr effizient und durchgängig möglich.

Modellierung und Simulation nachrichtentechnischer Schaltungen und Systeme

Karsten Einwich, Uwe Knöchel, Peter Schwarz
Fraunhofer-Institut für Integrierte Schaltungen, Außenstelle EAS Dresden
Zeunerstr. 38, 01069 Dresden, e-mail: knoechel@eas.iis.fhg.de

1. Einleitung

Der Telekommunikationsmarkt ist in den vergangenen Jahren dynamisch gewachsen. Ein Ende dieser Entwicklung ist derzeit nicht abzusehen. Wichtige Bereiche sind die Mobilkommunikation, kabelgebundene Kommunikation (besonders Hochgeschwindigkeits-Modems) und satellitengebundene Übertragungstechniken für Rundfunk, Fernsehen und Multimedia. Allgemein können für den Entwurf dieser Systeme folgende Aussagen getroffen werden:

- kurze Entwicklungszeiten (Minimierung der Time-to-market)
- Design Re-use
- hoher Integrationsgrad (System on Chip - SoS)
- aufwendige digitale Signalverarbeitung, HW/SW-Systeme
- analoge Schnittstelle, oft hochfrequenter Übertragungskanal

Um die begrenzte Bandbreite des Übertragungskanals bestmöglich auszuschöpfen, werden komplizierte digitale Übertragungsverfahren angewendet, die im Sender und Empfänger einen hohen Aufwand an digitaler Signalverarbeitung erfordern.

Daher hat der Anteil digitaler Schaltungen am Gesamtsystem stark zugenommen. Analoge Baugruppen bilden in der Regel die Schnittstelle zum (oft hochfrequenten) Übertragungskanal. Obwohl analoge Systemkomponenten nur einen geringen Anteil der Komplexität des Systems haben, muß für sie ein hoher Aufwand an Entwicklungszeit in Anspruch genommen werden. Die Entwicklung von analog- und mixed-signal-Schaltungen wird von den CAE-Werkzeugen schlechter unterstützt als der reine Digitalentwurf. Lücken gibt es insbesondere beim Übergang vom System- zum Schaltungsentwurf.

Wenn die zu entwerfenden Systeme nicht zu kompliziert sind, ist der Einsatz weitverbreiteter Mixed-Signal-Simulatoren wie Saber, Eldo, Spectre oder auch PSpice ausreichend. Bei komplizierten und großen Systemen sind solche Simulatoren jedoch oft ungeeignet. Ursachen dafür können in zu langen Simulationszeiten, unzulässiger Schaltungsgröße oder fehlenden Modellen liegen. Für die Simulation derartiger Systeme werden im Beitrag Lösungsansätze vorgestellt, die eine effiziente Gesamtsystemsimulation unterstützen sowie digitale und analoge Komponenten einbeziehen.

2. Anforderungen an den Designflow

Die Forderung nach einer ständigen Verkürzung der Markteinführungszeit und nach auf den einzelnen Kunden zugeschnittenen Systemlösungen führt zu erhöhten Anforderungen an den Entwurfsprozeß. Dies sind u.a. :

- Parallelisierung der Entwurfsschritte
- Flexibilität
- Design Re-use
- Einsatz von IP's
- Designsicherheit

Parallelisierung der Entwurfsschritte

Meistens ist die Entwicklungszeit kritischer als der Entwicklungsaufwand. Der Entwicklungszeitverkürzung durch Effektivitätssteigerung sind allerdings Grenzen gesetzt. Deshalb müssen, um die Zeit bis zur Markteinführung weiter zu verkürzen, Entwurfsschritte parallelisiert werden. Dazu ist es notwendig, daß die Abhängigkeiten der Entwurfsschritte durch eine auf das System zugeschnittene Entwurfsmethodik soweit wie möglich gelockert werden. Dies erfordert neben einer Designmethodik, unterstützt durch angepaßte Tools, auch ein entsprechendes Architekturkonzept. So kann die Verlagerung von Hardware auf Soft- (Firm-) ware eine erhebliche Parallelisierung des Designprozesses ermöglichen, da Soft- und Hardware bei Verwendung entsprechender auf die Zielarchitektur anpaßbarer Codegeneratoren parallel entwickelt werden können. Bild 1 zeigt die Parallelisierung der verschiedenen Entwurfsphasen, wobei der Testprogrammentwurf mit einbezogen ist. Durch „virtuelles Testen" wird versucht auch diese Ent-

wurfsphase soweit wie möglich nach vorn, parallel zu den anderen Phasen zu schieben [EIN98], [EIN99].

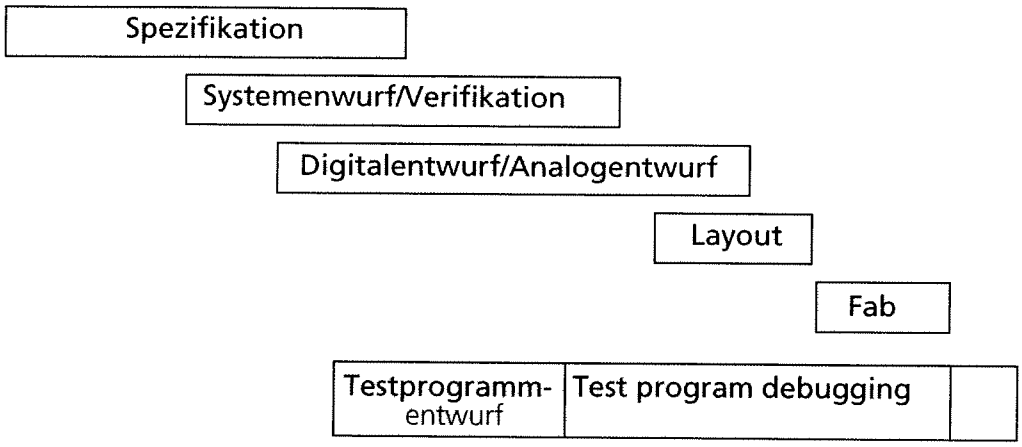


Bild 1:Parallelisierung der Entwurfsphasen

Flexibilität

Die Forderung nach auf den Kunden zugeschnittenen Speziallösungen führt zu erhöhten Anforderungen der Systembeschreibung und der Versionsverwaltung. Die Systembeschreibung muß zum einen so abstrakt wie möglich sein, um schnell Änderungen bzw. Anpassungen auf andere Architekturkonzepte durchführen zu können, und zum anderen einen weitestgehend automatischen Weg bis zum Silizium ermöglichen. Häufig werden verschiedene Versionen eines Systems zumindest teilweise parallel entwickelt bzw. Re-designs erfolgen parallel zu neuen Designs. Dabei ist besonders die Wartung, d.h. die Fehlerbeseitigung problematisch. Erschwerend kommt hinzu, daß die verfügbaren Designtools keine oder jedes eine andere Versionsverwaltung unterstützen.

Design Re-use

Die Wiederverwendung von Daten vorhergehender Designs ist unabdingbar. Dies führt in der Regel zu einer bottom-up/top-down Methodik. Problematisch dabei ist die oben beschriebene Versionsverwaltung. Die zur Zeit noch klassische Re-use Methodologie besteht im kopieren des alten Designstands. Dies führt zu immer unübersichtlicheren Designständen.

Einsatz von IP's

Der ständig steigende Integrationsgrad führt dazu, daß Systemkomponenten, die ursprünglich von verschiedenen Anbietern bereitgestellt wurden, auf einem Chip integriert werden können. Somit müssen unabhängig von einander entworfene Einzelkomponenten in einem Designsystem auf einen IC gebracht werden. Dabei will zum einen der IP-Anbieter sein know how weitestgehend schützen und zum anderen der IP-Nutzer eine möglichst hohe Designsicherheit und Systemverständnis erreichen.

Designsicherheit

Komplexere Systeme sind nicht als Gesamtsystem auf den unteren Abstraktionsebenen (Transistor-/Gatelevel) verifizierbar. Zur Gesamtsystemverifikation muß auf Verhaltens- (System-)ebene verifiziert werden. Die Einzelblöcke werden auf den unteren Ebenen verifiziert. Um eine hohe Designsicherheit zu erhalten muß überlappend hinsichtlich des Abstraktionslevels (repräsentiert das Verhaltensmodell die Realisierung mit ausreichender Genauigkeit oder umgekehrt erreicht die Realisierung die erforderlichen Systemparameter) und hinsichtlich des Ortes (sind die Blockinterfaces richtig implementiert) überprüft werden.

Emulation, HW/SW-COsimulation und Rapid Prototyping spielen bei der Designverifikation eine wachsende Rolle. Im folgenden Abschnitt werden Toolerweiterungen beschrieben, die die Erfüllung der obengenannten Anforderungen unterstützen.

3. Modellierungs- und Simulationstechniken für Mixed-Signal Anwendungen

3.1. Simulatorkopplung

Auf jeder Entwurfsebene oder für jede Entwurfsaufgabe haben die zugehörigen Simulatoren optimale Eigenschaften bezüglich Geschwindigkeit, Modellumfang, Beschreibungsmittel und Genauigkeit. Bei der Simulation eines komplexen Systems kann man durch Simulatorkopplung erreichen, daß für jedes Teilsystem der "ideale" Simulator eingesetzt wird.

Eine Simulatorkopplung muß folgende Aufgaben erfüllen:

- IPC (Inter Proces Communication) z.B. socket oder shared memory
- Synchronisation der Prozesse
- Synchronisation der verschiedenen Simulatorzeitachsen
- Abbildung/Interpretation verschiedener Signaltypen

Neben kommerziell verfügbaren Simulatorkopplungen können angepaßte und damit sehr effizient arbeitende Lösungen über die bei den meisten Simulatoren vorhandenen C- oder Fortranschnittstellen realisiert werden [EIN96]. Implementiert wurde die Simulatorkopplung Verilog-Leapfrog-COSSAP-Saber, die erfolgreich bei der Simulation von Telekommunikationsschaltkreisen (vgl. Abschnitt 4) eingesetzt wurde. Darüberhinaus existieren Kopplungen zwischen dem Inhouse-Simulator KOSIM und kommerziellen Simulatoren.

3.2. Analogeinbindung

Der Nachteil von Simulatorkopplungen, daß zwei Tools (für welche entsprechende Lizenzgebühren zu entrichten sind) für die Gesamtsystemverifikation notwendig sind, kann für eine große Klasse von Aufgaben umgangen werden, indem Gleichungslöser in andere Simulatoren (z.B. in den Systemsimulator COSSAP [SYN-99]) eingebunden werden. Dabei werden verfügbare Differentialgleichungslöser als „Nutzerelemente" in die Systembeschreibung des Systemsimulators einbezogen. [SCH98]

Ein weiterer Vorteil dieser Vorgehensweise ist, daß die Simulatorschrittweite und der verwendete Differentialgleichungslöser blockspezifisch angepaßt werden kann, was zu einer erheblichen Performancesteigerung führt.

Diese Methode hat sich vor allem bei der Gesamtsystemverifikation großer Mixed-Signal-Systeme bewährt. Neben der Einbindung in Systemsimulatoren ist auch eine Einbindung in digitale Simulatoren (VHDL/Verilog) möglich.

3.3. Modellbibliotheken

Spezialsimulatoren für nachrichtentechnische Systeme (z.B. COSSAP [SYN-99], SPW), die auf der obersten Entwurfsebene eingesetzt werden, verfügen über umfangreiche Modellbibliotheken nachrichtentechnischer Baugruppen und Algorithmen. Diese Simulatoren sind jedoch im Designflow vieler Firmen (noch) nicht enthalten, vor allem wenn nur gelegentlich nachrichtentechnische Systeme entworfen werden oder (aus Preisgründen) in kleineren Unternehmen. Stattdessen werden im Entwurfsprozeß oft Schaltungs- oder Logiksimulatoren eingesetzt. Durch Bereitstellung von Modellbibliotheken nachrichtentechnischer Standardbaugruppen können Schaltungs- und Logiksimulatoren so erweitert werden, daß sie zur Gesamtsimulation nachrichtentechnischer Systeme geeignet sind. Dabei kommen Verhaltensmodelle zum Einsatz. Sie ermöglichen gegenüber Transistor- bzw. Gatterschaltungen eine deutliche Simulationsbeschleunigung.

Zum Einsatz in Schaltungssimulatoren sind besonders Modelle geeignet, die in analogen Verhaltensbeschreibungsprachen beschrieben sind. Bislang existieren für die verschiedenen Simulatoren unterschiedliche, jedoch ähnliche Sprachen. Mit VHDL 1076.1 (VHDL-AMS) wurde 1999 eine Verhaltensbeschreibungssprache für Analog- und Mixed-Signal-Anwendung standardisiert. In diesem Jahr soll sie von den ersten Simulatoren unterstützt werden. Damit können simulatorunabhängige Modelle und Bibliotheken entwickelt werden. Im Bereich der Digitalsimulation ist die Beschreibungssprache VHDL bereits seit langem im Einsatz.

Im Rahmen des BMBF-Projekts „Modellierung nachrichtentechnischer Baugruppen und Systeme" wurde im Jahr 1994 eine Bibliothek mit zeitdiskreten und z.T. auch zeitkontinuierlichen Verhaltensmodellen von Standardkomponenten und Verfahren der Übertragungstechnik implementiert. Diese umfaßt u.a. Modelle für:

- Funktionsgeneratoren, Randomgeneratoren
- Modulations- und Kodierverfahren
- Kanäle und Leitungen
- analoge und digitale Filter
- AD- und DA-Wandler, Sample & Hold

Eine weitere Bibliothek enthält mathematische Operationen und Funktionen für Gleitkomma- und eine parametrisierbare Festkomma-Arithmetik. Diese Modelle sind besonders zur Modellierung von Verfahren der digitalen Signalverarbeitung (Filter, adaptive Systeme) geeignet. Die Parameter der Festkomma-Arithmetik (Wortlänge, Präzision, Rundungsverfahren u.a.) gestatten die Berücksichtigung von Arithmetikeffekten der Zielhardware, wie z.B. Rundungsrauschen oder Überlaufschwingen in einem sehr frühen Entwurfsstadium. Die Bibliothek enthält Modelle für:

- Grundoperationen, wie z.B. Addition
- trigonometrische Funktionen
- Exponential- und Logarithmusfunktionen
- Summation, Differenz, Faltung

3.4. HW/SW-Cosimulation

In der Nachrichtentechnik werden häufig Hardware-/Softwaresysteme eingesetzt. Vorteil des Einsatzes von Softwarekomponenten ist die hohe Flexibilität der Systeme. Durch Softwaremodifikation können mit gleicher Hardware verschiedene Produktversionen (z.B. für unterschiedliche nationale Standards) realisiert werden. Um das Zusammenspiel von Hardware und Software in ihrer Systemumgebung zu untersuchen, wird die HW/SW-Cosimulation eingesetzt [KTH-99].

Für den OAK-DSP wurde eine Umgebung zur HW/SW-Cosimulation realisiert. Die bit- und taktzyklusgenaue Kopplung (Eigenentwicklung) des OAK-Debuggers (Instruktion Set Simulator - ISS) mit dem VHDL-Simulator Leapfrog erlaubt den gemeinsamen Test des entwickelten C-Codes mit einem Modell der späteren HW-Umgebung.

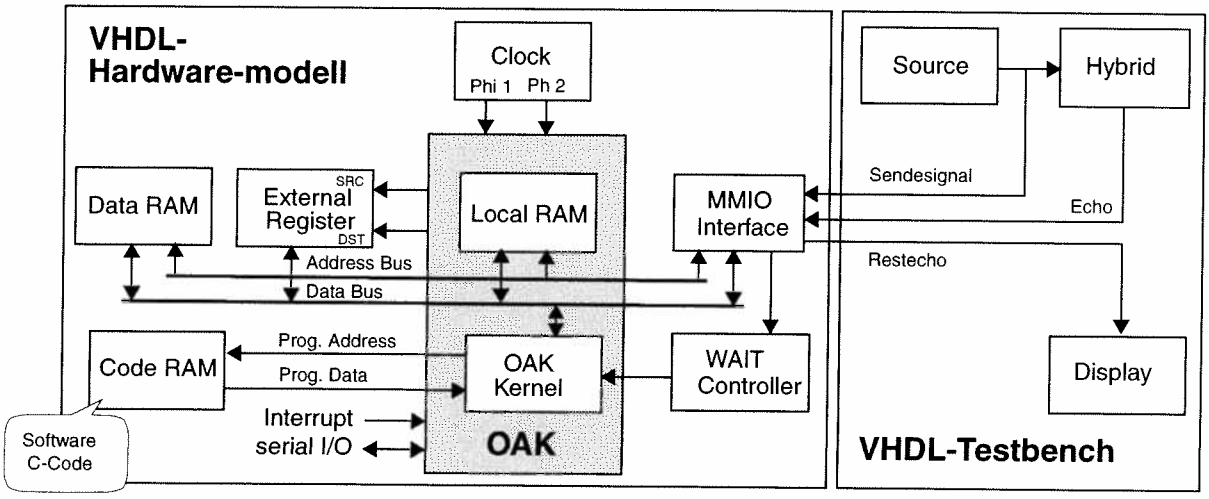


Bild 2: Testbench der HW/SW-Cosimulation

Bild 2 zeigt den Aufbau der Testbench zur Durchführung der HW/SW-Cosimulation für die Applikation Echocanceller. Sie kann in drei Bestandteile gegliedert werden:

- Der OAK-debugger als DSP-Entwicklungsumgebung bildet den Kern der Cosimulation. Er repräsentiert das DSP-Core und führt den DSP-Code aus. Dabei wird ein Debuggen des DSP-Codes ermöglicht.
- Die den DSP umgebende Hardware, im Beispiel externer RAM, Register, Interfaces und Taktgenerator werden in VHDL beschrieben. Der DSP-Code ist im Modell des Code-RAMs abgelegt. Diese Hardwareumgebung wird mit einem VHDL-Simulator ausgeführt und bildet in Verbindung mit dem OAK-debugger das HW/SW-System.
- Zum Test des HW/SW-Systems wird eine Testbench verwendet, die Eingangssignale bereitstellt und eine Visualisierung der Ergebnisse gestattet. Diese ist ebenfalls in VHDL realisiert. Um die Systemumgebung effizient beschreiben zu können, wurde eine Bibliothek mit Verhaltensmodellen (beha-

vioral VHDL) entwickelt. Sie enthält Modelle von Standardbaugruppen der Nachrichtentechnik, wie z.B. Signalgeneratoren, Filter, Kodierer und Modulatoren. Auf diese Weise kann eine komplexe Systemumgebung beschrieben werden.

Hauptvorteil der HW/SW-Cosimulation ist die Darstellung des Timingverhaltens von HW- und SW-Komponenten. Dafür wird eine komfortable Entwicklungsumgebung bereitgestellt, die ein Debuggen der Software (im Oak-Debugger) und der Hardware (im VHDL-Debugger) ermöglicht. Das Inhouse-Tool ADDA gestattet die Visualisierung analoger und digitaler Waveforms sowie Postprocessing.

4. Anwendungsbeispiel

Ein Teil der oben beschriebenen Toolerweiterungen wird beim Entwurf sogenannter SLICOFI-Systeme (Subscriber Line Interface Codec Filter) [TIE94] für öffentliche und private Vermittlungsstellen eingesetzt. Diese Systeme bestehen aus 3 bis 36 Schaltkreisen abhängig von den implementierten Features bzw. der Kanalanzahl (2 bis16). So besteht z.B. das Zweikanalsystem aus zwei Hochvoltschaltkreisen und einem CMOS Mixed-Signal Schaltkreis. Der Hochvoltschaltkreis [ZOJ95] arbeitet als Leitungstreiber (bis zu 150V). Der Mixed-Signal Schaltkreis realisiert u.a. D/A, A/D-Wandlung, Leitungsanpassung, Spannungscharakteristik, Signaling-Funktionen (z.B. Klingeln, Gebührenimpuls) und zahlreiche Test- und Meßfunktionen. Er enthält neben analogen und digitalen Filtern 2 DSP's + 1 DSP-IP-Core, 1 µ-Controller und drei verschiedene digitale Interfaces. Er besteht aus ca. 800.000 Transistoren.

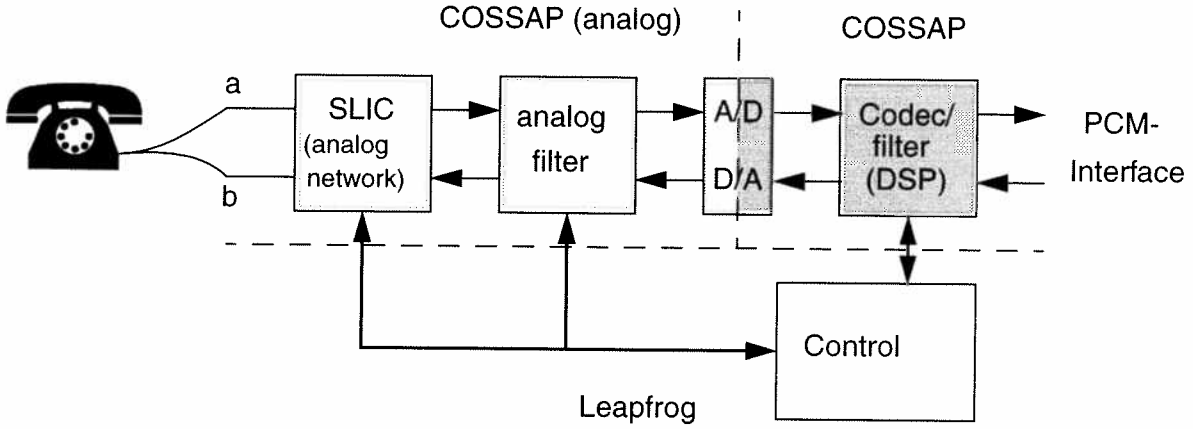


Bild 3:Blockschaltbild für einen Kanal

Bild 3 zeigt das Blockschaltbild für einen Kanal, wobei die Aufteilung auf verschiedene Simulationsalgorithmen gekennzeichnet ist.

Die analogen Komponenten werden mit einem Analog- bzw. Mixed-Signal-Simulator verifiziert, wobei die Funktionalität vom Systementwurf vorgegeben wird. Soweit wie notwendig müssen aus dem Entwurf resultierende Einschränkungen oder nicht ideales Verhalten in die Systemmodelle eingebracht werden. Für die Gesamtsystemverifikation werden die analogen Blöcke in COSSAP mittels der Einbindung eines analogen Gleichungslösers modelliert.

DSP-Algorithmen lassen sich sehr gut in COSSAP beschreiben. Aus den Filterplänen kann mittels Codegenerierung ein Assemblerprogramm für ein spezielles DSP-Core generiert werden. Die Controlalgorithmen lassen sich nur sehr schlecht durch Blockschaltbilder in COSSAP beschreiben. Deshalb werden die Controlalgorithmen im VHDL-Simulator Leapfrog (Cadence) entworfen. Aus dieser Beschreibungsform läßt sich ebenfalls automatisch Assemblercode für den speziellen µ-Controller generieren. Damit ist der Algorithmenentwurf weitestgehend vom Digitalentwurf entkoppelt, womit das DSP und µ-Controllerdesign parallel zum Algorithmendesign erfolgen kann.

Eine Gesamtsystemsimulation in einem Mixed-Signalsimulator ist derzeit für ein System dieser Größenordnung aus Rechenzeitgründen nicht möglich. So werden auf Transistorlevel nur Einzel- oder einige wenige verschaltete Einzelblöcke verifiziert. Die Rechenzeit liegt trotzdem im Bereich von Stunden je µs simulierter Zeit. Die digitalen Komponenten können auf Register-Transfer-Level effektiv zusammen simuliert werden. Die Rechenzeit liegt im Stundenbereich je ms, wird auf Gatterebene simuliert ist sie ca. 10 mal so hoch. Um das Gesamtsystem verifizieren zu können sind jedoch simulierte Zeiten im Sekundenbereich notwendig. D.h. selbst die schnellen RTL-Beschreibungen sind um Größenordnungen zu langsam für die Systemverifikation. Mit dem oben beschriebenen Systemmodell werden Rechenzei-

ten von 30s/ms erreicht, wobei die Genauigkeit durch Modellabgleich (u.a. Bittruesimulationen) sehr hoch ist. Damit sind auch komplizierte Abläufe und Einschwingvorgänge simulierbar.

5. Zusammenfassung und Ausblick

Im Beitrag wurden verschiedene Simulationstechniken vorgestellt, die insbesondere beim Entwurf nachrichtentechnischer Systeme eingesetzt werden. Die Simulation soll dabei einen Beitrag leisten, die Entwicklungszeiten von Schaltkreisen und Systemen zu verkürzen. Die Simulationsverfahren können je nach Art des zu untersuchenden Systems sehr unterschiedlich sein.

Bei der Simulation von sehr komplexen Systemen, wie im Beispiel dargestellt, haben sich besonders Simulatorkopplungen bewährt. Die verschiedenen Baugruppen des Gesamtsystems werden dabei mit dem jeweils geeigneten Simulator untersucht. Die Gesamtsystemsimulation erfolgt dann durch Kopplung der verschiedenen Simulatoren. Dadurch kann der Modellierungsaufwand vermindert werden (oft sind weiterverwendbare Modelle verfügbar). Sind nur sehr wenig analoge Baugruppen zu berücksichtigen und sind diese nicht zu kompliziert, können anstatt der Simulatorkopplung analoge Modelle direkt in einen Systemsimulator eingebunden werden. Auf diese Weise kann bei vertretbarer Genauigkeit der Rechenzeitaufwand gesenkt werden, oftmals teure zusätzliche Simulatorlizenzen werden eingespart.

Sind die Systeme weniger komplex, ist der Einsatz eines Systemsimulators (z.B. COSSAP) oft zu kostenintensiv. Im Beitrag wurde gezeigt, wie durch zusätzliche Modellbibliotheken Analog- oder VHDL-Simulationen für diesen Einsatzfall erweitert werden können. Weiterhin wurde ein Verfahren zur Hardware-Software-Cosimulation vorgestellt.

In weiterführenden Arbeiten wird die Gesamtsystemsimulation und das Systemmodell derart erweitert, das ein virtueller Test von Schaltkreisen möglich wird. Das ermöglicht ein Debuggen der Testsoftware noch bevor der Schaltkreis in Silizium verfügbar ist. Auf diese Weise können die Entwurfszeiten noch zusätzlich verkürzt werden.

6. Literatur

[EIN95] Einwich, K.; Haase, J.; Prescher, R.; Schwarz, P.: Makromodellierung für Mixed-Signal-Schaltungen. mikro-elektronik + mikrosystemtechnik, Heft 4/1995

[EIN96] Einwich, K.; Schwarz, P.; Trappe, P.; Zojer, H.: Simulatorkopplung für den Entwurf komplexer Schaltkreise der Nachrichtentechnik. 7. ITG-Fachtagung "Mikroelektronik für die Informationstechnik", Chemnitz, 18./19. März 1996, 139-144

[SCH98] Schwarz, P.; Clauß, C.; Einwich, K.; Knöchel, U.; Matz, K.: Hybride Simulation nachrichtentechnischer Systeme. 12. ASIM-Workshop "Systemsimulation", Zuerich, 15.-18.9.1998

[EIN98] Einwich, K.; Schwarz, P.; Trappe, P.; Chambers, T.; Krampl, G.; Zojer, H.; Sattler, S.: Virtual Test of Complex Mixed-Signal Telecommunication Circuits reusing System-Level Models. 4th IEEE International Mixed-Signal Testing Workshop, pp.237-242, The Hague, 1998

[EIN99] Einwich, K.; Altmann, S.; Leitner, T.; Krampl, G.; Hoppenstock, R., Sattler, S.: Applying High-Level Virtual Test to a Complex Mixed-Signal Telecommunication Circuit. 5th IEEE International Mixed-Signal Testing Workshop, une 15th - 18th, 1999, Delta Whistler Resort, Vancouver(Whistler) British Columbia, Canada, pp. 91-95

[KRA99] Einwich, K.; Krampl, G.; Hoppenstock, R.; Koutsandreas, P.; Sattler, S.: A multi-level modeling approach rendering virtual test engineering (VTE) economically viable for highly complex telecom circuits. Design, Automation and Test in Europe Conference DATE'99, Munich, Germany, 9 - 12 March 1999, Proceedings User's Forum, 227-231.

[KTH-99] Knöchel, U.; Tannert, U.; Haufe, J.; Schwarz. P.: Verifikation nachrichtentechnischer Systeme mit Systemsimulation und HW/SW-Cosimulation. GI/ITG/GME-Workshop, Paderborn 9. - 11.3.98, HNI-Verlagsschriftenreihe, Bd. 36, S. 175 -184

[SCH95] Schwarz, P.; Einwich, K.; Haase, J.; Prescher, R.: Mixed-Mode Design: Experiences with Multi-Level Macromodeling. Proc. Workshop "Advances in Analog Circuit Design", Villach, Austria, 26.-28. April 1995, 10/1-10/22

[SYN-99] COSSAP Produktinformation: http://www.synopsys.com/products/dsp/dsp.html

[TIE94] Tiefenbacher, M; Caldera, P; Dielacher, F; et.al.: A four channel CMOS codec filter circuit. SICOFI-4. IEEE J. Solid-State-Circuits SC-29 (1994) H.8

[ZOJ95] Zojer, B.; Koban, R.; Petschacher, R.; Sereinig, W.: Integration of a Subscriber Line Interface Circuit (SLIC) in a new 170V Smart Power Technology. Proc. ISDSP'95

Modellierung eines optischen Freiraum-Nachrichtenübertragungssystems

E. Kube, G.Hansel, Teleconnect GmbH Dresden
J.Haase, J.Becker, FhG IIS/EAS Dresden

Die optische Datenübertragung über Freiraumstrecken gestattet die schnelle Realisierung von Verbindungen mit hohen Bitraten im Entfernungsbereich bis etwa 5 km. Eine Modellierung dieser Übertragungssysteme erfordert die Einbeziehung elektronischer, optoelektronischer und optischer Komponenten. Es wird gezeigt, wie mit einem Netzwerkanalysator in Verbindung mit Verhaltensmodellen für die optoelektronischen und optischen Komponenten eine Systemoptimierung bezüglich der Minimierung der Bitfehlerrate durchgeführt werden kann.

1 Einleitung

Wie in der klassischen Hochfrequenztechnik gibt es auch im optischen Wellenlängenbereich zwei Möglichkeiten zur Wahl des Übertragungsmediums: die leitungsgebundene Übertragung und die Übertragung über elektromagnetische Wellen im freien Raum. Im optischen Bereich ist die leitungsgebundene Übertragungstechnik mit Lichtwellenleitern (LWL) breit eingeführt, und es können alle gegenwärtig denkbaren Anforderungen mit Bandbreiten bis in den Terabit-Bereich erfüllt werden. Die optische Freiraumübertragung, obwohl bereits in den 60er Jahren zwischen der Erfindung des Lasers und der Herstellung der ersten brauchbaren LWL aktuell, findet wieder zunehmende Beachtung und ermöglicht Übertragungsbandbreiten bis in den Gbit/s-Bereich auf Streckenlängen bis etwa 5 km [1],[2]. Die Übertragungssicherheit ist zwar durch die Eigenschaften der Atmosphäre im optischen Bereich (Nebel, Schnee, Luftturbulenzen) eingeschränkt, doch Vorteile wie schnelle Installationsmöglichkeit, große Übertragungsbandbreite und keine Probleme mit der Zuteilung von Frequenzbändern lassen sie gegenüber der Richtfunktechnik besonders in privaten Netzen immer mehr an Boden gewinnen.

Durch die Einbeziehung optoelektronischer Signalwandlungen und durch das veränderliche Verhalten der Atmosphäre als optisches Übertragungsmedium ist die Simulation eines kompletten Übertragungssystems relativ aufwendig. Sie ist dennoch sinnvoll, da immer weniger Zeit für eine Systementwicklung zur Verfügung steht und experimentelle Arbeiten nach der trial-and-error-Methodik minimiert werden müssen.

2 Systemsimulation

2.1 Simulationsaufgabe

Der prinzipielle Aufbau einer optischen Freiraum-Nachrichtenübertragungsstrecke ist in Bild 1 dargestellt. Das in elektrischer Form vorliegende digitale Quellensignal wird im optischen Sender in das optische Signal gewandelt, über die Atmosphäre als Übertragungsstrecke weitergeleitet und im optischen Empfänger wieder in ein elektrisches Signal transformiert. Das Ausgangssignal nach dem Entscheider soll möglichst fehlerfrei dem Eingangssignal entsprechen. Auf dem Übertragungsweg wird das Digitalsignal durch die Sende- und Empfangsschaltungen, die elektrooptischen Wandlungen und durch die Ausbreitung durch die Atmosphäre mehr oder minder deformiert und von additiven und multiplikativen Rauschprozessen beeinflußt. Ziel der Simulation muß es sein, diese Beeinflussungen darzustellen und durch Parametervariation die Bitfehlerrate des Gesamtsystems zu minimieren.

Bei der Analyse von optischen Übertragungssystemen werden vielfach die einzelnen Blöcke durch ihr Systemverhalten charakterisiert. Mit den Systemmodellen für die Blöcke wird das Verhalten des Gesamtsystems numerisch ausgewertet. Dazu werden die Amplitude des Empfangssignals am Entscheidungspunkt und das akkumulierte Rauschen an dieser Stelle getrennt berechnet. Aus den Signal- und Rauschamplituden am Entscheidungspunkt wird dann die Bitfehlerrate analytisch ermittelt [3],[4].

Bei dem im folgenden vorgeschlagenen Weg zur Analyse eines optischen Freiraum-Übertragungssystems wird auf diesem Vorgehen aufgebaut. Es werden jedoch folgende Änderungen vorgenommen:

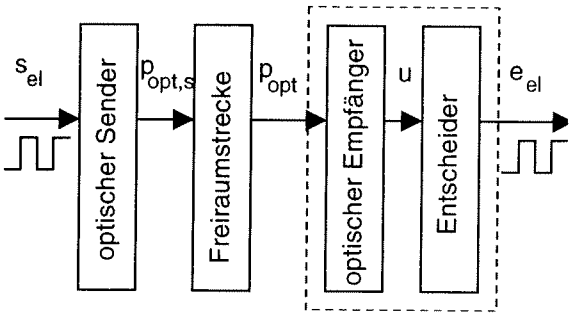


Bild 1: Optische Freiraum-Übertragungsstrecke

- Die elektrischen Sende- und Empfangsschaltungen werden nicht durch Blockmodelle sondern durch Schaltungsmodelle auf Netzwerkebene beschrieben und simuliert. Auf diese Weise kann der Einfluß von Schaltungsänderungen auf die Systemeigenschaften unmittelbar ermittelt werden.
- Für die optoelektronischen Bauelemente werden Verhaltensmodelle verwendet, die in die kompletten Sende- und Empfangsschaltungen einbezogen werden.
- Die Beeinflussung des Signals durch das Übertragungsmedium Atmosphäre wird durch ein Blockmodell beschrieben.
- Die Berechnung der Bitfehlerrate nach dem Entscheider erfolgt mit den Ergebnissen der Transientensimulation und der Rauschanalyse mit einem besonderen Auswerteprogramm.

Für die Simulation des Gesamtsystems kann ein Netzwerksimulator, der den Einsatz einer Verhaltensbeschreibungssprache erlaubt (wie z.B. Saber oder ELDO) oder mit Einschränkungen auch eine SPICE-Variante mit ABM-Option eingesetzt werden.

2.2 Bitfehlerratenberechnung

Zur Bitfehlerratenberechnung wird eine semianalytische Methode nach dem folgenden Schema durchgeführt:
1. Schritt: Über eine Simulation im Zeitbereich werden die Werte m_0 und m_1 des Signals u am Entscheider bestimmt, wenn die logischen Signale 0 bzw. 1 gesendet worden sind.

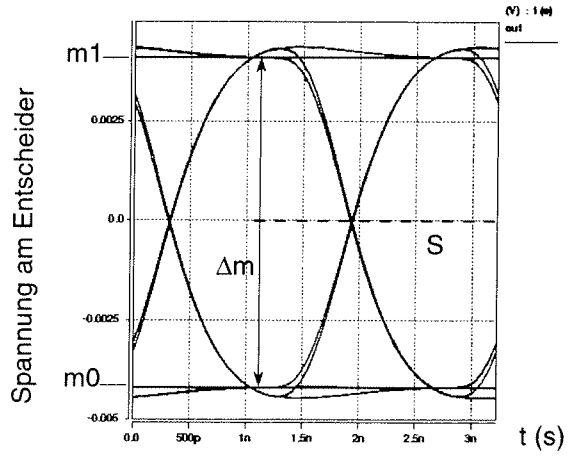


Bild 2: Werte von m_0 und m_1 im Augendiagramm

Dazu wird das Übertragungssystem mit einer Eingangsbitfolge simuliert, in der sich Null- und Eins-Signale entsprechend einer pseudostatistischen Zufallsfolge abwechseln, und die Werte von m_0 und m_1 können aus dem Augendiagramm des Signals am Eingang des Entscheiders entnommen werden (Bild 2). Dabei werden Rauscheinflüsse nicht berücksichtigt.

2. Schritt: Das Leistungsspektrum des Rauschens wird an den durch m_0 und m_1 gekennzeichneten Arbeitspunkten mit dem Rauschanalyseprogramm des verwendeten Netzwerksimulators berechnet. Dabei wird vorausgesetzt, daß die Rauschquellen durch ihre Leistungsspektren vollständig beschrieben sind, d.h. ihre Amplituden sind normalverteilt. Außerdem wird davon ausgegangen, daß die einzelnen Rauschquellen im System unkorreliert sind.

Die Rauschanalyse liefert so bei logischem 0-Signal am Entscheidungspunkt das von der Frequenz abhängige Leistungsspektrum $S_{U0}(f)$ und bei logischem 1-Signal $S_{U0}(f)$. Aus diesen beiden Verläufen ergeben sich die zugehörigen Streuungen σ_0 bzw. σ_1 der Amplitude des Signals am Entscheider zu

$$\sigma_0^2 = \int_0^{fg} \left(\sqrt{S_{U0}(f)}\right)^2 df \qquad (1)$$

$$\sigma_1^2 = \int_0^{fg} \left(\sqrt{S_{U1}(f)}\right)^2 df \qquad (2)$$

f_g ist die Grenzfrequenz, bis zu der das Rauschen berücksichtigt wird.

Signale u am Entscheider, deren Wert zum Abtastzeitpunkt größer als die vorgegebene Schwelle S ist, werden als logische 1-Signale erkannt. Signale, die kleiner als diese Schwelle sind, werden als logische 0-Signale registriert. Die Entscheiderschwelle liegt zwischen m_0 und m_1 (siehe Bild 2). Es wird angestrebt, die Entscheiderschwelle so zu legen, daß die Bitfehlerrate möglichst gering ist. Die Bitfehlerrate ergibt sich aus der Summe der Wahrscheinlichkeiten dafür, daß bei gesendetem logischem 0-Signal das Signal u am Entscheider einen Wert größer als die Entscheiderschwelle S und bei gesendetem logischem 1-Signal einen Wert kleiner S hat. Es kann davon ausgegangen werden, daß 0- und 1-Signale gleich häufig vorkommen. Damit ergibt sich für die Bitfehlerrate

$$BER = \frac{1}{2} \cdot \left(\int_S^{\infty} \frac{e^{-\frac{1}{2}\cdot\left(\frac{u-m_0}{\sigma_0}\right)^2}}{\sqrt{2\pi}\sigma_0} du + \int_{-\infty}^{S} \frac{e^{-\frac{1}{2}\cdot\left(\frac{u-m_1}{\sigma_1}\right)^2}}{\sqrt{2\pi}\sigma_1} du \right)$$

Näherungsweise liefert die Auswertung der Integrale ($m_0 < S < m_1$)

$$BER \approx \frac{1}{2\sqrt{2\pi}} \cdot \left(\frac{e^{-\frac{1}{2}\cdot\left(\frac{S-m_0}{\sigma_0}\right)^2}}{\left(\frac{S-m_0}{\sigma_0}\right)} + \frac{e^{-\frac{1}{2}\cdot\left(\frac{m_1-S}{\sigma_1}\right)^2}}{\left(\frac{m_1-S}{\sigma_1}\right)} \right) \qquad (3)$$

Es ist zu erwarten, daß sich die beiden Streuungen σ_0 und σ_1 wegen der Unterschiede im Rauschen der APD bei 0- und 1-Signal erheblich voneinander unterscheiden. Die optimale Entscheiderschwelle S liegt in diesem Fall nicht mehr in der Mitte zwischen den beiden Signalwerten m_0 und m_1.

3.Schritt: Obwohl die Dämpfung der Übertragungsstrecke starken zeitlichen Schwankungen z.B. durch Luftturbulenzen unterworfen ist, kann jedoch für den Entscheidungsprozeß mit einer konstanten Dämpfung gerechnet werden. Die Änderungsgeschwindigkeit der Dämpfung ist um Größenordnungen kleiner als die Änderungsgeschwindigkeit des zu übertragenden Signals. Die Rechnungen der ersten beiden Schritte werden für unterschiedliche Dämpfungen des Übertragungsmediums durchgeführt. Anschließend kann unter Berücksichtigung der Verteilungsfunktion der Dämpfung eine mittlere Fehlerrate berechnet werden.

3 Modellierung der Schaltungen

Die Modellierung der Sende- und Empfangsschaltungen erfordert Modelle für die einzelnen elektronischen und optoelektronischen Bauelemente. Während für die passiven und aktiven elektronischen Bauelemente in der Regel Modellbibliotheken von den Herstellern bereitgestellt werden, sind für kompliziertere optoelektronische Wandler wie Laserdioden und Avalanche-Photodioden keine entsprechenden Modelle für die Schaltungssimulation verfügbar.

Eine Modellierung auf Device-Niveau erfordert Angaben über physikalische und geometrische Parameter des optoelektronischen Bauelements, an deren Offenlegung die Hersteller kaum interessiert sind. Für die Applikation wird lediglich ein Datenblatt mit den wichtigsten Parametern und Abhängigkeiten herausgegeben. Gegebenenfalls muß der Anwender diese Daten durch Rückfragen beim Hersteller bzw. durch eigene Messungen ergänzen. Aufgrund dieser Angaben ist es nun möglich, Verhaltensmodelle zu generieren, die auf diesen Daten und einigen grundlegenden physikalischen Abhängigkeiten basieren.

Am Beispiel eines Empfängers mit Avalanche-Photodiode soll diese Vorgehensweise näher erläutert werden. Danach wird noch kurz auf die Modellierung des Lasers und der Übertragungsstrecke eingegangen.

3.1 Empfangsanordnung

Die wesentlichen Bestandteile des Empfängers sind die APD und ein rauscharmer Transimpedanzverstärker. Nachfolgende Schaltungsteile wie z.B. Regel- und Begrenzerverstärker haben auf das Signal/Rausch-Verhältnis eine untergeordnete Bedeutung und werden in die Simulation zur Optimierung der Bitfehlerrate nicht einbezogen.

Das Blockschaltbild mit den Bezeichnungen der Signale nach Bild 2 zeigt Bild 3.

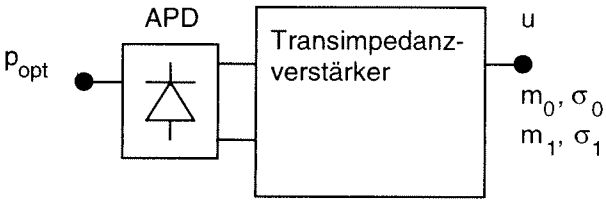


Bild 3: Optischer Empfänger

Als optisches Eingangssignal dient ein 622 Mbit/s-Impulsfolge, die mit ihren Pegeln in Bild 4 dargestellt ist.

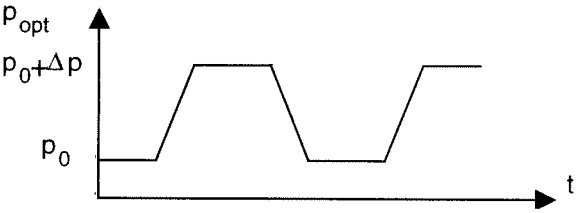


Bild 4: Optisches Eingangssignal

p_0 entspricht dem empfangenen optischen Eingangssignal, wenn ein logisches 0-Signal übertragen wird. Es ist im wesentlichen gleich dem Hintergrundlicht. $p_0+\Delta p$ ist das optische Eingangssignal, wenn ein 1-Signal übertragen wird. Das Signal u am Ausgang des Transimpedanzverstärkers ist das Eingangssignal des Entscheiders.

3.2 APD-Modell

3.2.1 Grundlegende Beziehungen

Ziel der APD-Modellierung ist es, die grundlegenden Zusammenhänge, die die Funktion einer Avalanche-Photodiode beschreiben, zu erfassen und ein Modell zu erstellen, das in einer Netzwerksimulation verwendet werden kann. Die wesentlichen Beziehungen, die dabei berücksichtigt wurden, sind im folgenden zusammengestellt. Mit der einfallenden optischen Leistung P_{opt} ist der primäre Fotostrom

$$I_p = R \cdot P_{opt} \qquad (4)$$

verbunden. Dabei ist R die spektrale Empfindlichkeit (responsivity). Für Licht einer vorgegebenen Wellenlänge ist dieser Wert konstant und kann dem Datenblatt entnommen werden. Diese Wandlung entspricht dem von PIN-Photodioden bekannten verhalten. Bei APD's wird dieser

primäre Photostrom durch Lawinenvervielfachung verstärkt. Im stationären Zustand gilt für den durch die einfallende optische Leistung generierten Strom

$$I = M \cdot I_p \tag{5}$$

In dieser Beziehung ist M ein Multplikationsfaktor, für den theoretisch

$$M = \frac{1}{1-\left(\frac{vd}{V_{BR}}\right)^m} \tag{6}$$

gilt. Es ist vd die Spannung über dem pn-Übergang in Sperrichtung, V_{BR} die Durchbruchspannung der Diode und m eine empirische Konstante, die durch Approximation aus Datenblattangaben gewonnen wird. Das dynamische Verhalten des generierten Stromes i wird durch die näherungsweise geltende Differentialgleichung

$$\tau_L \cdot \frac{di}{dt} + i = M \cdot i_p \tag{7}$$

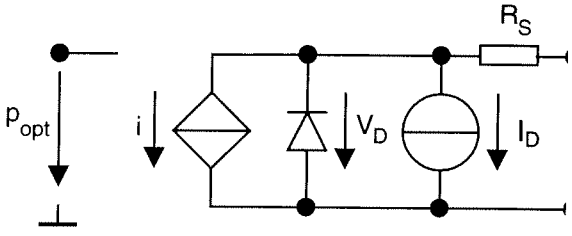
beschrieben. Dabei ist τ_L die Lawinenansprechzeit, die sich aus dem Verstärkungs-Bandbreite-Produkt f_T des Vervielfachungsvorgangs über die Beziehung

$$\tau_L = \frac{M}{2\pi \cdot f_T}$$

ergibt [5].

3.2.2 Klemmenverhalten

Im Modell der APD werden die Beziehungen (5) bis (7) durch eine gesteuerte Quelle, die den Strom i liefert, berücksichtigt. Dabei wird als Strom-Spannungs-Beziehung für die gesteuerte Stromquelle die in Bild 5 angegebene Gleichung verwendet. Zusätzlich werden im APD-Modell (Bild 5) der Zuleitungswiderstand R_S, die Diodenkapazität und der Dunkelstrom I_D (als Konstantstromquelle) berücksichtigt. Die Modellparameter werden aus Datenblattangaben unter Verwendung typischer Materialdaten ermittelt.



$$0 = \tau_L \cdot \frac{di}{dt} + i - M \cdot R \cdot p_{opt} \qquad \text{(M gemäß (6))}$$

Bild 5: APD-Modell

Für die Diode wird das im Netzwerksimulator verfügbare Modell [6] verwendet. V_{BR} bestimmt die Durchbruchspannung der Diode. Null-Sperrschichtkapazität, Diffusionsspannung und Gradationsexponent der Diode werden entsprechend der Kapazitätskennlinie der zu modellierenden APD gewählt.

Auf der Grundlage der in Bild 5 angegebenen Struktur wurden für die Simulatoren Saber und PSpice Verhaltensmodelle für eine Anzahl von handelsüblichen APD erstellt. Durch Einbau von analytische Beziehungen kann die Spannungsabhängigkeit von M und das dynamische Verhalten von i berücksichtigt werden.

3.2.3 Modellierung des Rauschens

Das Rauschen wird im APD-Modell durch eine zusätzliche Rauschstromquelle parallel zum generierten Photostrom berücksichtigt. Der Rauschstrom wird im wesentlichen durch das vom vervielfachten Photostrom I_P verursachte Schrotrauschen bestimmt. Der durch den Dunkelstrom hervorgerufene Rauschanteil wird vernachlässigt. Für das Leistungsspektrum S_{IR} des Rauschstromes I_R innerhalb der Bandbreite Δf wird die Beziehung

$$\frac{I_R}{\sqrt{\Delta f}} = \sqrt{S_{IR}} = \sqrt{2q \cdot M^2 \cdot f(M) \cdot I_P} \tag{8}$$

verwendet. Dabei sind q die Elementarladung und M der Multiplikationsfaktor. Der Zusatzrauschfaktor des Vervielfachungsprozesses f(M) wird durch die Beziehung

$$f(M) = k \cdot M + (1-k) \cdot \left(2 - \frac{1}{M}\right) \tag{9}$$

beschrieben. Dabei bezeichnet k das materialabhängige Ionisierungsverhältnis von Elektronen und Löchern. Der Zusatzrauschfaktor wird vielfach durch

$$f(M) = M^x \tag{10}$$

angenähert [3]. Für handelsübliche APD's sind die Werte für x in den Datenblättern angegeben.

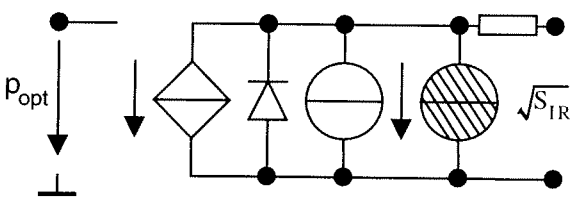


Bild 6: APD-Modell mit Rauschquelle

In Bild 6 ist ein APD-Modell mit Rauschquelle angegeben, wie es für die Kleinsignal-

Rauschanalyse im Schaltungssimulator verwendet werden kann.

3.3 Laserdiode

Das Laserdiodenmodell wird analog zur APD als Verhaltensmodell definiert. In einer ersten Stufe wird der Wandlungsprozeß elektrischer Strom - optische Leistung durch eine stückweise lineare Kennlinie dargestellt. Kohärenz- und Rauscheigenschaften sowie Modulations- und Einschwingverhalten werden zunächst nicht in das Modell einbezogen, da bei der optischen Freiraum-Übertragungstechnik die dadurch verursachten Störungen in der Regel gegenüber den Störungen durch die Atmosphäre zu vernachlässigen sind.

3.4 Übertragungsstrecke

Die optische Übertragungsstrecke hat als Eingangs- und Ausgangsschnittstellen die Strahlungsapertur der Laserdiode und die Empfangsapertur der Photodiode. Dazwischen liegen Lichtwellenleiter und Linsen zur Bündelung der zum Empfänger gerichteten Strahlung mit wenigen mrad Divergenz und zur Fokussierung der Empfangsstrahlung auf die Photodiode. Zwischen Sende- und Empfangsoptik befindet sich dann das eigentliche Übertragungsmedium, die Strekke durch die Atmosphäre mit einer Länge bis zu einigen km. Je nach dem Kohärenzgrad der Strahlungsquelle kann der Strahlengang mit Methoden der geometrischen Optik oder der Wellenoptik (Gaußstrahlen) berechnet werden. Auf diese Weise erhält man die bei bestimmter Streckenlänge konstante optische Grunddämpfung des Systems. Von wesentlicher Bedeutung für die Übertragungssicherheit ist vor allen die veränderliche Dämpfung durch die Atmosphäre, sei es durch Streuung an Aerosolen (Nebel, Regen, Schnee) oder durch schnelle statistische Schwankungen infolge von Luftturbulenzen. Weiterhin muß die Hintergrundstörstrahlung durch Sonnenlicht beachtet werden.

Für die Modellierung muß ein sinnvoller Kompromiß zwischen Aufwand und Nutzen gesucht werden. Zunächst werden in ein Verhaltensmodell für die Übertragungsstrecke einbezogen:

- Strahldivergenz des Senders
- Empfangsfläche der Empfangsoptik
- langsame Dämpfungsveränderungen durch verschiedene Wetterlagen mit Zeikonstanten von sec.
- schnelle Dämpfungsveränderungen durch Turbulenzen mit lognormaler Verteilung und Charakterisierung durch Mittelwert und Streuung
- Veränderliche Hintergrundstrahlung

Die Parameter der Atmosphäre werden aus den Ergebnissen langjährige Ausbreitungsmessungen und Angaben von meteorologischen Stationen für bestimmte Klimazonen ermittelt.

4 Ergebnisse der Simulation

Der Simulation lag ein optisches Freiraum-Übertragungssystem mit der Bitrate 622 Mbit/s zugrunde. Um die Ergebnisse einer breiteren Anwendung zugänglich zu machen, wurde die Simulation mit dem weit verbreiteten Simulationstool PSpice durchgeführt.

Die Sende- und Empfangsschaltung einschließlich der optoelektronischen Wandlerbauelemente und die Übertragungsstrecke werden jeweils als subcircuit-Files oder über Schematics eingegeben. Als Signalquelle wird eine pseudostatistische Impulsfolge definiert. Für die Gesamtschaltung wird dann eine Transientensimulation durchgeführt und so die worst-case-Augenöffnung der Impulsfolge Δm vor dem Entscheider bestimmt. Mit einer Rauschanalyse werden die Rauschspannungen σ_0 und σ_1 bei den Signalpegeln m_0 und $k \cdot m_1$ berechnet, wobei der Faktor k von der Impulsform abhängt und bei NRZ-Signalen näherungsweise 0,5 beträgt.

Nach Gleichung (3) wird dann mit einem vom Simulator unabhängigen Rechenprogramm die Bitfehlerrate BER errechnet. Durch Parametervariation kann so die Abhängigkeit der BER von Schaltungs-, Signal- und Störparametern errechnet werden. Als Beispiel zeigt Bild 9 die BER des 622 Mbit/s-Übertragungssystems in Abhängigkeit von der optischen Empfangsleistung Δp mit der Hintergrundlichtleistung p_0 als Parameter.

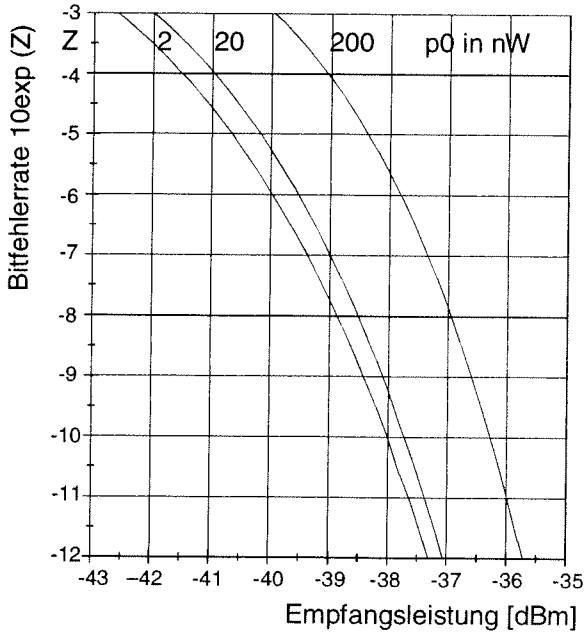


Bild 7: BER eines 622 Mbit/s-Systems

Werden Dämpfungsschwankungen durch Turbulenzen in der Atmosphäre in die Rechnung einbezogen, so schwankt im Bereich der Grenzempfindlichkeit die Bitfehlerrate in Abhängigkeit von der Zeit. Die Folge sind Burst-Störungen, die jedoch in gewissen Grenzen durch protokollgesteuerte Wiederholungs- und Fehlerkorrekturverfahren minimiert werden können. Bild 10 zeigt die Schwankungen der BER im ms-Bereich, wie sie unter Einbeziehung des Verhaltensmodells der Übertragungsstrecke berechnet wurden.

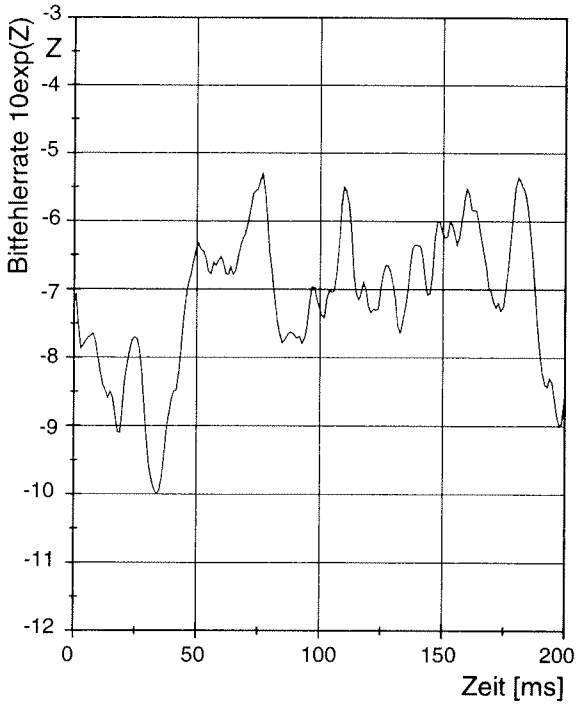


Bild 8: BER bei turbulenzgestörter Übertragung

5 Bewertung des Verfahrens

Der vorgeschlagene Weg erlaubt die direkte Einbeziehung der Sende- und Empfangsschaltungen sowie der Übertragungsstrecke in die Simulation einer optischen Freiraum-Nachrichtenverbindung zur Bewertung der Übertragungsqualität. Die Übereinstimmung der Simulationsergebnisse mit den Meßdaten an realen Systemen liegt im Bereich der Toleranzen der Komponenten und ist somit für die Konzeptphase einer Systementwicklung ausreichend.

Das diesem Beitrag zugrunde liegende Vorhaben CADWOK „CAD-Werkzeuge und Bibliotheken für Wandler mit optischen Komponenten“ wird mit Mitteln des Bundesministeriums für Bildung, Wissenschaft, Forschung und Technologie (BMBF) unter dem Kennzeichen 16 SV 609/8 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt allein bei den Autoren.

Schrifttum

[1] Kube, E.: Nachrichtenübertragung mit Lichtstrahlen in der Atmosphäre. Nachrichtentechnik 19 (1969), H.6, S.201-207.

[2] Szajowski,P.F. u.a.: 2,4 km Free-Space Optical Communication 1550 nm Transmission Link Operating at 2,5 Gb/s Proceedings of SPIE, Vol. 3532, 1998.

[3] Geckeler,S.: Lichtwellenleiter für die optische Nachrichtenübertragung. Berlin-Heidelberg: Springer-Verlag 1986.

[4] Morikuno,J.J., Kang,S.M.: Computer Aided Design of Optoelectronic Integrated Circuits und Systems. Upper Saddle River: Prentice Hall PTR, 1997

[5] Unger,H.G.: Optische Nachrichtentechnik. Teil II: Komponenten, Systeme, Meßtechnik. Heidelberg: Hüthig Buch Verlag, 1992.

[6] Hoefer,E.E.E., Nierlinger,H.: SPICE-Analyseprogramm für elektronische Schaltungen. Berlin: Springer-Verlag, 1985.

Praktische Anwendung symbolischer Analysewerkzeuge in der Elektroniksimulation

Dr.-Ing. Ralf Sommer
Institut für Techno- und Wirtschaftsmathematik e.V.
Erwin-Schrödinger-Str., D-67663 Kaiserslautern
E-mail: sommer@itwm.uni-kl.de

Motivation

Grundlage jeder Schaltungsentwicklung ist die Analyse und Simulation des Schaltungsverhaltens. Was ein Simulator aber nicht ersetzen kann und was sich gleichzeitig als einer der wichtigsten Gesichtspunkte beim Analogschaltungsentwurf herausgestellt hat, ist ein *Schaltungsverständnis* mit einer ungefähren Vorstellung der Funktionsweise einzelner Schaltungsteile. Eine solche Einsicht in die Funktionsweise kann beispielsweise eine qualitative Schaltungserklärung liefern, wie sie bei vielen Schaltungsbeschreibungen anzutreffen ist: "Wenn die Eingangsspannung steigt, schaltet Transistor 1 durch, dadurch reduziert sich die Spannung über R2...". Eine derartige Schaltungserklärung gibt zwar die Funktionsweise einer Schaltung wieder, aber sie erlaubt keine quantitativen Aussagen. Dem gegenüber kann eine *symbolische Formel* auch für eine Schaltungsauslegung genaue Zusammenhänge vermitteln.

Formeln, die das Schaltungsverhalten charakterisieren, konnten bisher nur durch mühselige Handrechnungen abgeleitet werden. Um derartige Handrechnungen durchführen zu können, müssen starke Vereinfachungen in Form von Faustformeln angewendet werden. Der Fehler, der durch die Einführung solcher Faustformeln verursacht wird, ist dabei kaum abzuschätzen. Dazu kommt, daß Handrechnungen sehr fehleranfällig sind und sehr schnell sehr komplex werden können, wodurch die Berechnung alternativer Lösungen sehr stark eingeschränkt wird. Den Schaltungsentwickler bei genau dieser schwierigen Handarbeit in sehr flexibler Weise zu unterstützen und ihm auch die Möglichkeit zu geben, komplexere Alternativen durchzurechnen, bildete die Grundlage für die Entwicklung von Analog Insydes [1]. Analog Insydes, das für "**IN**telligent **SY**mbolic **DE**sign System" steht und als Add-on zu dem Computeralgebrasystem *Mathematica* [2] entwickelt worden ist, ist eine vielseitige Toolbox zur symbolischen Berechnung analoger Schaltungen, die nachfolgend vorgestellt werden soll.

Die Toolbox Analog Insydes

Analog Insydes, das seit kurzem in einer kommerziellen Version zur Verfügung steht, bietet ein weites Spektrum von Funktionen sowohl für die formelmäßige Berechnung linearer Schaltungen industrieller Größenordnung als auch im Bereich der nichtlinearen Modellierung. Die Programmfunktionalität wird durch ein leistungsfähiges hierarchisches Netzlistenformat, das auch die Einbindung beliebiger ABM-Modelle (Analog Behavioral Models) erlaubt, sowie Schnittstellen zu numerischen Simulatoren (z.B. PSpice) abgerundet. Eine Zusammenstellung der Leistungsmerkmale von Analog Insydes findet sich in Abbildung 1 und einen Eindruck der graphischen Oberfläche vermittelt Abbildung 9.

Symbolische Analyse am Beispiel eines Hochfrequenzverstärkers

Bevor auf eine allgemeine Vorgehensweise zur Anwendung symbolischer Schaltungsanalyse eingegangen wird, soll zunächst ein kleines Beispiel vorgestellt werden. Dabei geht es um die Analyse eines Hochfrequenzverstärkers, der einer industriellen Schaltung entstammt. Die Aufgabe soll sein, die für den Frequenzgang der Schaltung (Abbildung 2a) verantwortlichen Parameter zu extrahieren, d.h. die Schaltung zu charakterisieren und gleichzeitig eine Schaltungserklärung mit Hilfe der abgeleiteten Formeln zu finden. Ausgangspunkt der Analyse ist die PSpice Simulation in Abbildung 2b.
Im ersten Schritt werden von Analog Insydes die Netzliste, die Arbeitspunkte, Kleinsignalparameter, Modellparameter sowie die Simulationsdaten der Frequenzgangkurve als Referenz aus PSpice eingelesen. Stehen alle Daten in Analog Insydes zur Verfügung, so ist der nächste Schritt die Auswahl geeigneter Transistormodelle. Im Gegensatz zur numerischen Analyse ist es bei der symbolischen Analyse nicht sinnvoll, mit den komplexesten

Netzlisten-basierte Beschreibung linearer elektronischer Schaltungen und regelungstechnischer Blockschaltbilder

Hierarchische Teilschaltungseingabe und Bauteilmodellierung mittels Subcircuit-Definitionen und Verhaltensbeschreibungen

Netzlistenkonverter zum Einlesen von SPICE2G6- oder PSpice™-Schaltungsbeschreibungen

Lineare Netzwerkelemente: Widerstand, komplexe Impedanz, Leitwert, komplexe Admittanz, Kapazität, Induktivität, unabhängige Strom- und Spannungsquelle, alle Arten von gesteuerten Quellen (VCVS, CCVS, VCCS, CCCS, auch frequenzabhängige Transimmittanzen sind realisierbar), idealer Operationsverstärker (OP), idealer Transkonduktanzverstärker (OTA), Nullator, Norator, Fixator, Kurzschluß- und Leerlaufzweige.

Regelkreiselemente: Proportionalglied, Differenzierglied, Integrator, ideales Laufzeitglied, allgemeine Übertragungsfunktion

Nichtlineare Elemente: Verhaltensbeschreibung von beliebigen nichtlinearen, mehrdimensionalen, dynamischen Strom-Spannungsrelationen

Benutzer-erweiterbare Modellbibliothek, einschließlich PSpice™-Bipolar- und MOSFET-Modellen

Automatische Aufstellung von symbolischen oder gemischt symbolisch/numerischen Netzwerkgleichungssystemen in Formulierung als Modifizierte Knotenanalyse (MNA), Sparse-Tableau-Analyse (STA) oder erweiterte Sparse-Tableau-Analyse (ESTA)

Automatische Gleichungsaufstellung für nichtlineare dynamische Schaltungen

Symbolische Berechnung von Übertragungsfunktionen, Eingangs- und Ausgangsimpedanzen, n-Tor-Parametern, etc.

Berechnung von Dimensionierungsgleichungen und Faustformeln durch Approximation symbolischer Schaltungsgleichungen und Übertragungsfunktionen

Symbolische Analyse im Zeitbereich durch (inverse) Laplacetransformation

Numerische Transientanalyse nichtlinearer differential-algebraischer Gleichungssysteme (DAE)

Numerische parametrische DC Analyse für nichtlineare differential-algebraische Gleichungssysteme

Lösung von Schaltungsgleichungen nicht nur nach Strömen und Spannungen sondern auch nach Bauteilparametern

Graphikfunktionen zur Darstellung von Bodediagrammen, Ortskurven, Pol-/Nullstellen-Verteilungen, Transient-, DC- und Übertragungsverhalten

Importfilter zum Einlesen von Simulationsdaten von PSpice™- und HSPICE™-CSDF-Ausgabedateien sowie Schaltbildern (Schematics) über DXF-Graphikimport.

Abbildung 1: Zusammenstellung der Leistungsmerkmale von Analog Insydes

a) b)

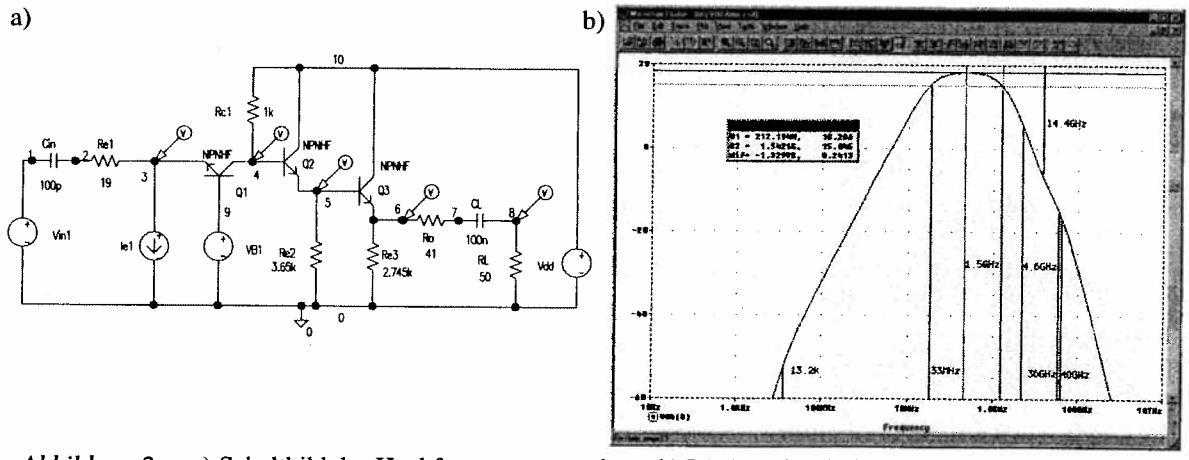
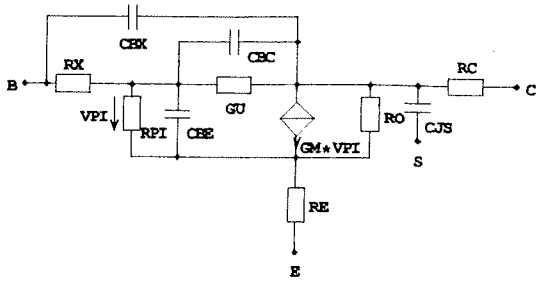


Abbildung 2: a) Schaltbild des Hochfrequenzverstärkers, b) PSpice Simulation des Frequenzgangs der Schaltung mit eingezeichneten Polen und Nullstellen der Übertragungsfunktion

Modellen zu rechnen. Im Gegenteil, angepaßte Modelle, die auch die Variablen enthalten, in denen der Schaltungsentwickler denkt (z.B. Stromverstärkung β statt Transferleitwert g_m), führen hinterher zu interpretierbaren Ergebnissen und reduzieren dabei die Komplexität der Aufgabe in bezug auf die symbolische Verarbeitung deutlich. Bei dem hier betrachteten Beispiel des Hochfrequenzverstärkers zeigt sich allerdings, daß es zur Erfassung der gesamten Schaltungscharakteristik notwendig ist, mit dem kompletten PSpice Kleinsignalmodell (Abbildung 3) zu rechnen, da der genaue Verlauf des Frequenzgangs von den parasitären Elementen im Transistor mitbestimmt wird. Eine semi-symbolische Analyse, d.h. die Aufstellung der symbolischen Schaltungs-



NAME	Q_Q2	Q_Q1	Q_Q3
MODEL	NPNHF	NPNHF	NPNHF
IB	7.37E-06	7.01E-06	7.19E-06
IC	9.00E-04	8.93E-04	8.93E-04
VBE	8.15E-01	8.14E-01	8.14E-01
VBC	-9.00E-01	-3.10E+00	-1.72E+00
VCE	1.72E+00	3.91E+00	2.53E+00
BETADC	1.22E+02	1.27E+02	1.24E+02
GM	3.48E-02	3.45E-02	3.45E-02
RPI	3.51E+03	3.69E+03	3.60E+03
RX	0.00E+00	0.00E+00	0.00E+00
RO	5.66E+04	5.95E+04	5.79E+04
CBE	3.85E-13	3.83E-13	3.83E-13
CBC	3.80E-14	2.87E-14	3.33E-14
CJS	5.00E-14	5.00E-14	5.00E-14
BETAAC	1.22E+02	1.27E+02	1.24E+02
CBX	0.00E+00	0.00E+00	0.00E+00
FT	1.31E+10	1.33E+10	1.32E+10

Abbildung 3: Verwendetes Kleinsignalmodell für die Bipolartransistoren und Arbeitspunktwerte

gleichungen und ihre Lösung mit eingesetzten numerischen Parameterwerten, zeigt eine vollständige Übereinstimmung mit der PSpice Simulation, während die Verwendung eines vereinfachten Kleinsignalmodells ohne Bahnwiderstände eine abweichende Kurve (gestrichelt) ergibt. Die Verläufe sind in einem gemeinsamen Bodediagramm in Abbildung 4 zusammengestellt. Nachdem die Gültigkeit des gewählten Modells sichergestellt ist, wird die semi-symbolische Analyse nun genutzt, um alle Pole und Nullstellen der Schaltung zu bestimmen (Abbildung 5a zeigt die semi-symbolische Übertragungsfunktion).

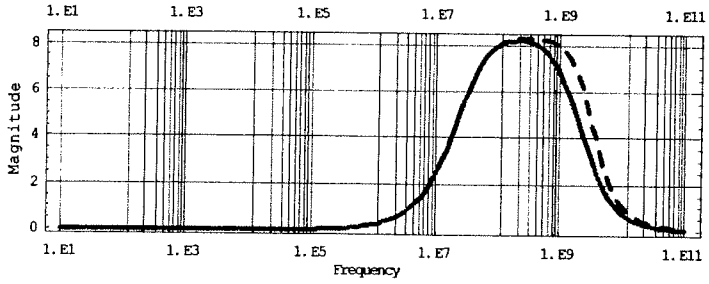


Abbildung 4: Vergleich PSpice Simulationsergebnis (grau), semi-symbolische Analyse mit vollem PSpice-AC Modell (schwarz), semi-symbolische Analyse mit vereinfachtem AC Modell ohne Bahnwiderstände (gestrichelt)

(a)
$$\frac{1.34708\times10^{44}\,s^2 + 2.96172\times10^{33}\,s^3 + 1.62792\times10^{22}\,s^4}{2.7751\times10^{56} + 3.32387\times10^{51}\,s + 1.6538\times10^{43}\,s^2 + 2.37724\times10^{33}\,s^3 + 7.61637\times10^{22}\,s^4 + 5.17678\times10^{11}\,s^5 + s^6}$$

(b)

Nullstellen	Pole
-9.11296×10^{10}	-2.50018×10^{11}
-9.08036×10^{10}	-2.28463×10^{11}
0.	-2.94478×10^{10}
0.	-9.54123×10^{9}
0.	-2.07023×10^{8}
0.	-83524.7

(c)

Nullstellen (Hz)	Pole (Hz)
-1.45037×10^{10}	-3.97916×10^{10}
-1.44518×10^{10}	-3.63611×10^{10}
0.	-4.68676×10^{9}
0.	-1.51853×10^{9}
0.	-3.29487×10^{7}
0.	-13293.4

Abbildung 5: a) Semi-symbolische Übertragungsfunktion, b) Pole und Nullstellen, c) umgerechnet in Frequenzen

Die Pole und Nullstellen (Abbildung 5b) –umgerechnet in Grenzfrequenzen (Abbildung 5c)– sind bereits in die Simulatorausgabe in Abbildung 2b eingezeichnet worden. Sie charakterisieren das Verhalten der Schaltung, so daß nun symbolische Formeln für sie abgeleitet werden sollen. Ohne den Einsatz symbolischer Näherungsverfahren wäre diese Aufgabe übrigens mathematisch unlösbar [3,4]. Es müßten dazu nämlich die Nullstellen des Nennerpolynoms symbolisch berechnet werden, was für Polynome 6. Ordnung nicht mehr möglich ist. Darüber hinaus, so ergibt eine Abschätzung über die Anzahl symbolischer Terme, würde allein das Nennerpolynom ca. 53 Millionen symbolische Terme umfassen, eng bedruckt wäre das eine Sammlung von 250 Büchern, jedes mit ca. 1000 Seiten.

In mehreren symbolischen Approximationsläufen werden Schritt für Schritt die einzelnen dominanten Pole und Nullstellen extrahiert, wobei Abbildung 8 einen Auszug aus der Berechnung der unteren Grenzfrequenzen zeigt. Die Ergebnisse sind in Tabelle 1 zusammengestellt.

Frequenz (exakt)	Frequenz (Näherung)	Formel
13,3kHz	13.3kHz	$\frac{gm_{Q3}}{C_L(1+gm_{Q3}(R_L+R_O))}$
32,95MHz	33,17MHz	$\frac{gm_{Q1}}{C_{in}(1+gm_{Q1}R_{e1})}$
1,52GHz	1,36GHz	$\frac{1}{(CBC_{Q1}+CBC_{Q2}+CJS_{Q1})R_{C1}}$
36,36GHz	36,21GHz	$\frac{(1+gm_{Q1}R_{e1})}{CBE_{Q1}R_{e1}}$

Tabelle 1a: Lage und genäherte Formeln der Pole

Frequenz (exakt)	Frequenz (Näherung)	Formel
14,5GHz	14,5GHz	$\frac{1-gm_{Q1}R\pi_{Q2}}{CBE_{Q2}R\pi_{Q2}}$

Tabelle 1b: Lage und genäherte Formel der Nullstelle

Ein Blick in die Tabellen läßt erkennen, daß für die meisten Grenzfrequenzen kompakte und gut interpretierbare Formeln berechnet worden sind. Betrachtet man die Formeln genauer, so ist zu erkennen, daß die Grenzfrequenzen lokal, d.h. von bestimmten zusammengehörenden Schaltungsteilen, bestimmt werden, beispielsweise die unterste Grenzfrequenz bei 13kHz vom Ausgangskreis mit R_L, C_L, R_O und dem Transferleitwert gm_{Q3} des Transistors Q3. Das ist auch ein Indiz dafür, daß die Schaltung gut entworfen ist, und der Schaltungsentwickler für eine gewisse Entkopplung der Schaltungsfunktionalität gesorgt hat, so daß er die speziellen Eigenschaften einfacher einstellen und anpassen kann.

Es sind aber nicht immer alle Schaltungseigenschaften einfach zu durchschauen: Für den obersten Pol bei 40GHz ergibt sich die etwas umfangreichere Formel aus Abbildung 6, wobei allerdings anzumerken bleibt, daß bei dieser Frequenz der Arbeitsbereich der Schaltung überschritten und die Gültigkeit der verwendeten Kleinsignalmodelle durchaus in Frage zu stellen ist.

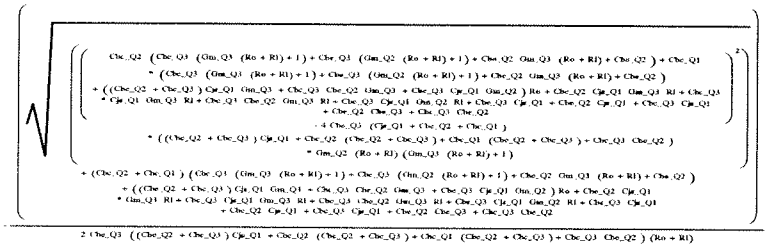


Abbildung 6: Symbolischer Näherungsausdruck für den Pol bei 40GHz

Praktisches Arbeiten mit der symbolischen Analyse

Wie aus dem kleinen Beispiel klar geworden sein dürfte, soll symbolische Analyse keineswegs die numerische Simulation ersetzen – im Gegenteil: Sie hilft die Fragen, die sich aus einer numerischen Simulation ergeben, zu beantworten, das heißt z.B. bestimmte in der Simulation beobachtete Effekte formelmäßig zu extrahieren und zu verstehen.

Dennoch ist die Arbeit mit einem symbolischen Analysewerkzeug gewöhnungsbedürftig, weil der Benutzer weitaus mehr Verständnis über mathematische und modellierungsbedingte Zusammenhänge haben muß, als dies bei der reinen Schaltungssimulation –nach dem Motto Verändern und Beobachten– notwendig ist. Um definierte und anwendbare symbolische Analyseergebnisse abzuleiten, wird eine Strategie vorgeschlagen (Abbildung 7), die in einem engen Abgleich mit numerischer Simulation und in einer systematischen Vorgehensweise zur Extraktion formelmäßiger Beschreibungen bestimmter Fragestellungen führt [5]. Die Fragestellungen können sich z.B. auf Frequenzcharakteristiken, Eingangs-, Ausgangs- bzw. Transferwiderstände oder andere Schaltungseigenschaften (auch nichtlineare) beziehen.

Begonnen wird im Anschluß an die numerische Simulation mit dem Einlesen der Netzliste, Modellkarten, Kleinsignalparameter, aber auch der zuvor berechneten numerischen Simulationsergebnisse, die die Referenz für alle späteren Berechnungen bilden. Über eine semi-symbolische Analyse der Schaltung, d.h. eine Analyse mit symbolischen Methoden, bei der alle Parameter durch konkrete Zahlenwerte ersetzt werden, kann die Gültigkeit der für die symbolische Analyse ausgewählten Modelle im Sinne der Beobachtbarkeit und Erfassung des zu un-

tersuchenden Effekts überprüft werden. Bei Übereinstimmung mit der numerischen Simulation erfolgt dann ein schrittweises Vereinfachen der ausgewählten Modelle. Durch Anwendung symbolischer Approximationstechniken wird der interessierende Effekt Schritt für Schritt formelmäßig extrahiert. Um einen definierten Fehler der abgeleiteten Ergebnisse nicht zu überschreiten, werden durch Vergleiche mit numerischen Simulationen alle symbolischen Gleichungen und Formeln verifiziert, und es wird sichergestellt, daß die symbolischen Analyseergebnisse im relevanten Bereich ihre Gültigkeit behalten.

Am Ende steht eine symbolische Beschreibung, die in vielfacher Weise Anwendung finden kann. Eine Möglichkeit ist, die abgeleitete Formel direkt zur Interpretation des Schaltungsverhaltens zu benutzen und die Abhängigkeiten qualitativ herauszuinterpretieren. Eine andere, zunehmend Bedeutung erlangende Anwendung, ist der Einsatz der abgeleiteten symbolischen Beziehungen direkt zur Verhaltensmodellierung oder zur Nachschaltung weiterer numerischer Verfahren, beispielsweise zur gezielten Schaltungsoptimierung.

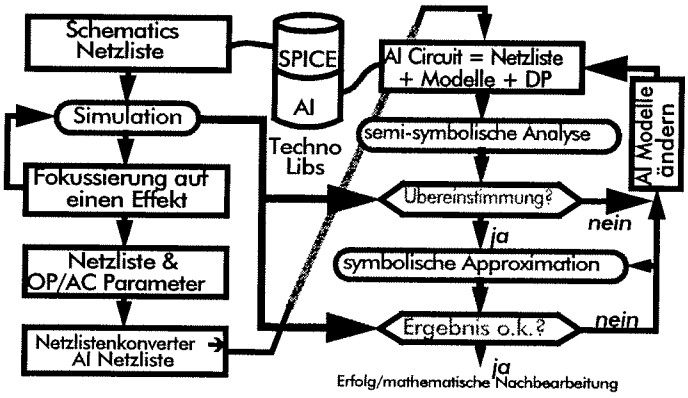


Abbildung 7: Ablaufschema der symbolischen Analyse mit Analog Insydes

Die symbolische Approximation erlaubt selbstverständlich die Analyse größerer Schaltungen als hier vorgestellt. So läßt sich z.B. ein kompletter Operationsverstärker µA741 im Minutenbereich analysieren. Voraussetzung ist allerdings, daß der interessierende Effekt eine gewisse Dominanz aufweist, denn die Berechnung umfangreicher Formeln braucht natürlich Zeit. Hier ist aber auch der Benutzer gefragt, denn Aufgabe der symbolischen Analyse ist es im wesentlichen, Einsichten zu vermitteln und da sollte dann schon einmal bei der Approximation ein größerer Fehler zugunsten eines interpretierbaren Ergebnisses toleriert werden.

Zusammenfassung und Ausblick

Die Kombination symbolischer und numerischer Techniken kann in vielen Fällen wertvolle generelle Einsichten in Schaltungsverhalten vermitteln, die alleine durch numerische Simulation nicht zu gewinnen sind. So geben die im gezeigten Beispiel berechneten Formeln weitaus mehr Aufschluß über das Schaltungsverhalten und die Wirkung der beteiligten Elemente als z.B. eine Empfindlichkeitsanalyse eines Simulators. Der in den Formeln abgebildete funktionale Zusammenhang der Bauelemente läßt Alternativen für die Veränderung der entsprechenden Schaltungseigenschaft deutlich werden, zeigt Abhängig- und Unabhängigkeiten und enthält quantitative Informationen, wie eine Veränderung vorgenommen werden kann. Neben einer gezielten Unterstützung des Schaltungsentwicklers bei Analyse, Modellierung und Dimensionierung ist bei der Anwendung symbolischer Schaltungsanalyse meist besonders interessant, die berechneten Formeln mit dem eigenen Schaltungswissen "abzugleichen".

Literatur

[1] E. Hennig, T. Halfmann, *Analog Insydes Tutorial.* ITWM, Kaiserslautern, Germany, 1998

[2] S. Wolfram, The Mathematica Book, 3rd ed., Wolfram Media/Cambridge University Press, 1996

[3] E. Hennig, R. Sommer, "Approximate Symbolic Pole/Zero Extraction by Equation-Based Simplification Using Eigenvalue Shift Prediction", in *Proc. IEEE International Symposium on Circuits and Systems 1998 (ISCAS98)* , Monterey (USA), Jun. 1998, vol. VI, pp. 25–28

[4] R. Sommer, E. Hennig, "Aspects of Symbolic Approximation and Nonlinear Circuit Analysis", in *Proc. 4th International Workshop on Symbolic Methods and Applications in Circuit Design 1996 (SMACD'96)*, Leuven (Belgium), Oct. 1996

[5] R. Sommer, M. Thole, E. Hennig, "A Generic Circuit Modeling Strategy Combining Symbolic and Numeric Analysis", *in Proc. 5th International Workshop on Symbolic Methods and Applications to Circuit Design (SMACD'98)*, Kaiserslautern, Oct. 1998

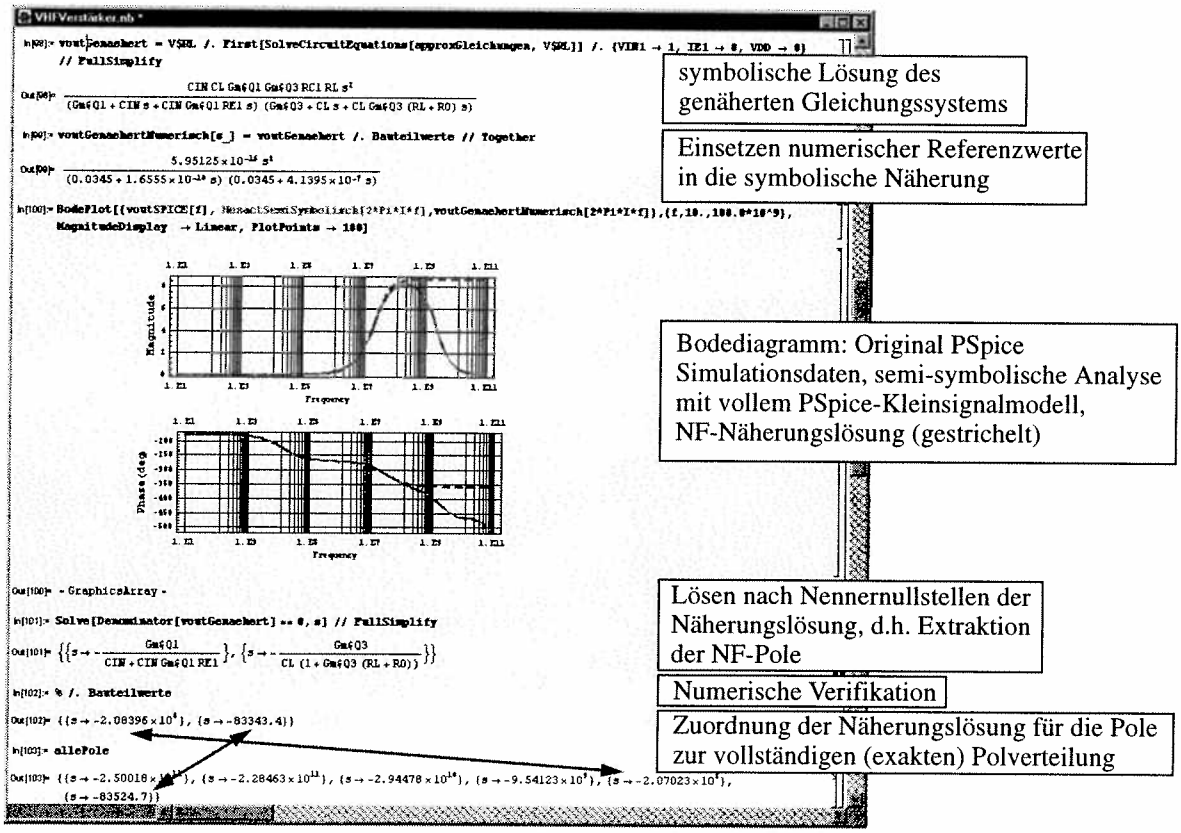


Abbildung 8: Auszug aus der Berechnung der unteren Grenzfrequenzen

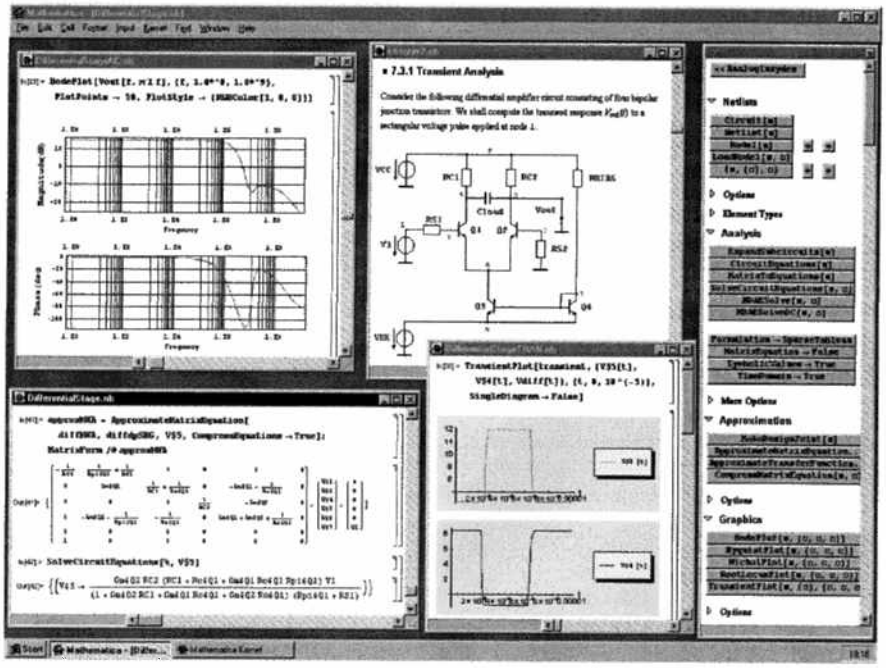


Abbildung 9: Bildschirmhardcopy Analog Insydes. Die einzelnen Fenster zeigen Ausschnitte aus dem Funktionsspektrum: Graphische Darstellungsmöglichkeiten (oben links), Import- und Dokumentationsmöglichkeiten (oben Mitte), Palette (rechts), symbolische Analyse (unten links), numerische Analyse (unten Mitte)

Erste Erfahrungen mit der Simulation von Mixed-Signal-Schaltungen mit einem VHDL-AMS-Simulator

Joachim Haase, Wolfgang Vermeiren
Fraunhofer-Institut für Integrierte Schaltungen, Außenstelle EAS Dresden
Zeunerstr. 38, 01069 Dresden, e-mail: [haase, vermeire]@eas.iis.fhg.de

1. Einleitung

Mit VHDL-AMS (analog und mixed-signal) steht seit diesem Jahr nach langjährigen Standardisierungsbemühungen eine Sprache zur Beschreibung digitaler, analoger und gemischt analog-digitaler Systeme zur Verfügung [WWW-1], [BLR-95] . Die Sprache erweitert den IEEE-Standard VHDL 1076-1993 [Bha-96], [Rei-98] um Elemente zur Modellierung des analogen Verhaltens elektrischer und nichtelektrischer Komponenten von der Bauelemente- bis zur Systemebene. Daneben wird festgelegt wie der mixed-mode Simulationszyklus abläuft. Dabei sind zur Berücksichtigung des analogen Verhaltens Algebrodifferentialgleichungssysteme wie bei Spice-kompatiblen Simulatoren auszuwerten. Diese Auswertung wird in die ereignisgesteuerte zeitdiskret arbeitende digitale VHDL-Simulation eingebettet.

Einer der wesentlichen Vorteile von VHDL-AMS ist die Möglichkeit zur Erstellung simulatorunabhängiger Modelle für digitale, analoge und gemischt analog-digitale Systeme. Wissen über derartige Systeme kann so langfristig dokumentiert werden. Der Austausch von Modellen und damit die Möglichkeit zum effektiveren Einsatz von Simulationsverfahren im Entwurfsprozeß wird vereinfacht. Die Wiederverwendbarkeit von Modellen und die Erweiterung des Einsatzbereiches von Simulatoren durch die Erstellung nutzereigener Modelle werden unterstützt.

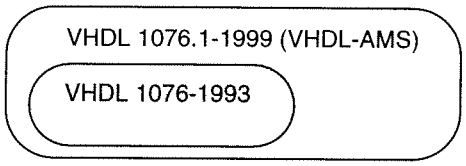


Bild 1: VHDL-AMS als Obermenge von VHDL 1076-1993

Die Anwendungsmöglichkeiten von VHDL-AMS reichen von der Simulation von Mixed-Signal-Schaltungen der Telekommunikation bis zur Modellierung von Anordnungen der Mikrosystemtechnik [Rom-98] . Ein anderer Einsatzfall ist die Einbeziehung des analogen Verhaltens der Umgebung in die Spezifikation von hauptsächlich digital arbeitenden ASIC's.

VHDL-AMS schließt das bewährte digitale VHDL ein (Bild 1). Auf die nun zusätzlich möglichen Beschreibungen analoger und analog-digitaler Systeme werden die Grundkonzepte von VHDL wie ENTITY/ARCHITECTURE-Methodik, Konfigurationen, Packages u. a. übertragen. Das erleichtert den Einstieg in die Nutzung von VHDL-AMS für die bisherigen Anwender von VHDL, verlangt von den mehr mit Spice und ähnlichen Simulationsprogrammen vertrauten Nutzern zunächst einige Gewöhnung. Dafür existiert mit VHDL-AMS nun aber auch eine Sprache, mit der das Verhalten analoger Systeme simulatorunabhängig beschrieben werden kann. Bisher verbreitete analoge Hardwarebeschreibungssprachen wie MAST [MaF-94] und HDL-A waren auf spezielle Simulatoren wie Saber bzw. ELDO zugeschnitten. Die von der Anwendung dieser Sprachen bekannten Prinzipien für die Modellierung rein analoger Systeme lassen sich mit leichten Modifikationen auf die Beschreibung rein analoger Systeme mit VHDL-AMS übertragen.

Eine neue Qualität wird bei der Beschreibung und Simulation gemischt analog-digitaler Schaltungen (Mixed-Signal-Schaltungen) erreicht. Einzelne Teilschaltungen oder auch Teile eines Modells können digital, andere analog beschrieben werden. Dafür stehen der Leistungsumfang des digitalen VHDL und der Spice-kompatibler Simulatoren, ergänzt um Beschreibungsmöglichkeiten für das analoge Verhalten zur Verfügung. Das erlaubt es, kompliziertere aber auch transparentere Modelle für diese Aufgagenklasse als mit Sprachen wie MAST und HDL-A zu erstellen. Beide Sprachen verfügen zwar auch zur Beschreibung gemischt analog-digitaler Probleme über einfache Sprachkonstrukte. Diese sind aber bei weitem nicht so mächtig wie die von VHDL-AMS.

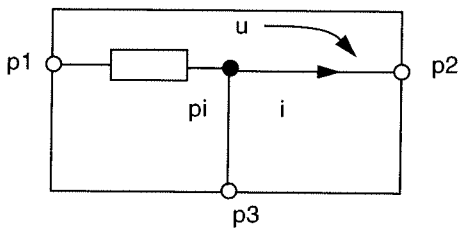
Mittlerweile sind erste leistungsfähige kommerzielle VHDL-AMS-Simulatoren angekündigt oder verfügbar [Gui-99] , [WWW-2], [VHD-99]. An der Weiterentwicklung dieser Werkzeuge wird intensiv gearbeitet. Im folgenden wird auf einige Erfahrungen bei der Modellierung und Simulation gemischt analog-digitaler Systeme mit VHDL-AMS eingegangen. Für Beispielrechnungen wurde ein VHDL-AMS-Simulator der Firma Mentor Graphics [VHD-99] verwendet.

2. Modellierungsprinzipien

2.1. Beschreibung analoger Teilsysteme

Kernstück der Erweiterungen von VHDL-AMS gegenüber VHDL 1076-1993 ist die Beschreibung des Verhaltens von Signalverläufen an den analogen Klemmen (TERMINAL's). An einer konservativen Klemme werden Fluß(THROUGH)- und Differenz(ACROSS)-Größe gegenüber dem zugehörigen Bezugsknoten berücksichtigt. Im Fall einer elektrischen Klemme ist die Flußgröße der Strom in die Klemme und die Differenzgröße die Spannung zwischen Klemme und Bezugsknoten. Klemmentypen können durch eine NATURE-Anweisung definiert werden. Die Definition der Klemmentypen gehört nicht zum Sprachumfang von VHDL-AMS. Überlicherweise werden diese Definitionen in entsprechenden Pakkages zusammengefaßt. Gegenwärtig sind diese Packages nicht standardisiert. Solange dies der Fall ist, muß der Nutzer bei der Verwendung fremder Modelle darauf achten, daß in diesen dieselben Definitionen für die Klemmentypen wie in seinen eigenen Modellen verwendet werden. Beim Typ ELECTRICAL wird es sicherlich keine Probleme geben. Anders kann es aber bei nichtelektrischen Klemmen aussehen. Nichtkonservativen Klemmen ist nur ein zeitkontinuierlicher Verlauf eines Signals zugeordnet. Die Anschlüsse regelungstechnischer Blöcke sind z. B. nichtkonservative Klemmen. Die nichtkonservativen Klemmen werden in Eingangs(IN)- und Ausgangs(OUT)-Klemmen unterteilt.

Bei der Beschreibung der ARCHITECTURE eines analogen Teilsystems wird von folgender Grundidee ausgegangen: Die Klemmen werden über im Modell zu definierende Zweige miteinander und eventuell dem zugehörigen Bezugsknoten verbunden. Falls erforderlich können auch innere Knoten eingeführt werden. Durch die Zweigdefinitionen werden Anfangs- und Endknoten der Zweige, Bezeichnungen für Zweigspannungen und -ströme und ihre Zählrichtungen festgelegt. Für jeden Zweig ist in der Modellbeschreibung eine Gleichung anzugeben. Durch die Gesamtheit dieser Gleichungen sind die Strom-Spannungs-Beziehungen der Zweige eines Modells festgelegt. Für jede nichtkonservative Ausgangsklemme ist eine Gleichung in das Modell aufzunehmen. Ist es nicht möglich, unter alleiniger Verwendung von Zweigströmen und -spannungen und den Signalen an den nichtkonservativen Klemmen das Verhalten eines Teilsystems zu beschreiben, können zusätzliche Hilfsgrößen (free QUANTITY's) deklariert und zur Beschreibung herangezogen werden. Für jede derartige Hilfsgröße ist in das Modell eine beschreibende Gleichung einzufügen. Solche Hilfsgrößen können beispielsweise bei der Verwendung von Ladungen und magnetischen Flüssen in einem Modell deklariert werden.



Anweisung in ENTITY-Beschreibung:

```
TERMINAL (p1, p2, p3 : ELECTRICAL);
```

Anweisungen in ARCHITECTURE-Beschreibung:

```
QUANTITY u ACROSS i THROUGH pi TO p2; -- Zweig
TERMINAL pi : ELECTRICAL; -- innerer Knoten
u == 1.0E3*i + ...;              -- u-i-Beziehung
```

Bild 2: Beispiel für Verwendung innerer Zeige und Knoten

Bild 2 illustriert an einem Beispiel das Vorgehen. p1, p2 und p3 sind elektrische Klemmen des Teilsystems. Im ENTITY-Block werden diese Klemmen als elektrische Klemmen deklariert. In der ARCHITECTURE-Beschreibung erfolgt die Deklaration des inneren Knotens pi. Zusätzlich ist beschrieben wie der den Knoten pi und die Klemme p2 verbindende Zweig deklariert wird. Seine Zweigspannung ist u, sein Zweigstrom i. Eine Modellgleichung beschreibt die u-i-Beziehung des Zweiges.

Das Beispiel in Bild 2 weist noch auf eine andere Möglichkeit hin. Klemmen und innere Knoten können auch unter Verwendung vorhandener Mehrpolmodelle (COMPONENTen) verbunden werden, im Beispiel durch einen Widerstand angedeutet. Die Verwendung einer COMPONENTe erfolgt wie in VHDL üblich.

Wünschenswert ist die Verwendung von Spice-Grundelementen wie Widerständen, Transistoren und Dioden in einem Modell. Das würde für simulatorunabhängige Modelle aber die Verfügbarkeit einer Bibliothek von VHDL-AMS-Modellen für diese Elemente erfordern. Eine derartige Bibliothek existiert gegenwärtig noch nicht. Der Anwender muß sich im Bedarfsfall mit der Erstellung eigener Modelle behelfen. Für konkrete Simulatoren existieren Sonderlösungen, die die Nutzung einzelner Spice-Elemente erleichtern [VHD-99].

2.2. Beschreibung analog-digitaler Teilsysteme

Bei der Beschreibung analog-digitaler Teilsysteme ist der definierte mixed-mode Simulationszyklus zu berücksichtigen [LiZ-97]. Der analoge (Differentialgleichungs-) Löser wird bei VHDL-AMS am Beginn jedes Simulationszyklus bis zum Zeitpunkt des nächsten digitalen Ereignisses abgearbeitet. Dabei können die aktuellen Werte der digitalen Signale gelesen werden. Danach werden die aktiven expliziten und impliziten digitalen Signale aktualisiert und alle nicht zurückgestellten Prozesse abgearbeitet. Bei der Bearbeitung der digitalen Anweisungen kann auf die Werte analoger Signale zugegriffen werden. Ein Delta-Zyklus wird zum selben Zeitpunkt abgearbeitet, wenn z. B. für ein Signal der Sensitivitätsliste eines Prozesses ein Ereignis registriert worden ist . Nach der Abarbeitung aller Delta-Zyklen werden die zurückgestellten Prozesse ausgewertet.

Die Änderung gegenüber dem VHDL-Simulationszyklus ist relativ gering. Der analoge Löser kann als eine Art Prozeß angesehen werden, der zu Beginn jedes Simulationszyklus abgearbeitet wird. Ist der analoge Löser aktiv, arbeitet er weitgehend unabhängig von der digitalen Signalbearbeitung. Er beeinflußt diese aber in jedem Fall durch die Änderung analoger Signale, die vom digitalen Teil lesend ausgewertet werden. Die Auswertung erfolgt jedoch erst zum Zeitpunkt des nächsten digitalen Ereignisses. Ist vorher eine Synchronisation von analogem und digitalem Teil gewünscht, muß bei der Bearbeitung des analogen Teils ein digitales Ereignis generiert werden.

Dazu kann durch den analogen Löser ein implizites digitales Signal q'ABOVE(e) beeinflußt werden. Dieses hat den Wert TRUE, wenn das analoge Signal q einen Wert größer als die Schwelle e annimmt und ist ansonsten FALSE. Um eine Synchronisation zu erzwingen, muß also eine Änderung eines Signals q'ABOVE(e), das beispielsweise in der Sensitivitätsliste eines Prozesses stehen kann, stattfinden. Weitere Einzelheiten sind in [Bak] zu finden.

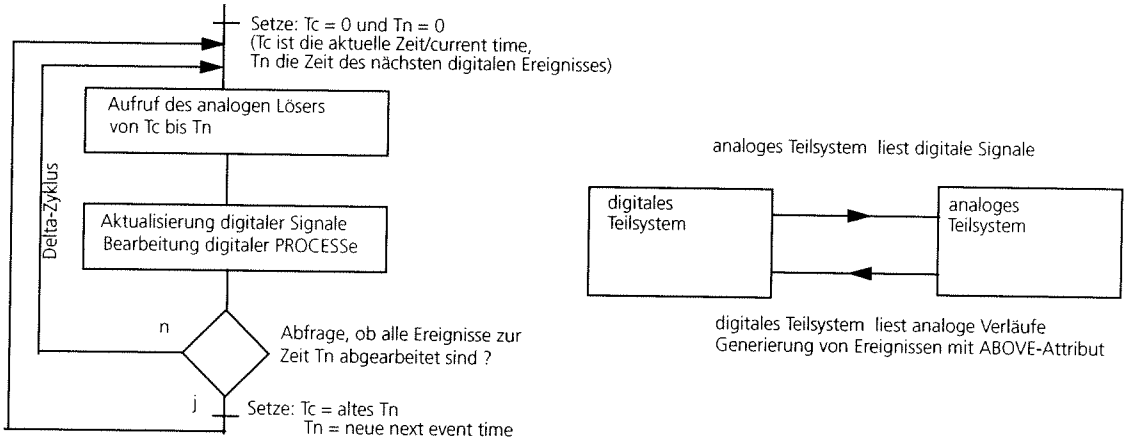


Bild 3: Prinzipieller mixed-mode Simulationszyklus und Signalaustausch zwischen Teilsystemen

Für die Modellierung hat dieser mixed-mode Simulationszyklus nach Erfahrung der Verfasser u. a. folgende Konsequenzen: Während der Initialisierungsphase wird zuerst der analoge Löser aufgerufen. Dabei wird auf die Initialisierungswerte der digitalen Signale, in der Regel deren Default-Werte, lesend zugegriffen. Probleme können sich u. a. ergeben, wenn reellwertige digitale Signale gelesen werden müssen. Diese sind standardmäßig mit einem unendlich großen negativen Wert initialisiert. Das kann zu Konvergenzschwierigkeit bei der Arbeit des analogen Lösers führen. Es macht sich deshalb erforderlich, ggf. für vom analogen Teilsystem zu lesende digitale, insbesondere reellwertige Signale verträgliche Anfangswerte (z. B. 0) vorzugeben.

Um eine Synchronisation der Auswertung von analogem und digitalem Teilsystem zu erzwingen, müssen eventuell unter Verwendung des ABOVE-Attributs digitale Signale generiert werden. Es kann nach Kenntnisstand der Autoren ansonsten nicht davon ausgegangen werden, daß Änderungen im Analogteil sich vor dem nächsten digitalen Ereignis auf das digitale Teilsystem auswirken.

Empfehlenswert ist bei der Erstellung von gemischt analog-digitalen Modellen die Trennung der Beschreibungen von analogem und digitalem Verhalten. Das erleichtert es, den Signalaustausch zwischen beiden Beschreibungen nachvollziehbar zu gestalten. Es hat sich bewährt, die Beschreibung des digitalen Verhaltens in einem PROCESS zusammenzufassen, wo das möglich ist. Dieser kann leicht durch ein q'ABOVE(e)-Signal in der Sensitivitätsliste aktiviert werden.

3. Beispiele

3.1. D/A-Wandler für Sigma-Delta-Wandler

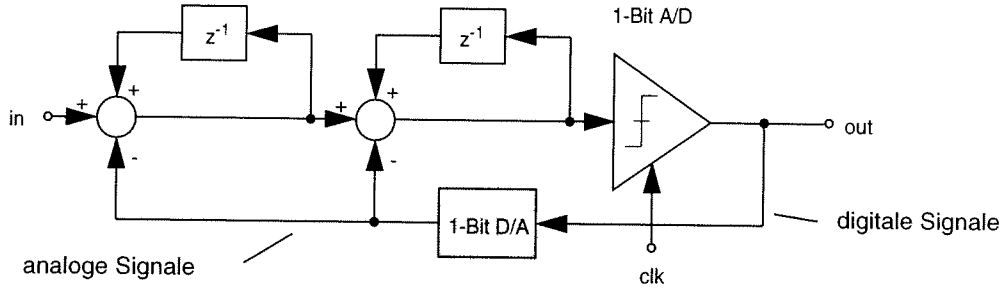
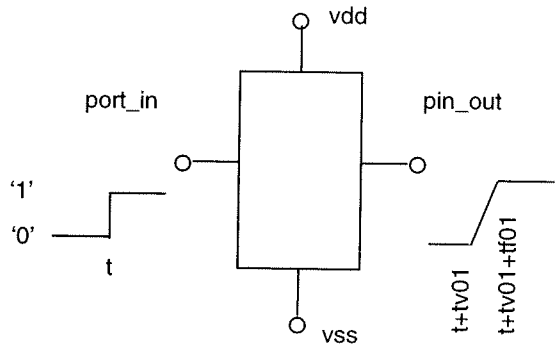


Bild 4: Blockschaltbild eines Sigma-Delta-Wandlers

Der in Bild 4 dargestellte Sigma-Delta-Wandler ist auf Blockniveau modelliert und mit sinusförmigem Eingang simuliert worden. Typische gemischt analog-digitale Komponenten sind der A/D- und der D/A-Wandler. Für den idealen 1-Bit-D/A-Wandler sind wesentliche Ausschnitte aus dem erstellten VHDL-AMS-Modell im folgenden angegeben.

```
ENTITY d2a_verz IS
       GENERIC ( tv_01     : TIME := 2 ns;          -- Verzoegerungszeiten und
                 tv_10     : TIME := 3 ns;
                 tf_01     : REAL := 5.0E-9;        -- Flankendauern
                 tf_10     : REAL := 6.0E-9);

       PORT (TERMINAL vdd, vss, pin_out    : ELECTRICAL;  -- analoge Klemmen
             SIGNAL   port_in              : IN UX01);    -- digital port
END ENTITY d2a_verz;

ARCHITECTURE v1 OF d2a_verz IS
...
-- Beschreibung von Klemmengroessen
QUANTITY  v_pin_out ACROSS  i_pin_out THROUGH  pin_out;
QUANTITY  v_pin_vss ACROSS  vss;
QUANTITY  vds       ACROSS  vdd  TO  vss;

-- Deklaration von Signalen, Konstanten, ...
SIGNAL    xout        : REAL    := 0.0;
CONSTANT  sample_time : TIME    := 10 ns;
...
BEGIN

PROCESS BEGIN
   WAIT ON port_in, signal_vds,  ...;
   ...
   IF DOMAIN = TIME_DOMAIN THEN
      IF port_in = '1' THEN
         xout <= INERTIAL signal_vds AFTER tv01;
       ELSE
         xout <= INERTIAL 0.0 AFTER tv10;
      END IF;
   END IF;
END PROCESS;

PROCESS BEGIN
   signal_vds <= vds;
   WAIT FOR sample_time;
END PROCESS;

IF DOMAIN /= TIME_DOMAIN USE
     v_pin_out - v_pin_vss == xout;
   ELSE
     v_pin_out - v_pin_vss == xout'Ramp (tf01, tf10);
END USE;
...
END ARCHITECTURE v1;
```



Die Ausgangsspannung an der elektrischen Klemme pin_out soll sich in Abhängigkeit vom logischen Signal am PORT port_in zeitverzögert ändert. Bei logischem 1-Signal soll die Ausgangsspannung gleich der Spannungsdifferenz zwischen vdd und vss sein. Dazu wird die Spannung zwischen den elek-

trischen Klemmen vdd und vss in festen Zeitintervallen abgetastet und dem Signal signal_vds zugeordnet. In Abhängigkeit von der Größe des logischen Signals am Port port_in wird bei Signaländerungen an port_in und von signal_vds nach einer durch tv01 oder tv10 festgelegten Verzögerungszeit das Signal xout gebildet. Innerhalb der Zeit tf01 oder tf10 wird nach Änderung von xout die Ausgangsspannung an der elektrischen Klemme pin_out auf den neuen Wert von xout geändert. Das Signal xout muß mit 0.0 initialisiert werden, da andernfalls bei der ersten Auswertung des Analogteils die Spannung an der Klemme pin_out unendlich klein werden müßte.

3.2. Beschreibung analog-digitaler Teilsysteme

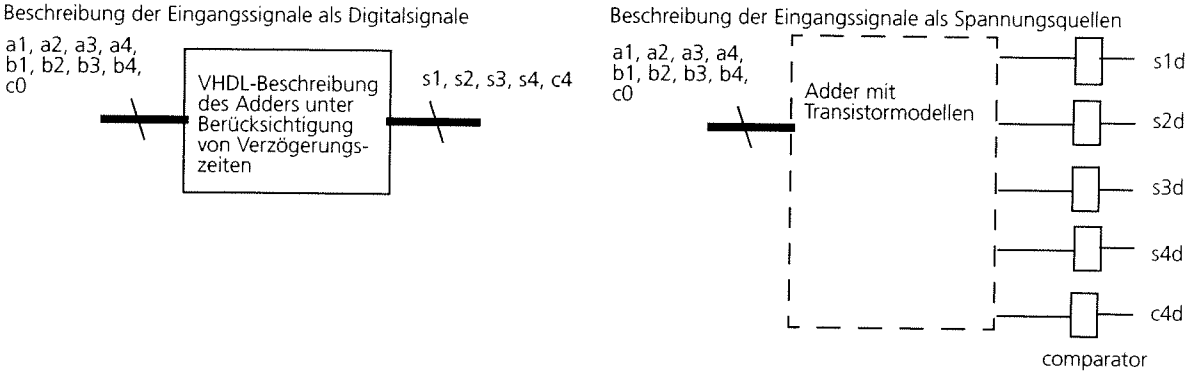
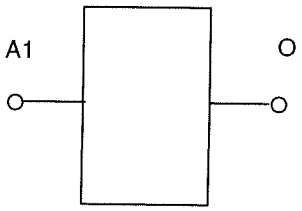


Bild 5: Schaltungsanordnungen für ein Beispiel zum Vergleich der Simulationsergebnisse auf Gatter- und Transistorebene

In Bild 5 sind zwei Schaltungsanordnungen angegeben, die demonstrieren wie ein VHDL-AMS-Simulator zum Vergleich der Genauigkeit einer Simulation auf Gatterebene mit der einer Simulation auf Transistorebene verwendet werden kann. Als Beispiel dient ein 4-Bit-Adder mit den Eingängen a und b und dem Übertrag c. Diese Adder-Schaltung wird zum einen auf Gatterebene und zum anderen auf Transistorebene simuliert. Es werden für beide Anordnungen die gleichen Eingangssignale verwendet. An die Ausgänge der Transistorschaltung werden Komparatoren geschaltet, deren VHDL-AMS-Modell im folgenden angegeben ist:

```
ENTITY comparator IS
      GENERIC (schwelle : REAL := 2.5 );
      PORT (   TERMINAL   A1 : ELECTRICAL;
               SIGNAL     O  : OUT STD_ULOGIC  := '0');
END comparator;

ARCHITECTURE ideal OF comparator IS

    QUANTITY   v1  ACROSS  A1;
    SIGNAL     S   : BOOLEAN;

BEGIN

    S <= v1'ABOVE (schwelle);

    PROCESS (S) BEGIN
      IF S = TRUE THEN
         O <= '1';
      ELSE
         O <= '0';
      END IF;
    END PROCESS;

END ideal;
```



elektrische Klemme (Leerlauf am Eingang)

PORT für logisches Ausgangssignal

Idealerweise müssen die Signale sx und sxd, sowie c0 und c0d übereinstimmen. Für die Beispielschaltung konnte das bestätigt werden. Aus Platzgründen muß auf die Angabe der Simulationserbenisse an dieser Stelle verzichtet werden. Durch den VHDL-AMS-Simulator, der sowohl eine Simulation auf Transistorebene als auch auf Gatterebene erlaubt und für die Ergebnisdarstellung dasselbe Ausgabetool verwendet, ist der Vergleich beider Simulationsergebnisse leicht möglich.

Das angegebene Komparatormodell stellt offenbar das einfachste gemischt analog-digitale VHDL-AMS-Modell dar. Es kann auch für andere ähnliche Vergleiche herangezogen werden.

4. Zusammenfassung und Ausblick

Durch die Verfügbarkeit von VHDL-AMS-Simulatoren kann jetzt ernsthaft damit begonnen werden, den Einsatz dieser Werkzeuge für komplexere Aufgaben vorzubereiten. VHDL-AMS vereint die Vorzüge digitaler VHDL- und Spice-kompatibler Simulatoren. Die Standardisierung der Sprache ist abgeschlossen, aber eine Reihe von Festlegungen müssen noch getroffen werden, um letztendlich wirklich zu simultorunabhängigen austauschbaren Modellen zu gelangen. Soweit bekannt, wird an der Lösung dieser Probleme gearbeitet.

Im gegenwärtigen Stadium dürfte eine wichtige Aufgabe darin bestehen, anhand ausgewählter umfangreicher Beispiele Möglichkeiten und Grenzen von VHDL-AMS und der auf ihr basierenden Simulatoren und anderen Werkzeuge zu testen. In diesem Zusammenhang können Prinzipien erarbeitet werden, die die Grundlage für den späteren effektiven Routineeinsatz bilden.

Erste Erfahrungen zeigen, daß sich VHDL-AMS insbesondere gut für die Simulation gemischt analog-digitaler Systeme im Zeitbereich eignet. Bei der Modellierung sind aber einige Besonderheiten zu berücksichtigen, die sich aus dem mixed-mode Simulationszyklus ergeben. Wie die dabei auftretenden Probleme überwunden werden können, ist im Beitrag, insbesondere in Abschnitt 2.2, diskutiert worden. Die aufgestellten Regeln sollten in Zusammenhang mit weiteren Untersuchungen präzisiert und um neue Empfehlungen erweitert werden.

Der Beitrag basiert auf Arbeiten, die im Rahmen des Themas "HDL's, Verhaltensmodellierung, Mixed-Signal-Modellierung und Simulation, Spezifikationserfassung" (Förderkennzeichen 01 M 30 39 B) im Förderschwerpunkt "Smart System Engineering" durchgeführt werden.

5. Literatur

[Bak] Bakalar, K.: Mixed-Mode Simulation. White Paper der 1076.1 Working Group (siehe [WWW-1])

[BLR-95] Berge, J.-M.; Levia, O.; Rouillard, J.: Modeling in Analog Design. Boston/Dordrecht/ London: Kluwer Academic Publishers, 1995.

[Bha-96] Bhasker, J.: Die VHDL-Syntax. Prentice-Hall, 1996.

[Gui-99] Guide to Mixed-Signal Simulation.Firmenschrift Analogy, Inc., 1999.

[MaF-94] Mantooth, H. A.; Fiegenbaum, M. F.: Modeling with an Analog Hardware Description Languag. Kluwer Academic Publishers, 1994.

[Rei-98] Reifschneider, N.: CAE-gestützte IC-Entwurfsmethoden. München: Prentice-Hall, 1998.

[Rom-98] Romanowicz, B. F.: Methodology for the Modeling and Simulation of Microsystems. Boston/Dordrecht/London: Kluwer Academic Publishers, 1998.

[LiZ-97] Litovski, V.; Zwolinski, M.: VLSI Circuit Simulation and Optimization. London: Chapman & Hall, 1997.

[VHD-99] VHDL-AMS Design Station User's Manual. Firmenschrift Mentor Graphics Corporation, 1998.

[WWW-1] Angaben im World Wibe Web unter http://www.vhdl.org/analog/

[WWW-2] Angaben im World Wide Web unter http://www.vhdl-ams.com/

Erstellung aggregierter Simulationsmodelle für die mittelfristige Produktionsplanung

Sven Völker
Technische Universität Ilmenau, Fachgebiet Wirtschaftsinformatik I
Postfach 100565, 98684 Ilmenau
email: sven.voelker@wirtschaft.tu-ilmenau.de

Kurzfassung

Um Simulationsmethoden in der mittelfristigen Produktionsplanung sinnvoll einzusetzen, werden - im Gegensatz zur Feinplanung - reduzierte Simulationsmodelle benötigt. In diesem Artikel wird ein Verfahren vorgestellt, mit dem auf Basis detaillierter Daten über das Produktionssystem reduzierte, datengetriebene Simulationsmodelle für die mittelfristige Produktionsplanung automatisch erzeugt werden. Die Grundidee der Erstellung reduzierter Modelle besteht darin, für den jeweiligen Planungszweck unkritische Arbeitsgänge im Simulationsmodell lediglich pauschal abzubilden. Die Zulässigkeit des dargestellten Verfahrens wird durch bedienungstheoretische Betrachtungen untermauert.

1 Einleitung

Die zentrale Aufgabe der mittelfristigen Produktionsplanung besteht in der Festlegung der Eckstart- und -endtermine der Fertigungsaufträge [Hoitsch 1993, S. 177 ff.]. Mit den gegenwärtig im Einsatz befindlichen Produktionsplanungs und -steuerungssystemen (PPS-Systemen) wird in diesem Bereich oft keine zufriedenstellende Planungsqualität erzielt. Am deutlichsten zeigt sich dies am sogenannten Durchlaufzeit-Syndrom [Corsten 1994, S. 445]. Durch eine zu frühzeitige Auftragsfreigabe steigen die Auftragsbestände im Produktionssystem stark an, was zu verlängerten Durchlaufzeiten und in der Folge zu Terminüberschreitungen führt. Auf die Terminverzüge wird mit einer noch früheren Auftragsfreigabe reagiert, wodurch sich das Problem weiter verschärft. Aus dem Durchlaufzeit-Syndrom resultiert die Tatsache, daß in Industrieunternehmen oft 80-95% der Auftragsdurchlaufzeiten Wartezeiten sind [Kurbel 1995, S. 154]. Dieser hohe Anteil an Wartezeiten ist mit entsprechend hohen Kapitalbindungskosten verbunden.

Die derzeit in der mittelfristigen Produktionsplanung bestehenden Probleme beruhen im wesentlichen auf der Verwendung unzureichender Planungsverfahren, die mit Modellen arbeiten, welche das reale Produktionssystem zu stark vereinfachen. Eine deutliche Verbesserung der Planungsqualität kann nur erreicht werden, wenn die Planung mit optimierenden Verfahren erfolgt, die realitätsnahe Simulationsmodelle zur Bestimmung der Güte von Planalternativen benutzen. Daß sich diese Erkenntnis auch in der Praxis durchsetzt, zeigt sich daran, daß Optimierungsverfahren, wie z. B. Simulated Annealing, zunehmend zur PPS eingesetzt werden [Nissen 1994, S. 295 ff., SynQuest 1998, S. 4]. Diese Verfahren sind jedoch mit dem Nachteil verbunden, daß relativ viele Planalternativen untersucht und somit auch viele Simulationsläufe durchgeführt werden müssen, um zu einem guten Planungsergebnis zu gelangen. Den sich aus der großen Anzahl von Simulationsläufen ergebenden Performance-Problemen kann durch eine Senkung des numerischen Aufwands für die Simulationsläufe begegnet werden.

Eine deutliche Senkung des Zeitbedarfs für Simulationsrechnungen ist nur möglich, wenn die Simulationsmodelle vereinfacht werden. Vereinfachte Modelle führen zwar zu weniger genauen Simulationsergebnissen; im Bereich der mittelfristigen Planung ist dies jedoch vertretbar, da hier ohnehin mit Zeit- und Kapazitätspuffern gearbeitet wird, die dafür sorgen, daß für die nachgelagerte Feinplanung noch ausreichende Freiheitsgrade bleiben. Diese Puffer fangen nicht nur Störungen ab, die zum Planungszeitpunkt noch unbekannt sind, sondern auch die Abbildungsfehler im Simulationsmodell.

Die Notwendigkeit vereinfachter Modelle zur Planung auf höheren Aggregationsstufen wird zwar allgemein anerkannt [Watson u. a. 1997, S. 767; Kosturiak u. a. 1995, S. 66], eine universelle Methode zur Erstellung derartiger Modelle wurde jedoch noch nicht entwickelt. Bisher werden die verschiedenen Aggregationsstufen von Simulationsmodellen meist nach der organisatorischen Gestaltung des Produktionssystems festgelegt. Dabei bildet ein Element eines Simulationsmodells ein Werk, eine Werkstatt, eine Maschinengruppe oder eine einzelne Maschine ab. Da Produktionssysteme in der Regel nicht nach dem Bedienungsverhalten der Kapazitätseinheiten, sondern nach zusammengehörenden Technologien oder feststehenden Arbeitsgangfolgen organisiert werden [Corsten 1994,

S. 32 ff.], kann bei gegebenem Reduktionsgrad nicht das Maximum an Genauigkeit der Simulationsergebnisse erreicht werden, wenn das Simulationsmodell nur nach organisatorischen Gesichtspunkten reduziert wird. Deshalb sollte sich ein Verfahren zur Modellreduktion daran orientieren, welche Bedeutung die einzelnen Elemente des Produktionssystems für die Planung haben. In diesem Artikel wird ein solches Verfahren vorgestellt.

2 Ein Verfahren zur Erstellung reduzierter Simulationsmodelle

In diesem Abschnitt wird ein Verfahren skizziert, mit dem auf Basis detaillierter Daten über das Produktionssystem reduzierte, datengetriebene Simulationsmodelle für die mittelfristige Produktionsplanung automatisch erzeugt werden können.

Die Grundidee der Reduktion besteht darin, für den jeweiligen Planungszweck unkritische Arbeitsgänge im Simulationsmodell pauschal zu betrachten, um dadurch den numerischen Aufwand für Simulationsläufe zu senken. Die Pauschalisierung wirkt sich sowohl auf die dynamischen Elemente des Simulationsmodells (die Fertigungsaufträge) als auch auf die statischen Elemente des Modells (die Kapazitätseinheiten) aus. Daher werden im Anschluß folgende Fragen diskutiert:

- Welche Arbeitsgänge werden pauschalisiert?
- Wie erfolgt die Pauschalisierung auf der Ebene der dynamischen Elemente des Simulationsmodells?
- Wie erfolgt die Pauschalisierung auf der Ebene der statischen Elemente des Simulationsmodells?

Auswahl zu pauschalisierender Arbeitsgänge

Die Auswahl der zu pauschalisierenden Arbeitsgänge erfolgt mit dem Ziel, in der Menge aller auszuführenden Arbeitsgänge diejenigen zu identifizieren, auf deren explizite Simulation verzichtet werden kann, ohne daß die Simulationsergebnisse für Planungszwecke zu ungenau werden. Bei dieser Auswahl ist zu berücksichtigen, daß die Arbeitsgänge um Kapazitätseinheiten konkurrieren und sich daher gegenseitig beeinflussen. Der tatsächliche Einfluß der pauschalisierten Arbeitsgänge auf den Produktionsprozeß kann nur geschätzt werden. Deshalb muß die Pauschalisierung entweder so vorgenommen werden, daß dieser Einfluß möglichst gering ist, oder so, daß er möglichst genau geschätzt werden kann. Daraus resultieren folgende zwei Ansätze zur Pauschalisierung:

⇒ Pauschalisierung nach dem Einfluß der Arbeitsgänge auf den Produktionsprozeß:
Da es nicht möglich ist, die tatsächlichen Auswirkungen, welche die Ausführung eines Arbeitsgangs auf den Produktionsverlauf hat, vor Ausführung der Simulation exakt zu bestimmen, werden Indikatoren benutzt, um diejenigen Arbeitsgänge zu ermitteln, von denen zu vermuten ist, daß sie nur einen geringen Einfluß auf den Fertigungsprozeß ausüben. Ein Beispiel für einen derartigen Indikator ist die Bedienungszeit: Es werden die kürzesten Arbeitsgänge pauschalisiert, weil diese nur eine relativ geringe Kapazitätsnachfrage nach sich ziehen und deshalb eine untergeordnete Bedeutung für das Produkionsgeschehen besitzen.

⇒ Pauschalisierung durch proportionale Verkleinerung des Simulationsmodells:
Dieser Ansatz zielt darauf ab, eine proportionale Verkleinerung des Simulationsmodells zu erreichen, also eine, bei der die Verteilungen der relevanten Zufallsgrößen unverändert bleiben. Beispielsweise kann die Pauschalisierung so vorgenommen werden, daß die Verteilung der Bedienungszeiten über der Menge der explizit zu simulierenden Arbeitsgänge mit der Verteilung der Bedienungszeiten über der Menge aller auszuführenden Arbeitsgänge übereinstimmt.

Welcher der alternativen Ansätze zu genaueren Simulationsergebnissen führt, soll eine umfangreiche empirische Untersuchung klären.

Im Hinblick auf die Qualität und Zuverlässigkeit der Planung müssen nicht alle Fertigungstermine mit gleicher Genauigkeit prognostiziert werden, da sie aus betriebswirtschaftlicher Sicht eine unterschiedliche Bedeutung aufweisen. Aus diesem Grund bietet sich ein dritter Ansatz an, der mit den bereits genannten Ansätzen zur Auswahl der Arbeitsgänge kombiniert werden kann:

⇒ Pauschalisierung nach betriebswirtschaftlich-planerischen Gesichtspunkten:
Ähnlich wie im ersten Ansatz werden Indikatoren genutzt, um die Bedeutung der Arbeitsgänge zu quantifizieren. Als Beispiel für einen solchen Indikator diene die Kundenpriorität: Arbeitsgänge, die zu Aufträgen niedrig priorisierter Kunden gehören, werden in der Simulation nur pauschal berücksichtigt.

Ergebnis der Auswahl zu pauschalisierender Arbeitsgänge ist die Zerlegung der Menge aller auszuführenden Arbeitsgänge in zwei disjunkte Teilmengen, die Menge der explizit zu simulierenden Arbeitsgänge und die Menge der pauschal abzubildenden Arbeitsgänge. Welche Konsequenzen diese Differenzierung für die Elemente des Simulationsmodells hat, wird im folgenden diskutiert.

Pauschalisierung auf der Ebene der dynamischen Systemelemente

Die dynamischen Elemente der Simulationsmodelle sind die Fertigungsaufträge. Sie werden im wesentlichen durch ihre Arbeitspläne charakterisiert. Die Reduktion der Modelle erfolgt, indem einige Arbeitsgänge aus den zu den Fertigungsaufträgen gehörenden Arbeitsplänen entfernt werden. An ihre Stelle tritt die Angabe von Zeitpuffern, die pauschale Übergangszeiten repräsentieren. Die Zeitpuffer beinhalten die Bedienungszeiten der pauschalisierten Arbeitsgänge sowie die Wartezeiten, die vor der Ausführung dieser Arbeitsgänge verstreichen. Die Bedienungszeiten sind Bestandteil der PPS-Daten und damit bekannt. Die Wartezeiten sind Zufallsgrößen, deren Erwartungswert und deren Varianz sich im Prozeßverlauf ergeben. Ihre Erwartungswerte werden nach bedienungstheoretischen Gesetzmäßigkeiten auf der Grundlage der Kapazitätsauslastung approximiert und fließen in die Zeitpuffer ein. Die geschätzten Varianzen der Wartezeiten können genutzt werden, um den durch die Modellreduktion entstehenden Abbildungsfehler zu quantifizieren. Abbildung 1 zeigt ein Beispiel für die Reduktion eines Arbeitsplanes mittels Substitution ausgewählter Arbeitsgänge durch Zeitpuffer.

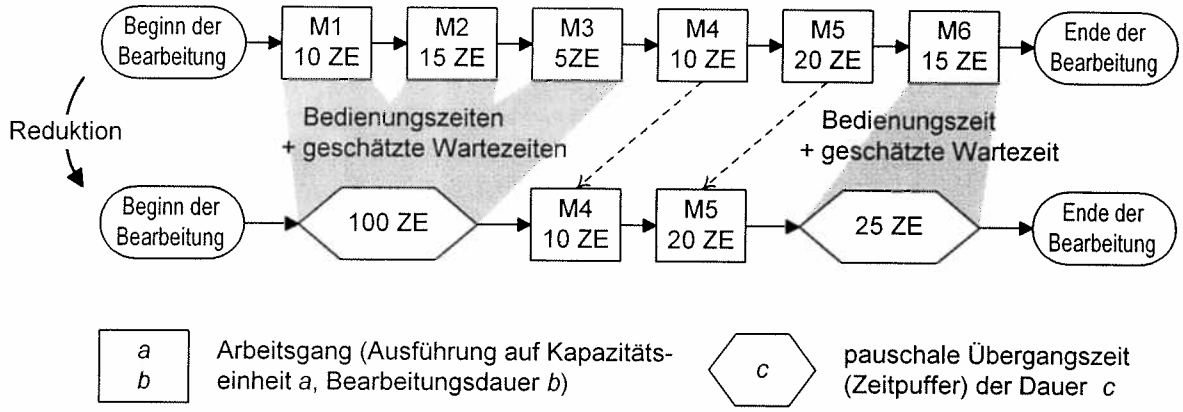


Abbildung 1: Reduktion eines Arbeitsplanes durch die Einführung von Zeitpuffern

Während des Simulationslaufes werden pauschalisierte Arbeitsgänge durch eine Verzögerung der Aufträge gemäß des Umfangs der sie repräsentierenden Zeitpuffer abgebildet. Da dabei die explizite Betrachtung des Warte- und Bedienungsprozesses entfällt und mehrere aufeinanderfolgende pauschalisierte Arbeitsgänge zusammengefaßt werden können, wird für den Simulationslauf deutlich weniger Zeit als bei einer detaillierten Betrachtung benötigt.

Die Berücksichtigung der Erwartungswerte der Wartezeiten bei der Bestimmung der Zeitpuffer ist offensichtlich nur möglich, wenn die mittleren Wartezeiten keinen zu starken zeitlichen Schwankungen unterworfen sind und wenn diese Werte bekannt sind. Es wird vorausgesetzt, daß sich das Produktionssystem oder zumindest Teile des Produktionssystems innerhalb des gesamten Simulationszeitraums in hinreichend langen Zeitabschnitten in einem annähernd stationären Zustand befinden. Die Modellreduktion konzentriert sich auf diese Zeitabschnitte.

Pauschalisierung auf der Ebene der statischen Systemelemente

Die beschriebene Reduktion der Arbeitspläne führt dazu, daß im Simulationsmodell weniger Arbeitsgänge abgebildet werden, als im realen Produktionssystem auszuführen sind. Demzufolge werden die zur Verfügung stehenden Kapazitäten im Simulationsmodell weniger belastet als in der Realität. Um diese Abweichung auszugleichen, muß die modellierte Kapazität der Kapazitätseinheiten entsprechend dem Umfang der Reduktion der Arbeitspläne verringert werden.

Dies geschieht, indem im Simulationsmodell Kapazitäten für die Ausführung der pauschalisierten Arbeitsgänge reserviert werden. Die Kapazitätsreservierungen werden periodengenau bestimmt. Sinnvoller- aber nicht notwendigerweise können die Perioden logische Einheiten (z. B. Tage oder Schichten) sein. Innerhalb der Perioden wird jede von der Modellreduktion betroffene Kapazitätseinheit für einen bestimmten Zeitraum blockiert, so daß sie nicht für die Abarbeitung von Aufträgen zur Verfügung steht (siehe Abbildung 2).

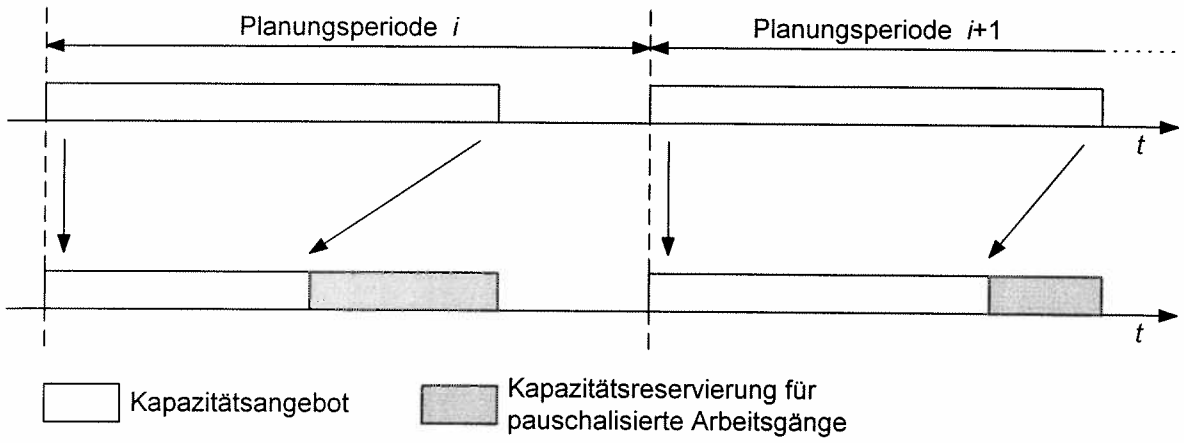


Abbildung 2: Absenken des Kapzitätsangebotes einer Kapazitätseinheit durch periodengenaue Kapazitätsreservierungen

Der Aufwand und die Methode zur Bestimmung des Umfangs der Kapazitätsreservierungen hängen von den verwendeten Kriterien bei der Reduktion der Arbeitsgangmenge ab. In Spezialfällen kann sich die Bestimmung der Kapazitätsreservierungen bis zur Trivialität vereinfachen. Im ungünstigsten Fall müssen die Kapazitätsreservierungen für jede Planungsperiode und jede Kapazitätseinheit separat berechnet werden. Der Umfang der Kapazitätsreservierung für eine bestimmte Kapazitätseinheit in einer bestimmten Planungsperiode entspricht den kumulierten Bearbeitungszeiten derjenigen pauschalisierten Arbeitsgänge, die auf dieser Kapazitätseinheit in dieser Planungsperiode bearbeitet werden müssen.

Durch die Abstimmung der Zeitpuffer in den Arbeitsplänen und der Kapazitätsreservierungen wird erreicht, daß die für die verbleibenden Arbeitsgänge simulativ ermittelten Fertigungstermine zuverlässige Schätzungen für die bei vollständiger Simulation ermittelten Termine sind. Daß es möglich ist, eine solche Abstimmung vorzunehmen, wird im nachfolgenden Abschnitt auf der Grundlage der Bedienungstheorie gezeigt.

3 Bedienungstheoretische Betrachtung zur Modellreduktion

In diesem Kapitel wird die Anwendung des vorgestellten Verfahrens auf lineare stochastische Bedienungsnetzwerke [siehe z. B. Kelly u. a. 1995] analysiert. Es wird die Frage beantwortet, ob es möglich ist, die Verweilzeiten der Aufträge an den Kapazitätseinheiten des Produktionssystems trotz der Modellreduktion korrekt zu bestimmen.

Die Analyse von Bedienungsnetzwerken erfolgt üblicherweise in den drei Schritten [Häfner 1992, S. 105]

1. Dekomposition des Netzwerkes,
2. Analyse des Warteschlangensystems und
3. Rekomposition des Netzwerkes.

Da beim vorliegenden Problem lediglich der zweite Schritt - die Analyse des Warteschlangensystems - Besonderheiten aufweist, wird hier nur auf diesen Punkt eingegangen. Die Betrachtungen beschränken sich also auf ein einzelnes Warteschlangensystem. Bezüglich der Dekomposition und Rekomposition von Bedienungsnetzwerken sei auf die Literatur [z. B. Häfner 1992, S. 103 ff.] verwiesen.

Es wird im folgenden unterstellt, daß sich die Kapazitätseinheiten des zu simulierenden Produktionssystems als Warteschlangensysteme vom Typ M/M/*s* [zur Symbolik siehe z. B. Ullmann 1996, S. 81 f.] modellieren lassen. Die Forderungen entsprechen den Fertigungsaufträgen, die an der betrachteten Kapazitätseinheit die Abarbeitung eines ihrer Arbeitsgänge verlangen. Die Zwischenankunftszeiten der Forderungen seien exponentialverteilt mit dem Parameter λ. Das Entfernen einer Forderung aus dem Ankunftsstrom entspricht dem Pauschalisieren genau eines Arbeitsgangs. Die Wahrscheinlichkeit, mit der eine beliebige Forderung aus dem Ankunftsstrom entnommen wird, betrage p_{red}. Wenn die Auswahl der zu pauschalisierenden Forderungen unabhängig von der Zwischenankunftszeit ist, sind die Zwischenankunftszeiten der nach der Reduktion verbleibenden Forderungen exponentialverteilt mit dem Parameter $\lambda'=(1-p_{red})\lambda$ [Häfner 1992, S. 109].

Für die Terminplanung müssen die Verweilzeiten der Forderungen an den Bedienungsstationen bestimmt werden. Die Verweilzeit beginnt jeweils mit dem Eintreffen einer Forderung an einer Bedienungsstation und endet, wenn die Forderung die Station verläßt. Damit umfaßt sie sowohl die Warte- als auch die Bedienungszeit. Die Verweilzeit einer Forderung an einer bestimmten Bedienungsstation ist eine Zufallsgröße, deren Erwartungswert mittels bedienungstheoretischer Gesetzmäßigkeiten bestimmt werden kann [siehe z. B. Gross u. a. 1998, S. 69 ff.]. Für die aus dem Ankunftsstrom entfernten Forderungen wird eine Verweilzeit unterstellt, welche diesem Erwartungswert entspricht. Dieser Wert wird im Simulationsmodell als Zeitpuffer in den Arbeitsplan eingefügt. Damit ergibt sich für die Verweilzeit der pauschalisierten Arbeitsgänge im reduzierten Simulationsmodell gegenüber dem vollständigen Simulationsmodell eine mittlere Abweichung von Null.

Die im Ankunftsstrom verbleibenden Forderungen werden explizit simuliert. Allerdings führen die oben beschriebenen Kapazitätsreservierungen zu einer verlangsamten Abarbeitung der Forderungen (siehe Abschnitt 2). Diese Verlangsamung wird im bedienungstheoretischen Modell als Senkung der Bedienungsrate von μ auf μ' modelliert. Der Umfang, in dem die Bedienungsrate zu verringern ist, hängt von der Reduktionswahrscheinlichkeit p_{red} und der Anzahl der Bedienungskanäle s ab. Die reduzierte Bedienungsrate μ' kann so festgelegt werden, daß der Erwartungswert der Verweilzeit für das Warteschlangensystem M/M/s mit den Parametern λ und μ identisch mit diesem Wert für die Parameter λ' und μ' ist.

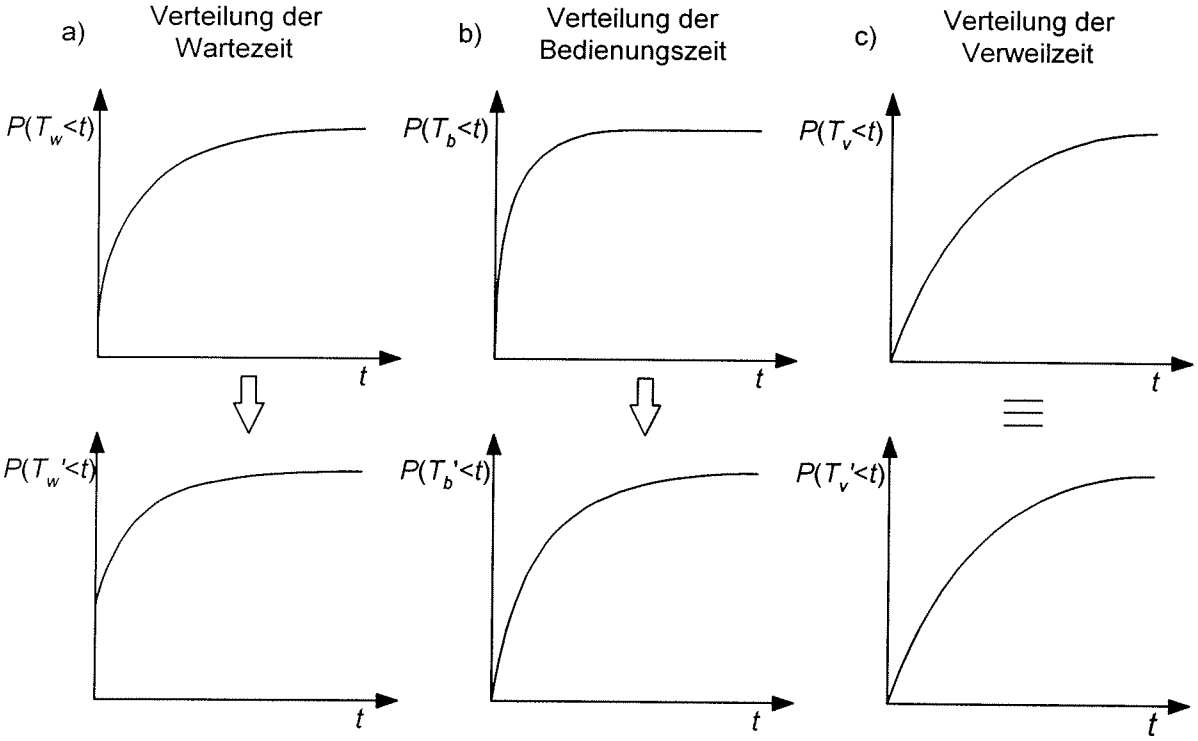


Abbildung 3: Einfluß der Modellreduktion auf die Verteilungen der Warte-, Bedienungs- und Verweilzeit

Abbildung 3 zeigt die Auswirkungen der Substitutionen $\lambda \rightarrow \lambda'$ und $\mu \rightarrow \mu'$ für den einfachsten Fall s=1. Obwohl sich die Verteilungen der Warte- und Bedienungszeiten verschieben, bleibt hier nicht nur der Erwartungswert, sondern die gesamte Verteilung der Verweilzeit erhalten:

Verteilung der Wartezeit: $P(T_W<t) \rightarrow P(T_W'<t)$

Verteilung der Bedienungszeit: $P(T_b<t) \rightarrow P(T_b'<t)$

Verteilung der Verweilzeit: $P(T_V<t) = P(T_V'<t)$

Dabei ergibt sich die reduzierte Bedienungsrate aus Beziehung $\mu'=\mu-p_{red}\lambda$. Auf die Herleitung dieser Gleichung wird hier aus Platzgründen verzichtet.

Da für die Terminplanung ausschließlich die Verweilzeit von Interesse ist und diese sowohl für die pauschalisierten als auch für die exlizit simulierten Arbeitsgänge im Mittel korrekt geschätzt wird, ist die Modellreduktion unter den obengenannten Bedingungen zulässig.

4 Zusammenfassung

In diesem Beitrag wurde das grundlegende Prinzip eines Verfahrens zur automatischen Erstellung reduzierter Simulationsmodelle für die mittelfristige Produktionsplanung vorgestellt. Die Zulässigkeit des Ansatzes wurde bedienungstheoretisch nachgewiesen. Ergänzend zu den bedienungstheoretischen Betrachtungen ist im Rahmen weiterführender Arbeiten vor allem eine umfassende empirische Untersuchung des Verfahrens vorgesehen.

Das entwickelte Reduktionsverfahren ist mit wesentlichen Vorteilen verbunden. Der Zielkonflikt zwischen kurzer Rechenzeit und hoher Genauigkeit der Simulationsergebnisse wird durch eine intelligente Reduktionsstrategie entschärft, die eine größtmögliche Reduktion des Simulationsmodells bei geringstmöglichem Informationsverlust anstrebt. Durch die freie Skalierbarkeit der Modellgenauigkeit kann die derzeit bestehende Lücke zwischen Kurz- und Mittelfristplanung geschlossen werden. Die Reduktion erfolgt situationsabhängig und erzeugt Simulationsmodelle, die auf das jeweilige Planungsproblem zugeschnitten sind. Die durch die Modellreduktion erzielte Einsparung an Rechenzeit erlaubt es, im Rahmen iterativer, simulationsbasierter Optimierungsverfahren eine größere Anzahl von Planalternativen zu untersuchen und dadurch bessere Planungsergebnisse zu erreichen. Auf diesem Weg trägt die Modellreduktion entscheidend zu einer Verbesserung der Planungsqualität bei.

Literatur

Corsten 1994 — Corsten, H.: Produktionswirtschaft - Einführung in das industrielle Produktionsmanagement, Oldenbourg, München u. a. 1994.

Gross u. a. 1998 — Gross, D., Harris, C. M.: Fundamentals of queueing theory, 2. Auflage, Wiley, New York u. a. 1998.

Häfner 1992 — Häfner, H.: Ein Warteschlangenansatz zur integrierten Produktionsplanung, Physica, Heidelberg 1992.

Hoitsch 1993 — Hoitsch, H.-J.: Produktionswirtschaft: Grundlagen einer industriellen Betriebswirtschaftslehre, 2. Auflage, Vahlen, München 1993.

Kelly u. a. 1995 — Kelly, F. P., Williams, R. J. (Hrsg.): Stochastic Networks, Springer, New York u. a. 1995.

Kosturiak u. a. 1995 — Kosturiak, J., Gregor, M.: Simulation von Produktionssystemen, Springer, Wien u. a. 1995.

Kurbel 1995 — Kurbel, K: Produktionsplanung und -steuerung, 2. Auflage, Oldenbourg, München u. a. 1995.

Nissen 1994 — Nissen, V.: Evolutionäre Algorithmen: Darstellung, Beispiele, betriebswirtschaftliche Anwendungsmöglichkeiten, Deutscher Universitäts-Verlag, Wiesbaden 1994.

SynQuest 1998 — SynQuest (Hrsg.): SynQuest Optimized Planning and Optimized Scheduling, Arbeitspapier, 1998.

Ullmann 1996 — Ullmann, K.-M.: Modellgestützte Produktionsplanung und -steuerung in einem stochastischen Produktionsnetzwerk: dargestellt an einem Ansatz zur Planung und Steuerung der innerbetrieblichen Briefbearbeitung bei der Deutschen Post AG, Lang, Frankfurt a. M. u. a. 1995.

Watson u. a. 1997 — Watson, E., Medeiros, D., Sadowski, R.: A simulation-based backward planning approach for order-release; in: 1997 Winter Simulation Conference Proceedings, S. 765-772.

Studentischer Arbeitsplatz zur Simulation busorientierter Systemtechnik in Gebäuden

M. Baumann I Chr. Burger I H.-J. Fiedler I G. Hohmann
Bauhaus- Universität Weimar, Fakultät Bauingenieurwesen
Professur Computergestützte Techniken
Coudraystrasse 13A, D- 99421 Weimar

1. Zusammenfassung

In neuerer Zeit ist auch im Wohnbau eine starke Zunahme elektrisch gesteuerter Funktionen zu verzeichnen. Neben Lösungen, die den Wohnkomfort erhöhen, tragen neue Anwendungen in den Bereichen Umwelt- und Energietechnik zu einem ständig steigenden Bedarf an Leitungswegen zur Energieversorgung der Verbraucher und zum funktionellen Verbund der Sensoren und Aktoren bei. Die entstehenden Lösungen sind Insellösungen, deren nachträgliche Integration zu einem Gesamtsystem sehr selten und wenn, nur mit großem Aufwand gelingt. Der Einsatz modifizierter, ursprünglich für die Prozeßautomatisierung entwickelter Bussysteme, ermöglicht eine Vermeidung der genannten Nachteile. Eine steigende Zahl von Anwendungsfällen dieser Systemtechnik ist auch im Wohnbau zu beobachten. Dabei berichten die am Ende der Planungsphase beteiligten Fachplaner für die Elektroinstallation, daß in dieser Phase entstehende Wünsche der Bauherren nur noch unter Überschreitung der vorgegebenen Bausumme berücksichtigt werden können. Dies könnte vermieden werden, wenn schon in frühen Planungs- und Entwurfsphasen die Potentiale der Gebäudesystemtechnik berücksichtigt würden. Um Architektur- und Bauingenieurstudenten dafür zu befähigen, müssen entsprechende Lehrinhalte in die Ausbildung aufgenommen und Möglichkeiten für den praktischen Erwerb von Applikationskenntnissen geschaffen werden. Die Professur Computergestütze Techniken an der Bauhaus- Universität Weimar macht mit der Lehrveranstaltung "Computergestützte Bauwerkssteuerung und Gebäudemanagement" entsprechende Lehrangebote. Ein mit wesentlichen Buskomponenten und der erforderlichen Projektierungs-, Test- und Inbetriebnahmesoftware ausgerüsteter Arbeitsplatz ermöglicht den Studierenden die simulative Erprobung sowohl der Einzelkomponenten als auch deren Zusammenwirken im Gesamtsystem. Bei fachübergreifender Zusammenarbeit von Studierenden der Bauhaus- Universität Weimar ist ein Projekt "Studentisches Wohnen" entstanden, bei dem der Einsatz der Gebäudesystemtechnik integraler Bestandteil ist.

2. Hintergründe

Seit der Nutzung elektrischer Energie werden die elektrischen Verbraucher direkt über Schalter oder Schütze mit Energie versorgt. Die steigenden Ansprüche an den Wohnkomfort, an Wohlbefinden, intelligente Alltagsbewältigung (z.B. Zeiteinsparung) aber auch ein wachsendes Umwelt- und Energiebewußtsein und neue technische Möglichkeiten wie beispielsweise:

- thermische Sonnenenergienutzung
- Photovoltaik
- Kleinst- Wärme- Kraftkopplungsanlagen
- Brennwertheiztechnologie
- Wand- und Strahlungsheizungen
- Wasser- und Windkraftnutzung
- Regen- und Brauchwassernutzung
- Kommunikationsanlagen

führen zu einer starken Zunahme der elektrisch gesteuerten Funktionen im Zweck- und im Wohnbau, die bei konventioneller Ausführung der Elektroinstallion zu einem erheblichen Schaltungs- und Verdrahtungsaufwand führen. Bild 1 zeigt am Beispiel einer einfachen Wechselschaltung die Verdrahtung beim Einsatz konventioneller Technik im Vergleich zur Systemtechnik.

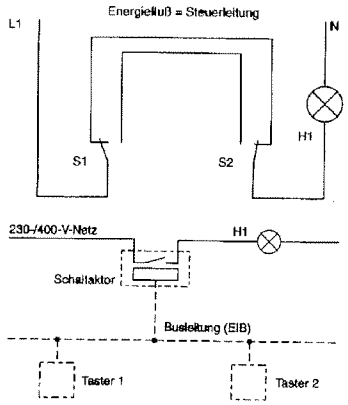


Bild 1: Wechselschaltungen: oben: konventionell, unten: Bussystem
(Quelle: Scherg, „EIB planen und installieren", Vogel 1995)

Neu entwickelte Geräte verfügen notwendigerweise über Sensoren und motorisch betriebene Stellglieder, die zusätzliche Datenleitungen erfordern. Trotz dieser bereits vorhandenen Datenleitungen ermöglichen die in der Regel firmenspezifischen Lösungen keine Einbindung in ein Gesamtsystem. In separaten, starren Datennetzen sind funktionell redundante Anordnungen von Sensoren, Aktoren und Steuerungseinheiten die Folge. Die Anpassung an ein sich während der Nutzungsphase als wünschenswert erweisendes Betriebskonzept erfordert in der Regel aufwendige Nach- oder Neuinstallationen.

Die technische Entwicklung, insbesondere auf dem Gebiet der Informationsverarbeitung (low cost computing power) eröffnet heute neue und sehr weitreichende Möglichkeiten, Menschen auch in ihrem Wohnumfeld von aufwendigen und lästigen Alltagsaufgaben zu entlasten. Die unmittelbar in den Sensoren und Aktoren lokalisierbare Intelligenz in Form von Microcontrollern, kommuniziert über eine Anwenderschnittstelle mit dem jeweiligen Anwendungsmodul (z.B. Tastsensor) und über einfache und billige Medien (twisted pair) und Verfahren zur Informationsübertragung untereinander. Dadurch stehen Sensorinformationen allen Netzteilnehmern zur Verfügung und die Energielastschalter lassen sich in oder in unmittelbarer Nähe der elektrischen Verbraucher anordnen. Durch geeignete Adressierungs-verfahren sind Zuordnungen der Sensoren zu den Aktoren frei und durch Umprogrammierung veränderlich wählbar. Erweiterten Ansprüchen an die Haustechnik, die während der Nutzungsphase entstehen, kann durch Nachrüstung leicht Rechnung getragen werden. Durch Modulation der Datentelegramme auf das bestehende 230/400V- Versorgungsnetz oder durch Funkmanagement-systeme kann das Regime im Gebäudebestand auch ohne Busleitung realisiert werden.

3. Die Ausbildung

Die Lehrveranstaltung "Computergestützte Bauwerkssteuerung und Gebäudemanagement" vermittelt Kenntnisse zu Methoden und Systemen, die insbesondere in der Nutzungsphase von Gebäuden eingesetzt werden. Auf diesen Kenntnissen aufbauend können Schlußfolgerungen für die Projektierungsphase abgeleitet werden. Die Gebäudesystemtechnik wird bei den Betrachtungen als der echtzeitfähige Teil eines hierarchisch angeordneten Gesamtsystems angesehen. Auf der Ebene der Gebäudesystemtechnik anfallende Informationen werden in übergeordneten Systemen (Gebäudeinformationssystemen oder Heimelektroniksystemen) für strategische Zielstellungen wie z.B. der Energieverbrauchsoptimierung verwendet. In dem Teil der Lehrveranstaltung, der sich mit der unmittelbaren Bauwerkssteuerung beschäftigt, werden die theoretischen und technischen Grundlagen der Meß- und Steuerungstechnik behandelt, die für das Verständnis der Probleme erforderlich sind. Abschließend wird der Europäische Installationsbus (EIB), als dezentrales, ereignisgesteuertes Bussystem mit serieller Datenübertragung vorgestellt, der von den meisten europäischen Herstellern von Elektroinstallationsmaterial unterstützt wird. Da zumindest bei den Studierenden, die bisher die Lehrveranstaltung besuchten, gute Kenntnisse über Rechnernetze und CAD- Systeme aus der Grundlagenausbildung vorhanden waren, konnten in zwei Doppelstunden die Voraussetzungen für praktische Arbeiten an dem an der Professur vorhandenen Experimentierfeld geschaffen werden.

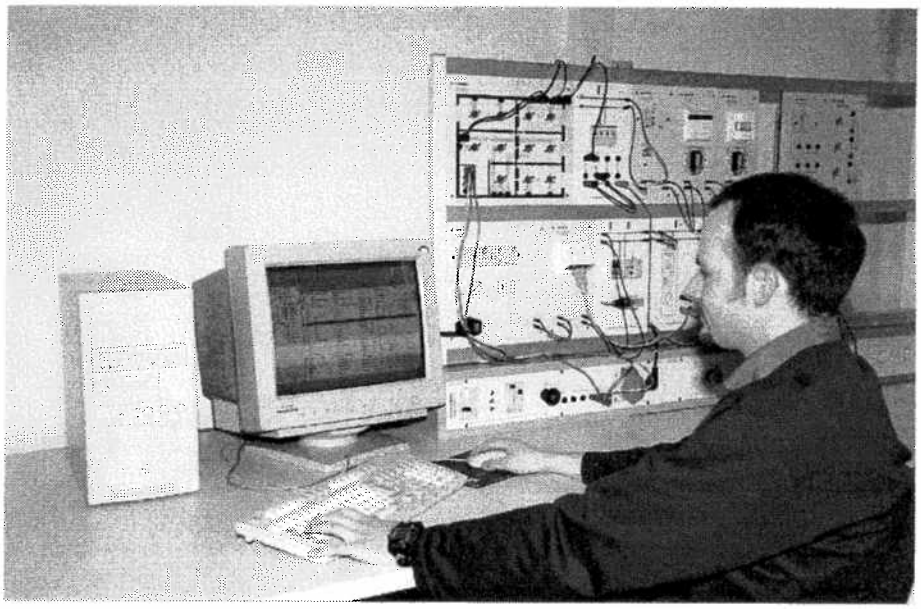


Bild 2: Experimentierfeld EIB

Für den Nachweis der Kompatibilität der Buskomponenten werden im Experimentierfeld Erzeugnisse verschiedener Hersteller eingesetzt. Über eine serielle Schnittstelle RS232 kommuniziert die auf einem PC implementierte EIBA- Software (European Installation Bus Association) mit der nach der jeweiligen Aufgabenstellung zu entwickelnden Buslinie. Folgende Programmkomponenten werden eingesetzt:

- ETS2 Projektierung, Inbetriebnahme/Test, Projektverwaltung
- Tool Design Grafische Projektierung und Dokumentation der EIB- Projekte
- Tool Monitor Diagnose durch Aufzeichnung der Bustelegramme
- Tool Rekonstruktion Projektdaten- Rückgewinnung aus installierten EIB- Anlagen.

Schwerpunkte der studentischen Arbeiten sind neben Projektierung, Inbetriebnahme und Test eines Projektes die Simulation der Änderung des Betriebskonzeptes (räumliche Veränderungen, Umnutzung) durch Umprogrammierung der Beziehungen zwischen den Kommunikationsobjekten (Sensoren, Aktoren) bzw. durch Nachrüstungen. Für die Darstellung einer linienübergreifenden Kommunikation zwischen Sensoren und Aktoren über Linienkoppler mit Filterfunktion sollen im Experimentierfeld die dazu erforderlichen Systemgeräte nachgerüstet werden.

4. Ergebnisse

An der Bauhaus- Universität Weimar hat sich eine Projektgruppe "Studentisches Bauen am Horn" etabliert. Die Studierenden dieser Gruppe haben sich das Ziel gesetzt, fachübergreifend und unter Anwendung neuester Erkenntnisse Wohngebäude speziell für Studenten an einem ausgewählten Standort zu projektieren. Durch Beteiligung eines auf dem Gebiet der Gebäudesystemtechnik ausgebildeten Bauingenieurstudenten wurden im Rahmen einer Diplomarbeit die ersten Voraussetzungen für eine Referenzinstallation mit Bustechnik geschaffen. Durch allgemeine Recherchen zu Nutzerbedürfnissen und zu den zur Verfügung stehenden technischen Komponenten und Konzepten sowie unter Beachtung der Anforderungen aus Betreibersicht (Studentenwerk) wurden Vorschläge zur Gebäudeausrüstung:

- Heizung
- Belüftung
- Wasserversorgung
- Elektroinstallation
- Verschattungsanlagen und
- Alarm- und Sichheitssystem unterbreitet.

Die Umsetzung der gewonnenen Erkenntnisse orientiert sich an der Tatsache, daß in Studentenwohnbauten meist nur allgemeine Wohnstandards erfüllt werden und die Gebäude von Beginn an als Niedrigenergiehäuser konzipiert, den Betreiber wenig zum Einsatz noch teurer Technik motivieren. Für die Gebäudeautomatisierung an diesen Wohnobjekten wurde deshalb eine mehrstufige Lösungen ausgearbeitet, die von absolut notwendigen Funktionen bis hin zu nur mit Bustechnik realisierbarer, komplexer Funktionalität (z.B. Einzelraum- Temperaturregelung in Kopplung mit der Steuerung der zentralen Wärmerzeugung, anwesenheitsbezogene Stelleingriffe etc.) reichen. Dabei wird vorgeschlagen, zumindest die ersten Stufen in die konkrete Planung einzubeziehen, weil durch sie das nahezu gesamte Energiesparpotential in den Wohnungen durch intelligente Regelungen ausgenutzt wird.

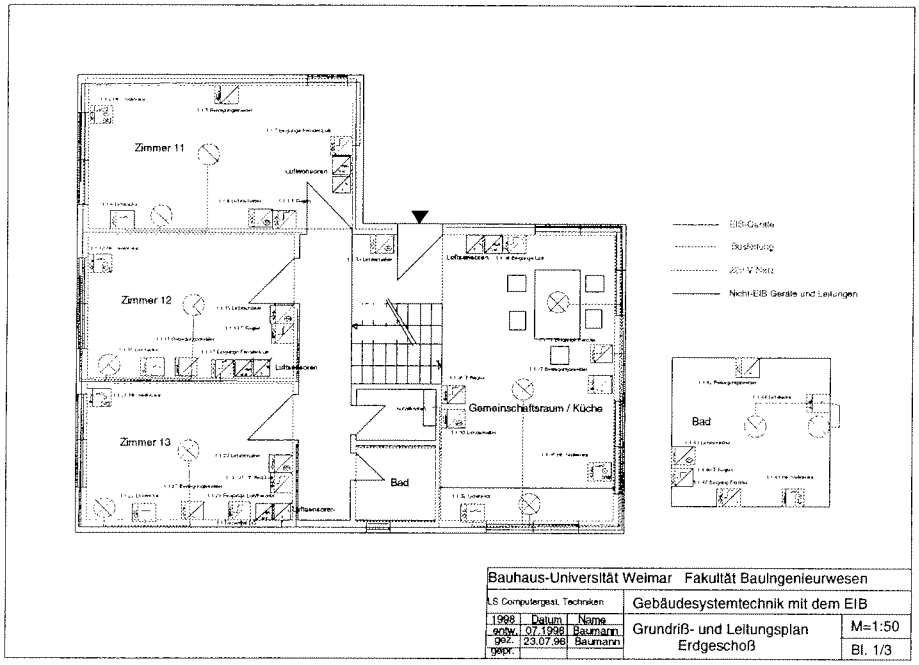


Bild3: Grundriß- und Leitunsplan für das Erdgeschoß eines Studentenwohnbaus

Aufgrund der Planung von mindestens zwei Wohnbauten mit fast identischer Grund- und Nutzfläche, gleicher Grundrißaufteilung, gleicher Bauweise, nahezu gleicher Geländeorientierung und, bei zufälliger Auswahl der Bewohner, mit vergleichbarem Nutzerverhalten wird der Vorschlag unterbreitet, die in der Projektierungsphase hinsichtlich des ökologischen und ökonomischen Nutzens einer Businstallation im Prototyp nur qualitativ und auf Herstellerangaben basierend zu treffenden Aussagen durch Vergleichsmessungen an einem zweiten Gebäude zu quantifizieren. Darüberhinaus hat sich bei der funktionellen Projektierung herausgestellt, das "Speziallösungen" zwar technisch möglich sind, von den Herstellern aber sehr wahrscheinlich applikativ nicht unterstützt werden können. Eigene Entwicklungsleistungen sind hierzu erforderlich.

5. Literatur

M. Baumann: Untersuchungen zum Einsatz busorientierter Systemtechnik in Gebäuden für studentisches Wohnen, Diplomarbeit, Bauhaus- Universität Weimar 1998

REKONSTRUKTION VON ELLIPSOIDMODELLEN FÜR PORTAITMINIATUREN DES 18. JAHRHUNDERTS

N. Popper[1], J. Scheikl[1], F. Breitenecker[1], P. Kammerer[2], E. Zolda[2]

[1]Abt. Simulationstechnik, Technische Universität Wien,
Wiedner Hauptstr. 8-10, A-1040 Wien, Österreich
[2]Patter Recognition & Image Processing Group,
Technisch Universität Wien, Treitlstr. 3, A-1040 Wien
joxg@osiris.tuwien.ac.at

PORTARITMINIATUREN AUS DEM BAROCK

In Ermangelung guter Portraitphotographen benutzten die Habsburger im 17. und 18. Jahrhundert gemalte Portraitminiaturen um in Europa Werbung für ihre Sprößlinge zu machen. Das Resultat dieser Bemühungen ging als Heiratspolitik in die Geschichte ein.

Heute stehen uns etwa 600 dieser Miniaturen für kunsthistorische Untersuchungen zur Verfügung. In Zusammenarbeit mit dem Institut für Kunstgeschichte der Universität Wien beschäftigt sich eine Arbeitsgruppe an der TU Wien mit der computerunterstützten Klassifizierung dieser Miniaturen. Ziel ist es die Miniaturen so zu klassifizieren, daß unbekannte Werke den Künstlern zugeordnet werden können.

Dazu werden für die Köpfe Ellipsoidmodelle angesetzt, und Gesichtspartien auf dem Ellipsoid lokalisiert.

Dieser Beitrag beschreibt nun, wie aus den vermessenen planaren Miniaturen dieses Ellipsoid rekonstruiert und Gesichtspunkte lokalisiert werden.

MALSCHULE IM 18. JAHRHUNDERT

Die Künstler des Barock erlernten ihr Handwerk nach strengen Regeln. Der Schüler erlernte Schemata die erst nach und nach individualisiert wurden. Für die Darstellung von menschlichen Köpfen waren das Ellipsoidmodelle, die die Form des Kopfes und die Lage von Augen, Nase, Mund, usw. anhand von Symmetrien im Gesicht festlegten.

Es wird nun dieses Modell aus dem Portrait rekonstruiert, und ein räumliches Modell des Kopfes der Person generiert mit dem die Kunsthistoriker Versuche durchführen können.

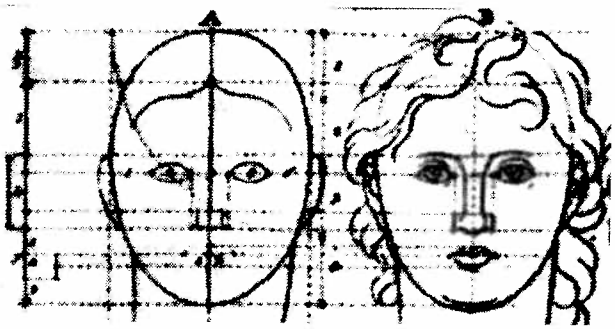


Abbildung 1: Malschema

MODELLANSATZ

Das Modell basiert auf der Annahme, daß ein Kopf, wenn er nach der oben beschriebenen Malschule portraitiert wurde, durch ein Ellipsoid dargestellt werden kann. Das Portrait entsteht durch eine Parallelprojektion in die Bildebene, der Kopf wird durch vier Ellipsen innerhalb dieser Projektion festgelegt. Die Umrißellipse bestimmt die Lage des Kopfes in der Bildebene, die vertikale Ellipse entlang der Gesichtssymmetralen und die horizontalen Ellipsen auf denen Augenbrauen bzw. Mund liegen bestimmen die Neigung des Kopfes.

Die Kunsthistoriker bestimmen einzelne Punkte, durch die die Lage der Ellipsen bestimmt ist. Das sind einerseits Punkte auf der Umrißellipse (in Höhe von Kinn, unteres und oberes Ohrenende sowie Augenbrauen), andererseits für die Symmetrieellipse ein Punkt an der Unterseite der Nase, sowie Punkte in den Mundwinkeln bzw. auf den Brauen für die horizontalen Ellipsen.

Diese Punkte wurden bis jetzt händisch auf den gescannten Bildern vermessen und stehen für die Rekonstruktion des Modells zur Verfügung.

Zu ermitteln sind Größe der erwähnten Ellipsen, sowie deren Lage im Raum.

REKONSTRUKTION

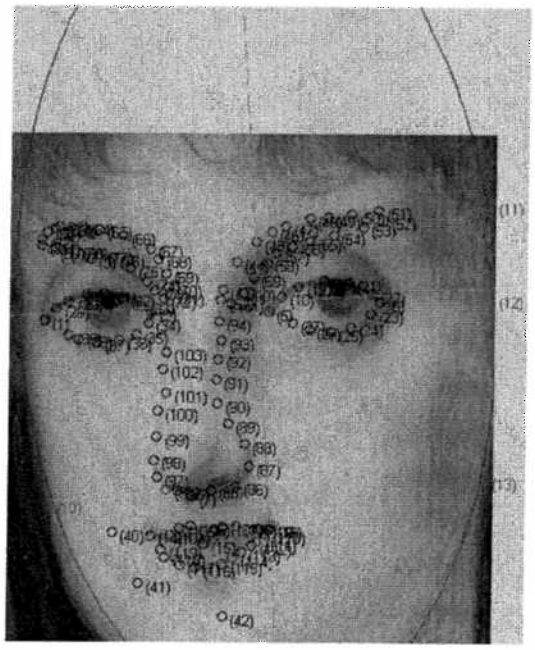


Abbildung 2: Umrißellipse

Zur Rekonstruktion des vom Künstler verwendeten Kopfmodells wird zuerst die Umrißellipse mit *Datafitting* Verfahren ermittelt. Es wird die Ellipse berechnet, welche die vermessenen Konturpunkte – im Sinne der kleinsten Fehlerquadrate - am besten annähert. Zu diesem Zweck wurden zwei verschiedene Verfahren getestet. Einerseits ein iteratives Optimierungsverfahren, andererseits ein direktes Verfahren, das auf ein erweitertes Eigenwertsystem zurückzuführen ist.

Das iterative Verfahren wurde gewählt, da auf Grund der Modellannahmen ein ausgezeichneter Startwert ermittelt werden kann. Der Hauptscheitelpunkt der Startellipse liegt am Kinn, die Nebenscheitel an den Ohren. Ausgehend von diesem Startwert wird mittels der MATLAB Optimisation Toolbox mit einer geeigneten Skalierung die Umrißellipse ermittelt.

Das direkte Verfahren arbeitet mit einer analytischen LSQ Methode. Es wird ein erweitertes Eigenwertsytem gelöst, bei dem die Parameter des gesuchten Kegelschnittes in den Eigenvektoren stehen[1].

$$D^T Da = Sa = \lambda Ca$$

D	$[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_n]^T$, mit $\boldsymbol{x} = [x^2\ xy\ y^2\ x\ y\ 1]$
C	Randbedingung
a	Ellipsenparameter
λ	Eigenwert

Mittels der Randbedingung C wird sichergestellt, daß der berechnete Kegelschnitt eine Ellipse ist. Der zum negativen Eigenwert gehörige Eigenvektor stellt die Parameter der Lösungsellipse dar. Die Methode ist invariant bezüglich Translation und Rotation.

3-D MODELL DES PORTRAITS

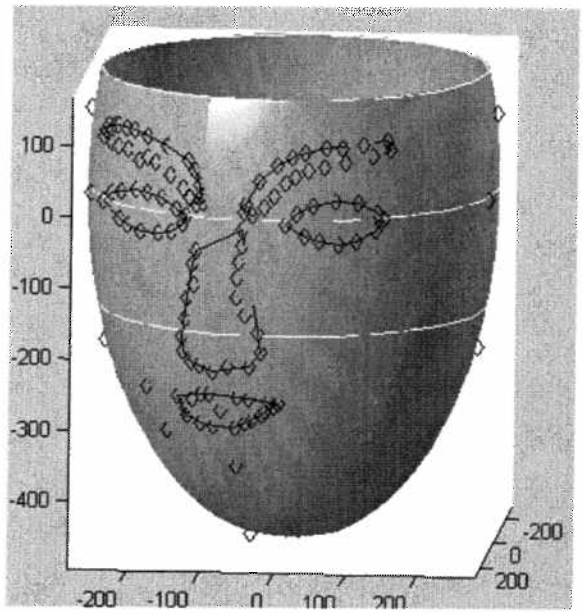


Abbildung 3: Ellipsoidmodell mit Datenpunkten

Anschließend wird mit Hilfe der Gesichtspunkte (Nase, Mund, Brauen, etc.) der Betrachtungswinkel und das dreidimensionale Modell berechnet. Dabei geht man von einer Orthogonalprojektion aus. Aus den erwähnten Symmetrien werden 2 der 3 Winkel berechnet. Der dritte Winkel, der das Nicken des Kopfes beschreibt, kann aufgrund numerischer Schwierigkeiten nicht sinnvoll ermittelt werden. In Abstimmung mit den Kunsthistorikern wurde aber die Einschränkung getroffen, daß dieser Winkel in Zusammenhang mit der Bestimmung die die Bilder hatten zu vernachlässigen ist.

Mit Hilfe der Umrißellipse wird nun das Rotationsellipsoid berechnet, das das Kopfmodell der jeweiligen Malschule repräsentiert.

Alle vorhandenen Datenpunkte in dem Portrait werden nun auf das 3-dimensionale Ellipsoidmodell projiziert. Die Relationen verschiedener Gesichtsbereiche können so von den Kunsthistorikern verglichen werden.

IMPLEMENTIERUNG

Der Programmablauf gestattet den Kunsthistorikern die Konturellipse zu bewerten und gegebenenfalls zu editieren, d.h. Punkte zu verschieben und hinzuzufügen. Weiters können die Punkte zu verschiedenen Gruppen zusammengefasst werden (Auge, Nase, etc.., aber auch bestimmte, für die weitere Betrachtung entscheidende Wangenpartien). Diese definierbaren Kurven werden auch auf dem Ellipsoid ausgegeben (siehe Abbildung 3). Bei allen Änderungen an den Datenpunkten muß darauf geachtet werden, daß das Ergebnis nicht entsprechend den Erwartungswerten der Benützer verändert wird, d.h. daß z.B. nicht die Konturpunkte so lange angepasst werden bis die Umrißellipse „stimmt". In diesem Fall würde der Modellansatz seine Sinnhaftigkeit natürlich verlieren.

Die vermessenen Datenpunkte werden auf das 3 dimensionale Modell projiziert. Der Anwender kann nun in jeder Position des Kopfes Punkte hinzufügen und verschieben. Zusatzinformationen für den Kunsthistoriker sind die Blickrichtung sowie die Beleuchtung des Portraitierten d.h. man kann ermitteln wo die Lichtquellen plaziert waren. In weiterer Folge ist geplant die Datenpunkte mittels Strichdetektoren zu bestimmen, um einerseits dem Anwender die mühsame Arbeit zu ersparen die Punkte einzeln einzutragen, andererseits um die Genauigkeit der Datenpunkte zu erhöhen.

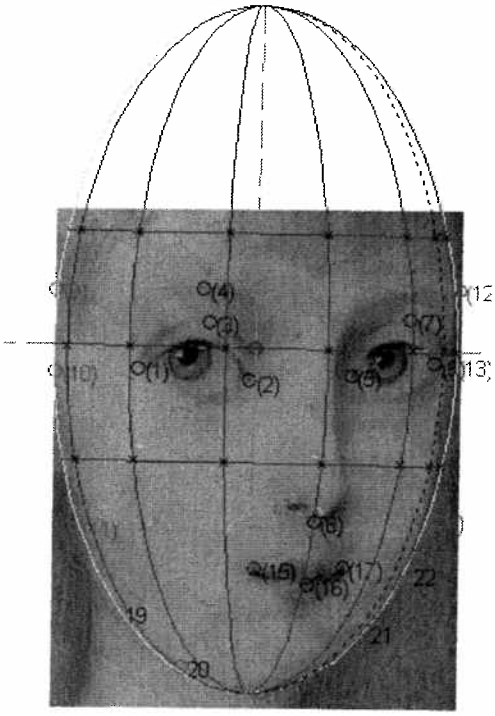


Abbildung 4: Längen- und Breitenkreise

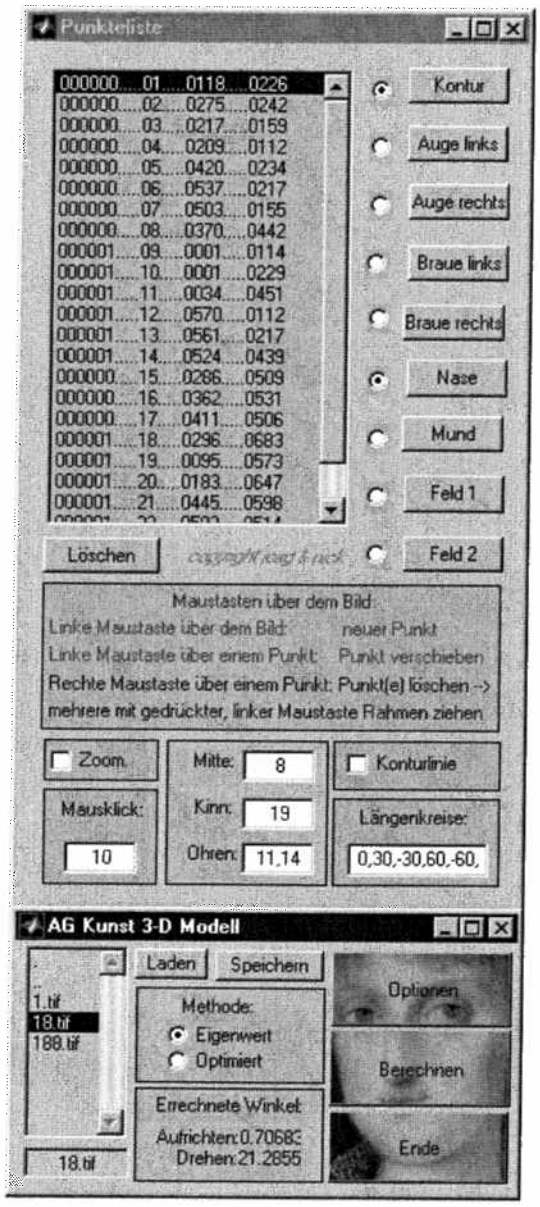


Abbildung 5: Benutzeroberfläche

Für die weitere Analyse der Kunsthistoriker werden weiters vom Anwender frei definierbare Längen- und Breitenkreise vom 3-dimensionalen Modell auf das Originalportrait zurückprojiziert. Auf diese Art können für verschiedenste Portraits standardisierte Bereiche definiert werden in denen nun von den Kunsthistorikern Strichanalysen durchgeführt werden können (siehe Abbildung 4). Die Schnittpunkte der Längen- und Breitenkreise werden, so wie alle anderen Daten, auch als Textdatei ausgegeben, und können für weitere Berechnungen herangezogen werden.

ZUSAMMENFASSUNG

Da das Modell Teil eines größeren Projektes ist stehen derzeit noch keine genaueren Ergebnisse zur Verfügung. Bisher hat sich jedoch gezeigt, daß mit diesem Modellansatz sehr gute Ergebnisse zu erzielen sind, d.h. trotz des relativ einfachen Ansatzes eines Ellipsoides werden die Gesichtsproportionen gut wiedergegeben. (Man beachte, daß es sich um eine *Rekonstruktion* handelt, bei der davon ausgegangen wird, daß der Maler ja eben diese Annahmen getroffen hat). Die Umrißellipse, von der das 3-dimensionale Modell entscheidend abhängt, hat sich des weiteren als sehr stabil bezüglich Punktverschiebungen erwiesen.

Genauere Ergebnisse zum Einsatz eines solcher Modelles bei der Arbeit der Kunsthistoriker sind sicher erst nach einer längeren, praktischen Anwendung durch die Experten zu erzielen, die Möglichkeiten die sich den Kunsthistorikern mit diesem Modell aber bieten sind sicher vielversprechend

LITERATUR

[1] M. Pilu; A.W. Fitzgibbon; R.B. Fisher;"Ellipse-Specific Direct Least-Square Fitting; ICPR; 1996; Vol. A; S. 253-257

CADMOS
Komponentenbasierte Modelle zur Systembeobachtung

J.Reinhardt
Institut für Informatik
Johannes Gutenberg-Universität Mainz
Staudingerweg, D-55128 Mainz
reinhardt@informatik.uni-mainz.de

Zusammenfassung
CBD - Component Based Software Development hat sich als Entwicklungsparadigma im Bereich des Software Engineering etabliert. Dabei ist die Kommunikation zwischen den einzelnen Softwarekomponenten sowie deren Organisation Gegenstand aktueller Untersuchungen.
Im Rahmen des Projektes CADMOS werden, basierend auf der Erstellung einer Entwicklungsumgebung zur modellbasierten Systembeobachtung, Ansätze zur Kommunikation und Organisation von Software-Komponenten entwickelt. Schwerpunkt ist dabei die Frage, wie bestehende oder neue Softwarekomponenten dynamisch in das Modell eines zu untersuchenden Systems integriert werden können. Gegenstand des folgenden Textes ist der in diesem Zusammenhang entwickelte Ansatz der Component Driven Graphs (CDG). Dabei wird auch die Frage diskutiert, inwieweit der CDG-Ansatz auch zur Entwicklung komponentenorientierter Software eingesetzt werden kann.

1 CADMOS: Systeme zur modellgestützten Systembeobachtung

1.1 Motivation und Umfeld

Die Arbeitsgruppe »Angewandte Informatik« des Instituts für Informatik der Johannes Gutenberg-Universität Mainz beschäftigt sich unter der Leitung von Prof. Dr. Perl seit Anfang der achtziger Jahre mit der Möglichkeit, Verfahren und Techniken aus dem Bereich der Informatik im Umfeld des Sports und der Sportwissenschaften einzusetzen. Aus diesen interdisziplinären Aktivitäten gingen unter anderem Anwendungen und Konzepte hervor, die sich mit der Beobachtung und Modellierung komplexer Systeme beschäftigen, wie sie beispielhaft durch Sportspiele gegeben sind [MLJP95].
Das Softwaresystem CADMOS wurde ursprünglich zur Vereinfachung und Automatisierung von Systembeobachtungen im Umfeld von Sportspielen konzipiert. Dabei sollte das zu beobachtende Sportspiel zunächst durch ein Modell beschrieben werden. Aufbauend auf dem erstellten Modell sollten Beobachtungen von realen Sportspielen durchgeführt werden.
Im Rahmen der Arbeiten der Arbeitsgruppe wurden schon eine Reihe Softwaresysteme zur Systembeobachtung entwickelt [GB94][JP97B], wobei jedoch jeweils ein dediziertes Softwaresystem für jedes zu beobachtende System entwickelt wurde. Im Arbeitskontext des Sports wurden die Softwaresysteme also jeweils spezifisch für eine Sportart konzipiert.
Der Leitgedanke von CADMOS bestand nun darin, ein Softwaresystem zu konzipieren, welches sportartunspezifisch eingesetzt werden kann. Da die Systembeobachtungen jedoch auf einem Modell basieren und ein sinnvoll einsetzbares, gemeinsames Modell für alle in Frage kommenden Sportspiele nicht existiert, wurde ein Softwaresystem konzipiert und realisiert, mit dem zunächst das Modell einer Sportart entwickelt und anschließend auf Grundlage dieses Modells das System beobachtet werden kann.
Als Modelle wurden zunächst Zustand-Ereignis-Graphen (ZE-Graphen) gewählt [UKHS83], da mit diesen Graphen in früheren Untersuchungen schon gute Ergebnisse erzielt wurden [JP97B]. Jedem Zustand oder Ereignis wurden Attribute zugeordnet, mit denen der Zustand oder das Ereignis näher beschrieben wurde (siehe unten).
Eine Systembeobachtung bestand in dieser Realisierung aus der Erfassung einer Reihe von Knoten des ZE-Graphen (Pfad). Die einzelnen Knoten des Pfades repräsentieren dabei die Zustände, in denen sich das beobachtete System im Laufe des Beobachtungszeitraums befand bzw. die Ereignisse

welche eine Zustandsänderung bewirkten. Die Erfassung des Pfades wurde um die Erfassung der Werte der den Knoten zugeordneten Attributen ergänzt. Die Attribute konnte dabei numerische oder alphanumerische Werte annehmen und repräsentierten Größen, die an dem beobachteten System durch einen Anwender mess- oder beobachtbar waren.

1.2 Erste Konzepte und Realisierungen

Den in 1.1 ausgeführten Grundgedanken folgend wurden zunächst drei Kernbereiche konzipiert und realisiert:

- Modellentwurf: Konzipierung und Realisierung eines Werkzeuges (ZE-Designer) zur komfortablen und grafischen Modellierung von ZE-Graphen [FS97].

- Interaktionen: Konzipierung und Entwicklung eines Werkzeuges (Interaktions-Assistent) zur Generierung von auf den Attributen des ZE-Graphen basierenden grafischen Benutzeroberflächen zur Systembeobachtung [UW97].

- Persistenz: Konzipierung und Entwicklung eines Werkzeuges, welches die Daten einer Systembeobachtung vorhält und zur weiteren Bearbeitung bereitstellt [AB97].

Mit dem ZE-Designer wird ein ZE-Graph für das zu beobachtende System entwickelt. Mit diesem Werkzeug werden auch die Attribute für jeden Knoten (Zustand/Ereignis) festgelegt. Zu jedem Knoten des ZE-Graphen wird anschließend mit dem Interaktions-Assistenten eine Eingabemaske entworfen, auf der die Eingabefelder für die im ZE-Designer angelegten Attribute plaziert werden können.
Unter der Steuerung des Interaktions-Assistenten wird anschließend auch die Systembeobachtung durchgeführt. Dabei bestimmt der Anwender (basierend auf seinen Beobachtungen) durch aufeinanderfolgendes Auswählen der jeweiligen Knoten des ZE-Graphen einen Pfad über den ZE-Graph (vgl. 1.1). Der Interaktions-Assistent fordert die Werte der dem ausgewählten Knoten zugeordneten Attribute an. Dabei verwendet der Interaktions-Assistent die zuvor entworfene Benutzerschnittstelle. Die Arbeit des Persistenzsystems ist für den Anwender transparent. Das Persistenzsystem legt die erfaßten Daten in einer Datenbank ab, wo sie für weitere Auswertungen zur Verfügung stehen.

Aufbauend auf den aus diesen Untersuchungen gewonnenen Erkenntnissen wurde ein allgemeinerer Ansatz erarbeitet, der sich in allen Ebenen des zugrundeliegenden Konzeptes niederschlug.
Auf der methodologischen Ebene zeigt sich die Modellierung von Sportspielen durch Zustand-Ereignis-Graphen als möglich aber schwierig, wenn man sich auf konventionelle ZE-Modelle beschränkt.
Insbesondere der Bereich der Attributierung, also der Zuordnung von Attributen zu den Knoten des ZE-Graphen, rückte in das Zentrum des Interesses. So konkretisiert sich erst bei der Modellierung der Typ der verwendeten Attribute. Numerische oder alphanumerische Typen genügen hier nicht, um alle oder auch nur einen Großteil der beobachtbaren Eigenschaften eines System abzubilden.
Zur Bereitstellung vieler beobachtbarer Attribute eines Systems existieren oft fertige Softwaresysteme, die *technisch* in der Lage sind, verschiedenartigste Datentypen (die für eine Systembeobachtung relevant sein können) aufzunehmen und zu verarbeiten. Beispielhaft sei hier an Video- oder allgemein bildbearbeitende Software gedacht.

2 Der Ansatz der Component Driven Graphs - CDG

2.1 Grundlegende Konzepte

Die Ergebnisse aus 1.2 und aktuelle Entwicklungen im Bereich des Component Based Development motivieren zu einem Ansatz, in dem der Einsatz von Softwarekomponenten eine zentrale Rolle spielt. Die Vorteile von Softwarekomponenten sind schon umfangreich beschrieben worden, wobei auch deren Einsatz im Rahmen von Simulationen Gegenstand von Untersuchungen waren [DAMH98]. Die Herausforderungen des Component Based Development liegen dabei aktuell weniger in der technischen Realisierung sondern in der Organisation und Kommunikation zwischen den Komponenten.
Im folgenden Ansatz wird aufzeigt, wie die Möglichkeiten des Component Based Development bei der modellgestützte Systembeobachtung genutzt werden können. Anschließend wird dargestellt, wie das geschilderte Verfahren zur Erstellung weiterer Softwaresysteme eingesetzt werden kann.
Grundlegend für das folgende Vorgehen ist, daß die den Knoten zugeordneten Attribute des Modells nur in einer abstrakten Form beschrieben werden. Die eigentliche Repräsentation und Verarbeitung der Werte dieser Attribute findet durch externe Softwarekomponenten statt. Zur Beobachtungszeit werden für Aufgaben wie Erfassung und Repräsentation der Werte die externen Softwarekomponenten eingesetzt.
Der Ansatz kann erweitert werden: Nicht nur die Attribute sondern auch die Knoten (im Falle des ZE-Graphen also Zustände und Ereignisse) können durch externe Softwarekomponenten repräsentiert werden.
Aus dem in 1.1 beschriebenen ZE-Graphen ist damit ein Rahmenwerk geworden, welches zur Koordinierung der Interaktionen der einzelnen Softwarekomponenten dient (vgl. Abb.1).
Zur vollständigen Beschreibung des Modells gehören daher die Softwarekomponenten und der die Interaktionen beschreibende Graph. Diese Art der Modellierung wird im folgenden als Component Driven Graphs – CDG, der zugrunde liegende Graph als CDG-Graph bezeichnet. Attribute heißen Komponenten-Attribute, wenn sie durch eine Softwarekomponente repräsentiert werden. Entsprechend werden Knoten die durch Softwarekomponenten repräsentiert werden als Komponenten-Knoten bezeichnet. Unter der Funktionalität eines Komponenten-Knoten oder eines Komponenten-Attributs wird die Funktionalität der zugeordneten Softwarekomponente verstanden.
Der CDG-Controller ist dabei das Softwaresystem, welches auf der Grundlage des CDG-Graphen die Kommunikation zwischen den Komponenten steuert. Mit dem CDG-Designer wird der CDG-Graph erstellt.
Bei einer Systembeobachtung (vgl. 1.1) werden die einzelnen, den Komponenten-Attributen zugeordneten Softwarekomponenten aufgefordert, einen Wert für das Attribut zu bestimmen. Da der CDG-Controller den Typ eines Wertes, der von einem Komponenten-Attribut erfaßt wird, nicht kennt (vgl.2.4), steht es in der Verantwortung der dem Komponenten-Attribut zugeordneten Softwarekomponente, diesen Wert aufzunehmen, vorzuhalten, darzustellen oder zu modifizieren (vgl. Abb.2).
Der zur Steuerung verwendete CDG-Controller identifiziert die einzelnen Werte eines Komponenten-Attributs dabei durch einen eindeutigen Schlüssel.
Im Laufe einer Systembeobachtung aktiviert (vgl. 2.3) der CDG-Controller über geeignete Mechanismen und basierend auf dem CDG-Graph die einem Komponenten-Attribut zugeordnete Softwarekomponente. Diese Softwarekomponente erfaßt einen Wert, hält diesen Wert vor und gibt dem CDG-Controller einen Schlüssel zurück, mit dem der Wert eindeutig identifiziert werden kann.
Wird eine Verarbeitung (Eingabe, Ausgabe, Modifikation) des erfaßten Wertes nötig, so wird immer der Schlüssel zur Identifizierung des Wertes verwendet.
Bei den ersten Realisierungen (vgl. 1.2) wurde das System beobachtet, indem ein menschlicher Beobachter dem Softwaresystem über geeignete Interaktionen mitteilte, in welchem Zustand sich das System gerade befindet respektive welches Ereignis diesem Zustand vorausging oder folgt. Anschließend erfaßte der Beobachter manuell die Werte der den Zuständen oder Ereignissen zugeordneten Attributen.
Beim Einsatz von Komponenten-Attributen kann die Erfassung dieser Attribute automatisiert erfolgen. Werden geeignete Komponenten-Knoten oder Komponenten-Attribute verwendet, so kann dies zu einer vollständig automatisierten Beobachtung führen.

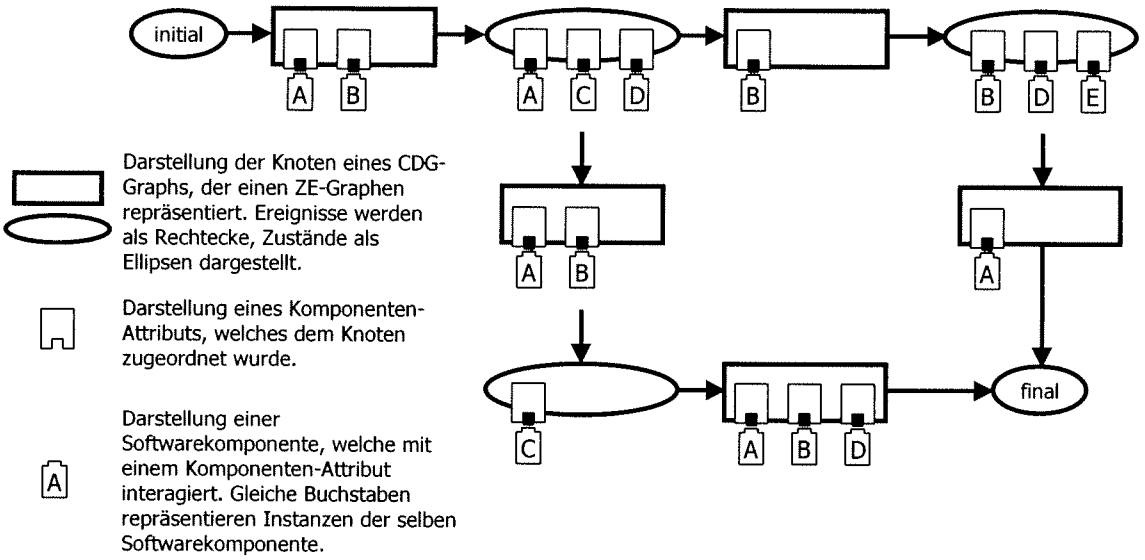


Abb.1: Beispiel eines einfachen CDG-Graphen. Der dargestellte Graph basiert auf einem ZE-Graphen, wobei die einzelnen Knoten durch Komponenten-Attribute erweitert wurde (vgl. Abb.2). Die Zustände mit der Bezeichnung »initial« und »final« bezeichnen den initialen oder finalen Zustand einer Beobachtung. (Im Rahmen des CDG Ansatzes muß der CDG-Graph nicht zwingend einen ZE-Graphen repräsentieren.)

2.2 Der CDG-Ansatz zur Organisation von Komponenten

Da über die Funktionalität der Komponenten-Attribute und Komponenten-Knoten keine einschränkenden Annahmen gemacht wurden, kann das aus der Zuordnung von Softwarekomponenten zu Knoten und Attributen des CDG-Graphen entstehende Software-System allgemeiner beschrieben werden: Die Interaktion von Softwarekomponenten wird durch ein zentrales Softwaresystem basierend auf einem zugrundeliegenden Graphen gesteuert.
Bei der Verwendung von Komponenten-Knoten und deren automatisierten Aktivierung über den CDG-Controller können komponentenbasierte Softwaresysteme beschrieben werden, deren Organisation durch den CDG-Graphen festgelegt ist. Die Aufgaben des CDG-Designers lassen sich damit mit den Aufgaben von Software-Entwicklungsumgebungen vergleichen. Dies gilt hierbei für Aspekte der Erstellung des CDG-Designers selbst, als auch für die Erstellung eines CDG-Graphen mit Hilfe des CDG-Designers.
Der CDG-Graph legt dabei geringstenfalls fest, in welcher Reihenfolge die Softwarekomponenten aktiviert werden dürfen. Der CDG-Controller (vgl. 2.1) löst dabei Aufgaben, die in das Umfeld der Object Broker (CORBA, COM) einzuordnen sind. Dies wird deutlich, wenn man bedenkt, daß bei geeigneter technischer Realisierung die vom CDG-Controller aktivierten Komponenten nicht auf einem Rechner lokalisiert sein müssen. Im Abschnitt 2.3 wird jedoch ausgeführt, daß der CDG-Controller auf Verfahren wie CORBA und DCOM aufbaut, um die Kommunikation zwischen den Komponenten technisch zu realisieren.
Gegenstand aktueller Untersuchungen ist, inwieweit der Ansatz der CDGs in seiner hier vorgestellten Form im Rahmen des Component Based Developments zur Montage von Softwaresystemen eingesetzt werden kann. Auf der anderen Seite liefern gerade Entwurfsverfahren des Component Based Development, wie das von der UML abgeleitete Catalysis [DDAC99] wertvolle Ansätze bei der Entwicklung des CDG-Designers.

2.3 Realisierung

Da der CDG-Controller die Realisierung der Typen der verwendeten Werte nicht kennt, werden in CADMOS die Typen einzelner Komponenten-Attribute in Klassen eingeordnet. Über diese Klassen bestimmen sich die Methoden und Eigenschaften, welche von entsprechenden Softwarekomponenten unterstützt werden müssen. Die zur Zeit existierenden Klassen orientieren sich an den Anforderungen der Systembeobachtung. So existiert eine Klasse, in der alle Typen zusammengefaßt werden, die bei der Beschreibung eines Zeitpunktes verwendet werden. Als Beispiel sei hier der Punktestand in einem Sportspiel zu einem festgelegten Zeitpunkt genannt. In einer weiteren Klasse sind alle Typen zusammengefaßt, die bei der Beschreibung einer Systemausprägung mit einer zeitlichen Ausdehnung Verwendung finden. So ist eine Videosequenz eine Systemausprägung mit einer zeitlichen Ausdehnung. Zu jeder dieser Klassen existiert ein fester Satz von Methoden und Eigenschaften, über die der CDG-Controller mit den Softwarekomponenten interagiert (vgl. Abb.2).
Im Gegensatz zu implementierungsorientierten Beschreibungen wie bei gängigen technischen Realisierungen von Komponenten- und Objektschnittstellen wie CORBA oder COM, werden beim CDG-Ansatz zur Beschreibung der Typen von (Attribut-)Werten *nicht* konkrete Typen (wie `DOUBLE`), sondern die beschriebenen Klassen verwendet. Dabei wird eine semantische Einordnung der Typen in die Klassen vorgenommen, wobei niemals direkt auf einen Wert des Typs zugegriffen wird, sondern nur über geeignete Methoden und Eigenschaften der Softwarekomponente. Da die Interaktion technisch auf Verfahren wie CORBA und COM basiert, werden bei der Aktivierung der Methoden der Softwarekomponenten natürlich bekannte Typen verwendet (siehe unten).

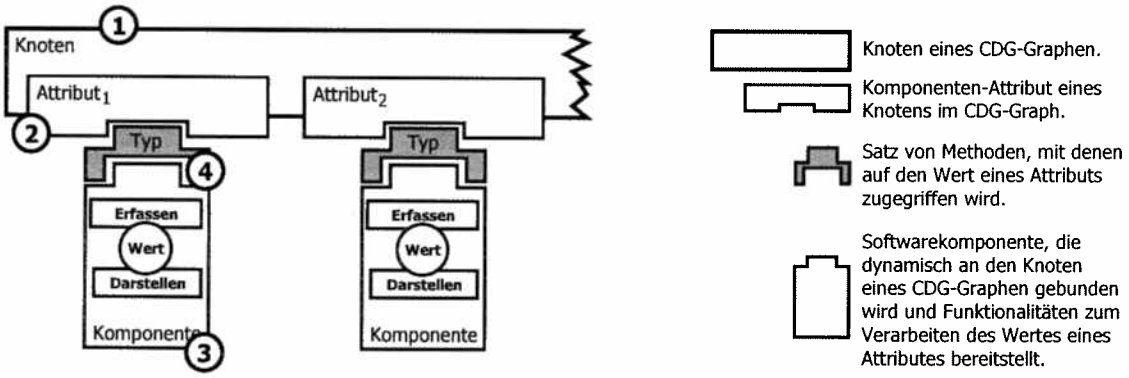


Abb.2: Einem Knoten ① können Komponenten-Attribute ② zugeordnet werden. Dabei stellt eine Softwarekomponente ③ alle Funktionen zur Verfügung, die zur Bearbeitung des Wertes des Attributs benötigt werden. Abhängig davon, welcher Typ-Klasse der Typ des Wertes zugeordnet wurde, interagiert die Softwarekomponente mit dem Komponenten-Attribut über einen Satz von Methoden ④.

Zum Entwicklungszeitpunkt des *Werkzeugs* steht nicht fest, mit welchen Komponenten zur Beobachtungszeit gearbeitet werden soll. Dies bedeutet, daß möglichst keine Verfahren zum Einsatz kommen sollen, bei denen die einzelnen Komponenten zur Erstellungszeit des Werkzeuges bekannt sein oder vorliegen müssen. In CADMOS werden bei der Realisierung der Kommunikation zwischen dem CDG-Controller und den externen Komponenten verschiedene *Konzepte* verfolgt, die sich jeweils in verschiedenen technischen Konkretisierungen niederschlagen:

- *Direkte Unterstützung:* Die anzubindende Softwarekomponente bietet schon die vom CDG-Controller benötigten Methoden oder Eigenschaften an und kann über Standardschnittstellen (DLL, COM/DCOM oder CORBA) angebunden werden.

- *Mapping:* Die Abbildung von vorhandenen Schnittstellen einer Softwarekomponente auf die vom CDG-Controller geforderten Schnittstellen.

- *Adapter:* Bei einer Softwarekomponente, deren Einbindung mit keiner der genannten Verfahren realisierbar ist, werden Adapter erstellt, welche die Anforderungen von CDG-Controller und externer Softwarekomponente abgleichen. Der Adapter kann dabei Funktionalitäten ergänzen, welche der CDG-Controller benötigt, die Komponente aber nicht bietet.

Die aktuelle Realisierung des CADMOS Projektes umfaßt den CDG-Designer sowie den CDG-Controller als Anwendungen für die Windows NT Systemumgebung. In gegenwärtigen CDG-Graphen werden COM/CORBA-Objekte mit direkter Unterstützung (siehe oben) eingesetzt, wobei die Erstellung beispielhafter Mappings und Adapter Gegenstand aktueller Arbeiten ist.

2.4 Anwendungen

Neben der in 1.1 ausführlicher diskutierten Sportspielbeobachtung werden in weiteren Bereichen Untersuchungen zum Einsatz von CADMOS vorgenommen:

- *Beobachtung menschlichen Verhaltens in der Psychologie*
  In einem Anwendungsfall aus der Psychologie, in dem Verhaltensmuster von Personen modelliert werden, sollen neben einfach zu erfassenden Attributtypen auch Videosequenzen in die Untersuchungen integriert werden. Dazu wird bei der Modellierung des Verhaltens ein geeignetes Komponenten-Attribut aufgenommen. Die entsprechende Softwarekomponente steuert eine externe Videoquelle an. Um für die Videobearbeitung typische Aktionen zu ermöglichen, blendet die Softwarekomponente bei Bedarf ein während der Beobachtung erfaßtes Video ein und erlaubt geeignete Manipulationen.

- *Software Profiling und Software Quality Assurance*
  Zur Qualitätssicherung soll ein Softwaresystem untersucht werden. Dazu wird ein Modell erstellt, welches das Verhalten eines Anwenders innerhalb der Anwendung beschreibt. Modelliert werden die Übergänge von einem Anwendungsbereich der beobachteten Software in einen anderen. Die einzelnen Anwendungsbereiche werden dabei von Knoten des CDG-Graphs repräsentiert. Komponenten-Attribute sollen das Terminal der Anwender überwachen und feststellen, wann der Anwendungsbereich verlassen wird. Dabei sollen in entsprechenden Komponenten-Attributen ausgewählte Informationen des aktuellen Terminals vorgehalten werden.

In beiden Anwendungsfällen besitzt der CDG-Controller keine Funktionalitäten, welche die Datenerfassung *technisch* vornehmen könnten. Die kompletten problemabhängigen Anforderungen werden auf Funktionalitäten von Komponenten-Attributen oder Komponenten-Knoten abgebildet.

3 Literatur

[AB97] A. Brod: Entwicklung eines objektorientierten Persistenzsystems für Delphi. Mainz: Diplomarbeit, Johannes Gutenberg-Universität, 1997

[DA98] D. Adamski, M. Hiller: Erstellung einer Simulationsumgebung aus Komponenten. In: Zürich: Simulationstechnik - Tagungsband zum 12. ASIM-Symposium in Zürich, Hochschulverlag AG an der ETH Zürich, 1998

[DDAC99] D'Souza, Wills: Objects, Components, and Frameworks with UML, Addison Wesley, 1999

[FS97] F. Spickermann: CADMoSS – Integrierte Modell- und Datenbankgenerierung für Sportspiele; Mainz: Diplomarbeit, Johannes Gutenberg-Universität, 1997

[GB94] G. Boguschewski, Th. Meiberth, J. Perl: Das Tischtennis-Simulations-System TiSSy; In: Tischtennis Lehre, 8(1), 5-8, 1994

[JP97B] J. Perl: Möglichkeiten und Probleme der computerunterstützten Interaktionsanalyse; In: J. Perl (Hrsg.): Informatik im Sport Vol.5; Köln: Sport und Buch Strauß, 1997

[MLJP95] M. Lames, J. Perl: Sportinformatik - Gegenstandsbereich und Perspektiven einer sportwissenschaftlichen Teildisziplin. In: Leistungssport, 25(3), 1995

[UKHS83] U. Knitelius, H.-J. Schröder: Strukturelle Analyse und Implementation interpretierender Petri-Netz-Vergröberungen; Diplomarbeit Osnabrück: Fachbereich Mathematik, 1983

[UW97] U. Wagner: CADMoSS – Interaktive Generierung von Benutzungsoberflächen zum Erfassen und Analysieren von Sportspielen. Mainz: Diplomarbeit, Johannes Gutenberg-Universität, 1997

Flexibler Aufbau von fachgebietsspezifischen Simulationssystemen mit dem komponentenbasierten Simulationssystem SimCo-DB

Thomas Wiedemann, TU Berlin

1. Einleitung

Fachgebietsspezifische, bausteinbasierte Simulationssysteme haben sich in der Vergangenheit bei einer Vielzahl von Simulationsuntersuchungen bewährt. Im Vergleich zu traditionellen Simulationssprachen wie GPSS oder SIMULA können mit Bausteinsystemen wie TAYLOR II oder ARENA in sehr kurzer Zeit sehr komplexe Modelle mit anschaulicher Animation und Ergebnisdarstellung aufgebaut werden.

Vergleicht man die unterschiedlichen Modellierungsansätze fachgebietsspezifischer Simulationssysteme, so existiert auf den ersten Blick eine sehr unterschiedliche Anzahl von Basiselementen mit sehr heterogenen Bausteinparametern. Leider ist die Austauschfähigkeit und Portabilität von Simulationsmodellen gerade durch diese Vielfalt an Bausteinkonzepten geringer als bei älteren Simulationssprachen. Da die Funktionsweise der einzelnen Bausteine auch nur allgemein bekannt ist und keine Details der Implementierung zugänglich sind, muß der Anwender auf die Korrektheit der internen Algorithmen vertrauen. Dies wirft besonders bei der Verifizierung und Validierung schwerwiegende Probleme auf, da wesentliche Bestandteile des gesamten Computermodells nicht zugänglich und meist nicht ausreichend dokumentiert sind.

Mit dem vorliegenden Artikel soll der Versuch unternommen werden, unter Beibehaltung aller Vorteile von bausteinorientierten Simulationsmodellen ein universelles Framework für die Ablage, die Handhabung und die funktionale Beschreibung derartiger Modelle zu definieren.
Wesentliche Anforderungen an dieses System, auch in Anlehnung an [Rabe99], sind:

- eine maximale Offenheit aller benutzten Basistechnologien und Standards,
- sehr flexible und austauschfähige Bedienoberflächen und Visualisierungsformen,
- allgemeingültige und verifizierbare Funktionsbeschreibungen der Modellbausteine.

Entsprechend dieser Aufzählung werden nachfolgend Lösungsansätze vorgestellt und diskutiert.

2. Basistechnologien zur Ablage und Verarbeitung der Modelldaten

In der Vergangenheit wurden Modelle diskreter, bausteinorientierter Simulationssysteme in der Regel in proprietären Binärformaten gespeichert. Angesichts der Komplexität der Modellbausteine und der historischen Entwicklung dieser Systeme erscheint dies auch logisch. Aus der Perspektive aktueller Softwaretechnologien bereiten diese speziellen Datenformate jedoch große Probleme bei der Erweiterung und insbesondere beim Austausch oder bei externen Manipulaton von Modellen. Im Bereich der allgemeinen Datenverarbeitung wurde das gleiche Problem durch den Einsatz von Datenbanken mit genormten Schnittstellen wie SQL gelöst. Ein Einsatz von Datenbanken erscheint daher auch Simulationsbereich zukünftig dringend angeraten.

Die Datenbankbeschreibung realer Systemen stellt bekanntlich selbst einige Anforderungen. Ein üblicher Ansatz ist die Definition von Entities für alle zu beschreibenden Systemobjekte und deren Relationen zueinander. Analog wie die Systemmodellierung im Simulationsbereich erfordert diese Systembeschreibung jedoch erhebliche Erfahrungen des Anwenders. Da bei einer anwendungsspezifischen Datenbankdefinition zudem auch die eigentliche Simulationsdurchführung jeweils angepaßt werden müßte, kommt diese Vorgehensweise nicht in Frage. Akzeptabel ist einzig die Verwendung eines fixen, universellen Datenbankmodells, welches auch als allgemeine Datenstruktur des gesamten Modelldatenmanagements dient.
Das im folgenden näher vorgestellte Datenbankkonzept entstand auf der Basis des komponentenbasierten Simulationssystems SimCo ([Wie97]). Alle simulationsrelevanten Algorithmen sind dabei in Softwarekomponenten des Entwickungssystems Delphi 4.0 verpackt. Ein übergeordnetes Simulationssystem im herkömmlichen Sinne existiert nicht. Die Notwendigkeit einer flexiblen und komfortablen Modellmanipulation führten zur Integration einer Datenbank als zentrales Medium der Datenablage und auch zur Verhaltensdefinition des Systems.

Eine Besonderheit des verwendeten Datenbankmodells für Modell- und Simulationsdaten nach Abb. 1 ist die Ablage aller Daten mittels einer gleichen Tabellenstruktur. Dies bedeutet, daß die Daten zwecks akzeptabler Zugriffszeiten durchaus über mehrere Einzeltabellen oder auch verschiedene Datenbanken verteilt sein können, die Datenfelder jedoch stets mit gleichen Namen und Eigenschaften definiert sind. Der Grund für diese Generalisierung liegt in der damit verbundenen Vereinfachung aller folgenden Bedienoberflächen und funktionalen Implementierungen. So wird auch bei eigentlichen Simulationsdurchführung mit dem gleichen Datenmodell gearbeitet. Der Preis für die Vorteile der Generalisierung liegt natürlich in der relativ hohen Redundanz, welche aber bei den gegenwärtigen Speichergrößen von RAM und Festplatte als noch vertretbar erscheint.

Mit dem Attribut *Datatype* werden folgende grundlegende Datentypen unterschieden:

- Model – definiert eine Gruppe von Modellen in Sinne eines Gesamtexperiments,
- Version – ist eine konkrete Realisierung eines Simulationsmodells,
- Object – stellt die eigentlichen stationären oder dynamischen Objekte dar,
- Parameter – beinhaltet einen Parameter zu einem der vorstehen genannten Datentypen,
- Function – referenziert eine im Simulatorkern implementierte Funktion,
- Results – dienen zur Ablage von Simulationsergebnissen und –statistiken.

In jedem Datentyp können durch die Attribute Maintype und Subtype modell- oder anwendungsspezifische Klassen gebildet werden. So können in einem Modell zur Fertigungsmodellierung alle Maschinen als Hauptklasse mit einheitlichem Wert in Maintype und Differenzierungen der Maschinen mit verschiedenen Subtypen abgebildet werden. Die Relationen zwischen den einzelnen Datentypen werden durch die Attribute ID und ParentID hergestellt, d.h. alle Parameter eines Objektes erhalten dessen ID als ParentID zugewiesen.

Das Attribut DatasetID charakterisiert eindeutig alle zusammengehörigen Daten eines Modells und ist vorgesehen für das Kopieren, Löschen und Archivieren von kompletten Modellen.

	Datentypen				
***Field** : type*	**Model/Version /Run**	**Object**	**Parameter**	**Function**	**Results**
Datatype *: integer*	0	1	2	3	-1.. - N
ID (=DB-ID):long	= DatasetID	ObjectID	ParID	CalcID	ResultID
DatasetID *: long*	DatasetID	DatasetID	DatasetID	DatasetID	DatasetID
ParentID : *:long*	Nil	ModelVerRun	ObjectID	ObjectID	ObjectID
Name1 *: string*	Modelname	"Objectname"	"Parname"	"Mname"	""(empty)
Maintype *: long*	1=Model 2=Version	Objecttyp	Partype	Methodtype	Resulttype
SubType *: long*	0	Objectsubtype	Parsubtype	M.-subtyp	R.-subtype
TypeRef*: long*	RefID to Label	RefID to Label	RefLabel	RefLabel	RefLabel
plist *: long*	Parlist	Parlist	{SubParlist}	{ Parlist }	{ Parlist}
i *: long*	ModelID		IntValue1	IntValue1	IntValue1
j *: long*	RunID		IntValue2	IntValue2	IntValue2
k *: long*	VersionID		IntValue3	IntValue3	IntValue3
d *: double*		Actual Value	DblValue1	DblValue1	DblValue1
e *: double*	{Generation time}	{Gener. time}	DblValue2	DblValue2	DblValue2
f *: double*	{Last Change}		DblValue3	DblValue3	DblValue3
s *: string*	**Prototype model**	**Prototype object**	StrValue1	StrValue1	StrValue1
c *: string*	Coment2	Coment2	StrValue2	StrValue2	StrValue2
p1*: string*	Short Form of Properties	Short Form of Properties	Short Form of Prop.	Short Form of Properties	Short Form of Prop.

Abb. 1 Das universelle Datenformat aller Modell- und Ergebnisdaten

Für die Ablage der eigentlichen Nutzdaten jedes Modelldatenelementes sind jeweils drei Felder von Ganzzahlen (i,j,k), von Gleitkommazahlen (d,e,f) und von zwei Strings (s,c) vorgesehen. Diese Definition von jeweils 3 Zahlenwerten basiert auf dem sehr häufigen Auftreten von Wertetripeln, so z.B. bei der Koordinatendefinition mit x,y,z, bei der Farbcodierung (Werte für RGB) oder Ablage von statistischen Verteilungen mit Verteilungstyp, Mittelwert und Streuung. Prinzipiell besteht bei der Zuordnung freie Auswahl.

Zur Unterstützung verschiedener Anwendungskontexte existiert in Ergänzung dieser Datenstruktur eine Tabelle mit Texten zur fachgebietsspezifischen Bezeichnung der Nutzdaten. Die Kopplung der Daten mit den Bezeichnungen der Felder i bis c erfolgt über das Attribut TypeRef, welches jeden Datentyp eindeutig eine Referenznummer und damit einem Set von Bezeichnern (Label) zuordnet.

Der Schlüssel zur flexiblen Definition von fachgebietsspezifischen Simulationsmodellen liegt im Verweis auf einen Prototypen im Attribut s der Modellversion. Die Prototypdefinition stellt dabei ein Metamodell dar. Jedes diese Metamodelle definiert:

- die spezifischen Modellbausteine mit frei wählbaren Namen und einer zugeordneten Liste von Parametern,
- einer Sequenz von Funktionsreferenzen, mit welcher das Verhalten der Bausteine modelliert wird,
- und eine Liste von Texten, welche als Label für Anpassung der Bedienoberfläche für die jeweilige Thematik dienen.

Beim Aufbau eines Simulationsmodells ist zuerst die Zuordnung zu einem Modellprototypen zu treffen. Danach können die einzelnen Modellobjekte angelegt werden und diesen ebenfalls ein bestimmter Objekttyp aus dem Prototyp-Metamodell zugewiesen werden.

Bei der Anlage der Objektparameter werden nicht automatisch alle Parameter des Prototypobjektes übernommen, sondern können teilweise auch ignoriert werden. Es erfolgt dann bei der Simulationsdurchführung ein Zugriff auf den Wert des Objekt-Prototypparameters. Falls auch dieser nicht definiert ist, wird ein Default-Parameter des gesamten Prototypmodells verwendet. Der Vorteil dieser Parameterhierarchie liegt in der starken Reduzierung der notwendigen Parameterangaben. So müssen grundlegende Parameter wie Reportoptionen oder die Defaultvorgaben für Kapazitäten nur einmal im Metamodell definiert werden und gelten dann für alle späteren Modellobjekte, falls keine abweichenden Parameter angelegt sind. Dieses an die objektorientierte Vererbung angelehnte Prinzip erlaubt auch eine sehr einfache parametrische Modellierung, wodurch beispielsweise die Abhängigkeit des Durchsatzes eines Bediensystems von der Größe der Pufferlager durch schrittweises Ändern des Prototypparameters Puffferlagerkapazität einfach dargestellt werden kann.

3. Die Modellierungszugänge

Bei der Handhabung und Manipulation der Modelldaten lassen sich infolge der verwendeten Datenbank drei grundlegende Formen des Zuganges unterscheiden.

3.1 Direkter Datenbankzugriff

Ein generell verfügbarer, primärer Zugang ist direkt über die bereits in den aktuell verfügbaren Datenbankprodukten integrierten Tabellen- und Abfragen gegeben. Einfache Modelle können somit auch direkt durch Eingaben in die Datenbanken erfolgen.

Obwohl dieser Zugang aus Komfortgründen und eines hohen Fehlerpotentials eher ungünstig erscheint, ist er für die Verifizierung und Kontrolle der aufsetzenden Modellierungstools von grundlegender Bedeutung.

Abb. 2 zeigt eine mit Delphi erstellte, einfache Datenbankmaske, welche in sehr kurzer Zeit auch mit anderen Datenbankoberflächen entwickelt kann.

Der direkte Datenbankzugriff über ODBC auf die gegenwärtig verwendete Access-Datenbank ist gleichzeitig auch die Schnittstelle für externe Applikationen, welche Modelle generieren oder modifizieren können.

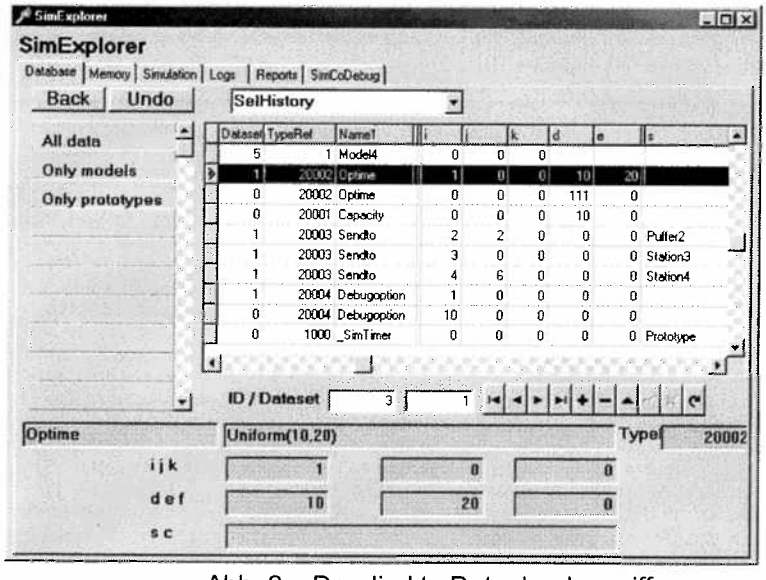


Abb. 2 Der direkte Datenbankzugriff

3.2 Universelles Modellierungstool mit teilweiser fachspezifischer Anpassung

Die Datenbank dient außer zur Ablage der fertigen Simulationsmodelle auch zur Definition der fachspezifikbestimmenden Metadaten. Vor allen für letztere Aufgabe ist neben dem direkten Datenbankzugriff auch eine etwas komfortablere, universelle Bedienoberfläche sinnvoll.

Ein dem Komfort bekannter Bausteinsimulatoren etwa ebenbürtiger Modellierungszugang kann durch auf die Datenbank aufsetzende Formularmasken oder spezielle Datenbankapplikationen erfolgen.

Die Existenz und der Umfang dieser zusätzlichen Datenbankoberfläche stellen jedoch keine notwendige Voraussetzung für die Durchführung einer Simulation dar.

Das gegenwärtig verwendete universelle Entwicklungswerkzeug verfügt über eine Maske zur Anzeige und Manipulation der Modelldaten (Abb. 3), über eine Steuermaske für die Kontrolle der Simulation und zur Ausgabe von Simulationsmeldungen und über eine universelle Reportmaske zur Auswertung des Simulationstraces.

In der Anzeigemaske wird das gesamte Modell mittels eines hierarchischen Baumes dargestellt. Der Anwender kann dabei sehr differenziert die Sichtbarkeit einzelner Modelldaten einstellen. Bei einer Selektion eines Eintrages wird dieser in der rechten Seite mit allen Einzelwerten und mit zugehörigen Bezeichnern dargestellt. Alle Bezeichner können wahlfrei über den im Zweig Prototypes befindlichen Eintrag _Labels definiert werden. Nur bei Existenz eines Bezeichners für ein Nutzdatenfeld i bis c wird auch dessen Sichtbarkeit in dieser Maske zugelassen. Damit ist auf dieser Ebene bereits eine einfache fachspezifische Ausrichtung gegeben.

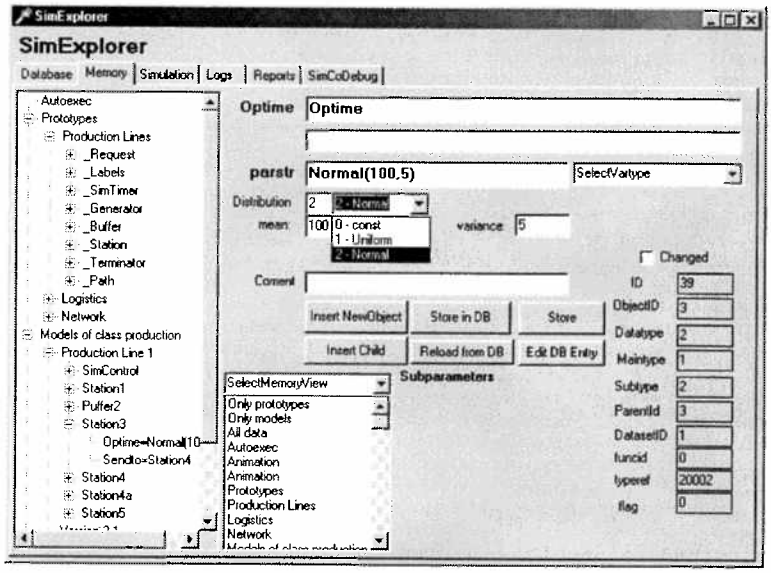


Abb. 3 Das universelle Modellierungswerkzeug

3.3 Spezielle Benutzeroberflächen

Für sehr spezielle Anwendungsfälle oder für immer wiederkehrende Standardsimulationen lassen sich genau auf die Thematik ausgerichtete Benutzeroberflächen generieren. In Anlehnung an im Officebereich übliche Assistenten oder an bekannte Modellgeneratoren im Simulationsbereich (vgl. [Eck99]) können auch entsprechende Modellierungsassistenten erstellt werden. Meist reicht für diese Aufgabe eine einzige Maske mit den entsprechenden Eingabefeldern aus. Die Assistenten legen nach erfolgter Eingabe der Ausgangsparameter und einer Plausibilitätsprüfung in der Datenbank die entsprechenden Modellobjekte und deren Parameterlisten an.

In Abb. 4 ist ein Assistent für die Modellierung von flexiblen Fertigungssystemen dargestellt. Eingabeparameter sind in diesem Fall die Anzahl Stationen und die Verweise auf die notwendigen Datenbanktabellen für die Arbeitspläne, Aufträge und Umrüstpläne der Maschinen.

Zur anwendungsspezifischen Visualisierung und Animation kann eine entsprechende Maske mit Ausgabefeldern für Statuswerte (vgl. Ab. 5) und wichtige Betriebsgrößen angelegt werden. Die Ausgabefelder werden von dem unsichtbar im Hintergrund laufenden Simulationskern in einstellbaren Zeitabständen mit Werten bestückt.

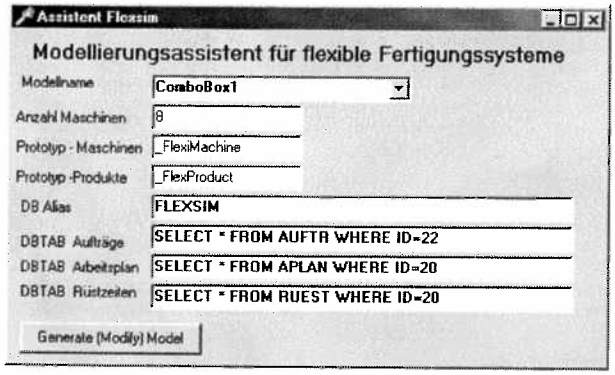


Abb. 4 Ein Modellierungsassistent

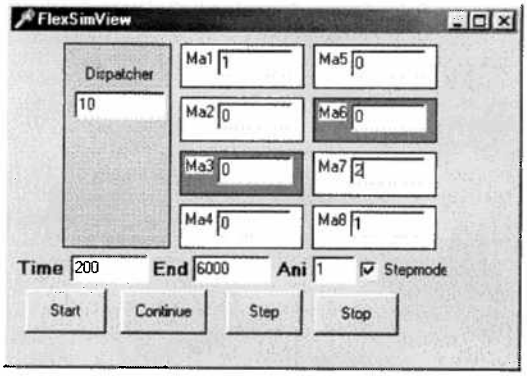


Abb. 5 Interaktive Zustandsvisualisierung

Da diese Visualisierungsmasken auf Standardcontrols von Windows basieren und eine bidirektionale Verknüpfung zwischen Anzeigeelement und dem zugehörigen Simulationsobjekt existiert, können auch Interaktionen des Anwenders zur Laufzeit auf das Simulationsmodell zurückgeführt werden. So kann durch Anklicken ein Ausfall der Maschine hervorgerufen werden, worauf das Simulationsmodell entsprechend reagiert.

4. Modellierung des Bausteinverhaltens

In der Einleitung wurde bereits die Problematik der Verifizierung des gesamten Computermodells auf der Basis von vordefinierten Modellbausteinen erwähnt. Ein gewisser Widerspruch ist dabei die einerseits gewünschte Bausteinabstraktion und Überschaubarkeit während der Modellierung und zum anderen die Notwendigkeit in Einzelfällen über sehr genaue Kenntnis der Bausteinrealisierung zu verfügen.

Im Kombination mit datenbankbasierten Modellablage und –manipulation kann dieses Problem zumindest teilweise durch die nochmalige Unterteilung der Bausteinimplementierung in Subbausteine gelöst werden. In Anlehnung an eine ähnliche Vorgehensweise im Mikroprozessorbereich werden diese Subbausteine nachfolgend als Simulations-Mikrofunktionen bezeichnet. Das Subjekt jeder Mikrofunktion ist das entsprechende Modellobjekt, an welches wiederum andere Objekte übergeben werden und oder welches Nachrichten austauscht.

Grundlegende Simulations-Mikrofunktionen sind :

- **Entry** Prüft, ob ein Eintritt erfolgen kann und übernimmt das übergebene Objekt bei erfolgreicher Prüfung,
- **EntryRequest** Führt nur die Prüfung durch und meldet das Ergebnis zurück,
- **Operation** Realisiert eine zeitverbrauchende Operation,
- **Sendto, TrySendto** Versucht das Objekt an den oder die Nachfolger zu übergeben, ruft dazu die Enter-Funktionen der Nachfolger auf, TrySendto testet nur ab,
- **Generate** Erzeugt neue dynamisch Objekte,
- **Test** Führt einen logischen Test aus und verzweigt entsprechend,
- **Terminate** Entfernt das Objekt aus dem System,
- **Animate** Führt eine optionale Animation oder einfache Visualisierung aus.

Der Umfang der entsprechenden Sourcetexte umfaßt bei den einfachen Mikrofunktionen nur einige Dutzend Zeilen. Die Schnittstelle ist generell gleich und wird bei der gegenwärtigen verwendeten Implementierungssprache Delphi 4.0 durch die Übergabe und die Rückgabe des Standarddatentyps TSimObject in der Form

```
function TSimObject.EnterRequest(input:TSimObject) : TSimObject; // Microcodefunction
```

dargestellt. Im Beispiel stellt der Übergabeparameter *input* ein dynamisches Objekt dar, welches das System durchläuft. Der Codeumfang dieser Mikrofunktionen soll mit Absicht sehr gering und überschaubar gehalten werden, damit bei einer Veröffentlichung der Sourcetexte durch den qualifizierten Anwender eine Verifizierung erfolgen kann.

Durch das einheitliche Interface aller Mikrofunktionen und mit einem dem Late-Binding ähnlichen Registrierungsverfahren können neue Funktionen sehr einfach zur Laufzeit eingebunden werden. Damit wird die Voraussetzung für eine Erweiterung der vorhandenen Funktionsbibliothek durch den Anwender geschaffen. Ziel der laufenden Tests ist die Schaffung einer ausreichenden Menge von Mikrofunktionen, deren Kombination für alle an Bediensysteme angelehnten, fachspezifischen Modelle ausreichend sind.

Zur Modellierung verschiedener fachspezifischer Aufgaben wurden einige Metamodelle mit einem typischen Set an Bausteinen definiert, so

- Reihenfertigung mit Eingangslager , Puffer , Maschine, Transporter, Ausgangslager;
- Flexible Fertigung mit Auftragspool, Dispatcher , Fertigungszellen mit integriertem Pufferlager;
- Logistik mit Quelle, Knoten, Transportkante, Senke.

Da sich diese Systeme auf ähnliche Bediensysteme zurückführen lassen, sind auch die Funktionssequenzen relativ ähnlich. Durch die gleiche Basistechnologie zur Modellbeschreibung und Simulation lassen sich auch Modelle auf der Basis verschiedener Metamodelle koppeln.

5. Die Architektur des Gesamtsystems

Durch die Offenheit der gesamten datenbankbasierten Metamodell- und Modellablage stellt sich das Gesamtsystem als relativ lose Kopplung verschiedener Softwaremodule dar (Abb. 6). Insbesondere in den Bereichen der Modellgenerierung und Simulationsauswertung können sehr rasch zusätzliche Applikationen auf die Datenbank aufgesetzt werden.

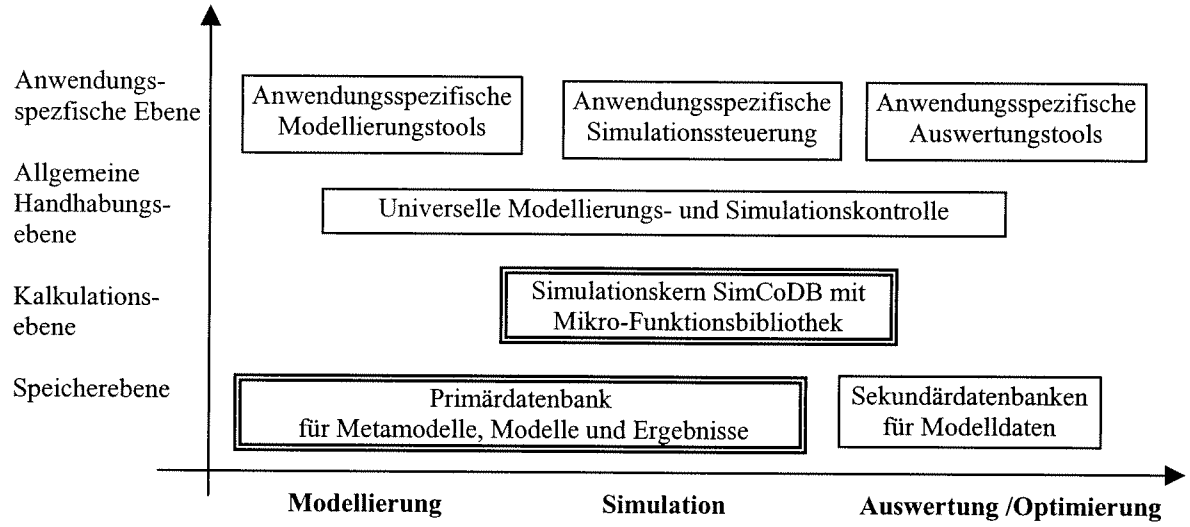


Abb. 6 Die Systemarchitektur

6. Zusammenfassung und Ausblick

Die Verwendung einer Datenbank als allgemeines Speichermedium für alle während der Modellierung und Simulation anfallenden Daten hat sich als sehr leistungsfähig und flexibel erwiesen. Gewisse Performanceprobleme treten nur bei der Speicherung sehr umfangreicher Simulationsresultate in Form von Tracedaten auf. Hier muß mehreren Läufen im Rahmen von Optimierungen eine Kompression auf statistische Werte bereits im Speicher erfolgen.

Die fachspezifische Ausrichtung über die Definition von Metamodellen erlaubt eine beliebige Anzahl von Modellierungsbausteinen. Da die Ablage und Handhabung dieser Metamodelle identisch ist mit denen der eigentlichen Modelle, erfordert dies keinen zusätzlichen Einarbeitungs- und Lernaufwand beim Anwender.

Die Modellierung des Bausteinverhaltens erfolgt allgemein durch Sequenzen von Simulations-Mikrofunktionen als Referenzen innerhalb der Datenbank. Bis zu dieser Ebene ist keine Programmierung notwendig. Neue Mikrofunktionen können mit dem Entwicklungssystem Delphi erstellt und eingebunden werden. Es dabei auch prinzipiell möglich, anstelle des SimCo-DB-Simulationskerns andere Produkte zu nutzen, wenn diese die Funktionalitäten der Simulations-Mikrofunktionen vollständig umsetzen können.

Die aktuellen Arbeiten konzentrieren sich auf die Optimierung der Simulations-Mikrofunktionen und die Schaffung einer zweiten, universellen Benutzerschnittstelle zum Einsatz im Internet.

Literatur

[Eck99] Eckardt F.,Palleduhn P.,Gmilkowsky P.: Konzeption und Entwicklung eines objekt-orientierten und wissensbasierten Simulationssystems zur automatischen Generierung von Simulationsmodellen diskreter Produktionsprozesse. In Proceedings der Tagung „Simulation und Visualisierung '99" an der Universität Magdeburg, 4.-5. März 1999, S. 225-2238

[Rabe99] Rabe, Markus: Beginnt ein neues Zeitalter der Simulation ? In Proceedings der Tagung „Simulation und Visualisierung '99" an der Universität Magdeburg, 4.-5. März 1999, S.3-18

[Wie97] Wiedemann, T., "Perspectives of Component-based Modelling and Simulation, Proceedings of the WCSS (Singapore, Sep. 1-3 1997)

Modellierung von Netzen durch Graphgrammatiken

Oliver Noll
Institut für Informatik
Johannes Gutenberg-Universität Mainz

Einleitung

In Zusammenarbeit mit der Drägerwerk AG in Lübeck entwickeln die Klinik für Anästhesiologie und das Institut für Informatik der Johannes Gutenberg-Universität Mainz ein Expertensystem zur automatischen Entwöhnung von beatmeten Patienten. Der Zustand eines Patienten wird im Expertensystem über Parameter wie etwa die arteriellen Drücke des Kohlendioxids und des Sauerstoffs dargestellt. Das regelbasierte System bestimmt in Abhängigkeit von diesem Patientenzustand in „Wenn-Dann-Regeln" die einzustellenden Beatmungsparameter. Dabei ergaben sich folgende Probleme:

- Einzelne Parameter sind nicht ständig meßbar und müssen daher in Regeln abgeschätzt werden.
- Es erwies sich als schwierig, diese Parameter in Regeln abzuschätzen.
- Durch die ungenauen Abschätzungen verschlechtern sich die Entscheidungen.

Dies führte dazu, in das Expertensystem ein Modell der Atmungsphysiologie (kurz: physiologisches Modell) zu integrieren. Wir entschieden uns, die Atmung druckgesteuert in einem Mehrkompartimenten-Modell darzustellen.

> Die Luft bewegt sich im System aufgrund von Druckdifferenzen. Diese entstehen wie bei einem Luftballon aus dem aktuell enthaltenen Volumen, der Dehnbarkeit, der sog. Compliance C und dem Widerstand, der als Resistance R bezeichnet wird. Um im Bild des Luftballons zu bleiben, besteht ein Problem bei der Modellierung darin, daß die Lunge nicht aus einem einzelnen Ballon besteht, sondern aus vielen kleinen, die durch unterschiedliche Compliance- und Resistancewerte verschiedenes dynamisches Verhalten zeigen. Solche kleine Teile der Lunge werden als Kompartimente bezeichnet. Genaueres dazu findet man etwa in [Nunn87].

Für die Modellierung erweisen sich Level-Raten-Modelle als geeignetes Mittel. Mit diesen ist man in der Lage, das dynamische Verhalten des Systems zu beschreiben. Ferner besitzen sie eine graphische Repräsentation in Level-Raten-Diagrammen, die im wesentlichen bipartite Graphen sind und als Netze bezeichnet werden. Diese Darstellung hat den Vorteil, das modellierte System veranschaulichen zu können. Ferner liefert sie die Möglichkeit, die Kontrolle darüber zu behalten, ob man syntaktisch korrekte Netze und damit auch syntaktisch korrekte Modelle erzeugt hat. Stellt man das Modell in seiner Gesamtheit als einen Graphen dar, so geht der ursprüngliche Vorteil der Anschaulichkeit wieder dadurch verloren, daß das Modell komplex ist.

Bei der Erstellung des physiologischen Modell als Level-Raten-Modell hat man sich für eine feste Anzahl von Kompartimenten zu entscheiden. Es wäre allerdings wünschenswert, eine Spezifikationssprache zu besitzen, die einem a priori nicht vorschreibt, wie viele Kompartimente zu generieren sind.

Diese beiden Probleme bei der Modellbildung lassen sich mit zweistufigen Graphgrammatiken beheben. In diesen Ausführungen werden wir am Beispiel des physiologischen Modells zeigen, wie sie sich zur graphischen Spezifikation eines komplexen Modells einsetzen lassen, so daß zum einen die syntaktisch korrekte Modellstruktur festgelegt ist, zum anderen sich Teilaspekte des Modells lokalisieren lassen, so daß die Darstellung übersichtlich bleibt und schließlich das Modell im Sinne der simulierbaren Gleichungen in der Attributierung bestimmt ist. Zusätzlich werden wir an einzelnen Stellen auf die Besonderheiten des physiologischen Modells eingehen.

Netze und Level-Raten-Modelle

Syntaktisch sind Netze bipartite Graphen mit zwei Sorten von Knoten und einer Relation zwischen diesen. Sie ist so angelegt, daß nur Knoten von unterschiedlichem Typ in Relation zueinander stehen können. Unabhängig von ihrer speziellen Interpretation haben alle Netze gemeinsam, daß eine Sorte der Knoten in irgendeiner Form Zustände und die andere allgemein Vorgänge repräsentiert. So auch bei Level-Raten-Modellen, die in unserer Anwendung im Vordergrund stehen und oft zusammen mit Netzen im allgemeinen bearbeitet worden sind (vgl. etwa [Perl83]). Die für uns wesentlichen Aspekte werden anhand eines Beispiels erläutert. Dabei ist die Diagramm-Darstellung an [Forrester72] angelehnt:

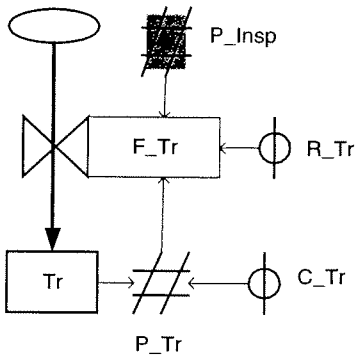


Abbildung 1: Ein Level-Raten-Diagramm

Das Diagramm zeigt die beiden wesentlichen Knotensorten: Levels, die den Zustand des Modells beschreiben, werden als Rechtecke und Raten, die die Flüsse angeben, als Ventile symbolisiert. Die Fluß-Relation zwischen beiden ist ein fett gezeichneter Pfeil. Die übrigen Knotensorten dienen als Hilfsmittel zur besseren Modelldarstellung. Die Systemumwelt wird in einer Quelle als Ellipse und Konstanten-Parameter mit einem durchgestrichenen Kreis dargestellt. Funktionale Anteile werden mit Hilfe des Kreuzsymbols veranschaulicht. Informationskanten sind dünn gezeichnet. Handelt es sich bei einem Knoten um einen Hyperknoten, der noch konkretisiert werden muß, wird er grau unterlegt gezeichnet.
Abbildung 1 stellt den Fluß in die Trachea dar. Der Zustand ist in dem Level „Tr" modelliert. Der Vorgang des Flusses von Luft beim Ein- und Ausatmen ist die Rate „F_Tr". Die Luft strömt aus der Umwelt in die Trachea und zurück. Wie oben bereits ausgeführt worden ist, modellieren wir die Atmung druckgesteuert. Der Druck ergibt sich als Quotient aus Volumen und Compliance. Daher wird funktionaler Knoten eingeführt. Im Beispiel entsteht der Fluß aus der Druckdifferenz von der Trachea („P_Tr") und dem von der Beatmungsmaschine zu jedem Zeitpunkt erzeugten Inspirationsdruck („P_Insp"). Dieser ist abhängig von einigen Parametern und eine Funktion f der Zeit. Er hat also in dem Diagramm noch nicht die Konkretisierungsstufe erreicht, daß er stets bestimmt werden kann. Er stellt einen Hyperknoten dar. Als Level-Raten-Modellgleichungen ergeben sich die Zusammenhänge des Diagramms folgendermaßen:

$$P_{Insp}(t) = f(t)$$
$$F_{Tr}(t, t+\Delta t) = \frac{1}{R_{Tr}}\left(P_{Insp}(t) - P_{Tr}(t)\right)$$
$$Tr(t+\Delta t) = Tr(t) + \Delta t * F_{Tr}(t, t+\Delta t); Tr(0) = Tr_0$$
$$P_{Tr} = \frac{Tr(t)}{C_{Tr}}$$

Gleichungen 1 : Die Modellgleichungen zu obiger Abbildung

Will man die Gleichungen in die Diagrammdarstellung integrieren, hat man die Knoten zu attributieren: Alle Sorten von Knoten erhalten das Attribut „Wert". Mit Ausnahme der Konstanten-Knoten wird für sie noch ein Attribut „Schaltregel" eingeführt zur Integration der zugehörigen Aktionen.
Das eben behandelte Beispiel stellt einen Ausschnitt aus dem Gesamtmodell dar, das mit Hilfe von Graphgrammatiken integriert wird. Dazu wird zunächst der Begriff der Graphoperationen erläutert.

Graphoperationen

Wir verwenden Graphoperationen und Graphgrammatiken in der von Göttler eingeführten Darstellung (vgl. [Göttler88]). Deren Kern ist, daß Graphoperationen selbst als Graphen angegeben werden. Dies erlaubt es, auf der höheren Spezifikationsebene die Darstellung der Modellebene beizubehalten und sich in der Graphoperation auf die wesentlichen Aspekte des Modells zu konzentrieren. Sind an die Knoten Attribute geknüpft, wird in die Graphoperation ein Attribut-Auswertungs-Mechanismus eingebaut. Dieser besteht aus den drei Teilen PRE, BODY und POST. Die Bedingung PRE, die auch mit boolschen Konstanten belegt werden kann, gibt an, ob eine Operation durchgeführt werden kann. POST ist eine Zusicherung, die nach Anwendung der Graphoperation erfüllt sein muß. Die wesentliche Arbeit findet im BODY-Programm statt. Dort werden die Werte der Attribute der bearbeitet.
Ohne auf die exakten Definitionen einzugehen, wird das Arbeiten mit Graphoperationen demonstriert, indem das begonnene Beispiel fortgesetzt und in Beziehung zu den mit ihm interagierenden Modellteilen gesetzt wird. Auf einer abstrakteren Ebene steht die Trachea über die Flußrelation mit

dem Fluß in der Lunge in Beziehung und ihr Zustand bestimmt auch wieviel Sauerstoff bzw. Kohlendioxid in ihr vorhanden hast. Dies führt zu folgender Graphoperation:

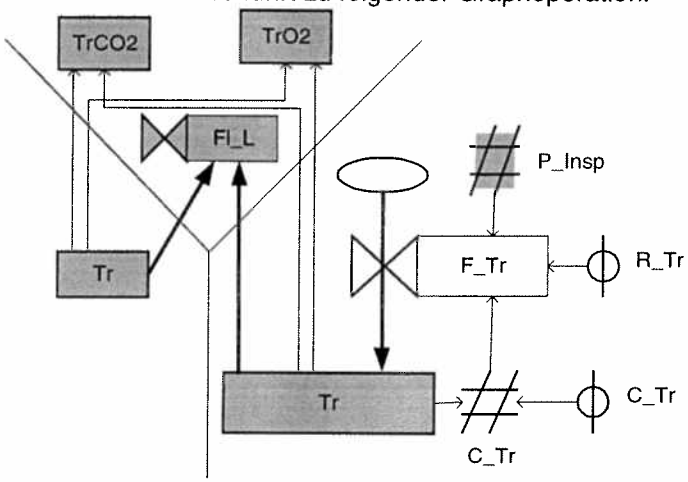


Abbildung 2 : Graphoperation zur Konkretisierung der Trachea

Durch das Metasymbol Y wird der Graph, der die Graphoperation darstellt in drei Abschnitte geteilt. Auf der sog. Linken Seite befinden sich die Teile der Graphoperation, die aus dem Wirtsgraphen gelöscht werden, auf den die Graphoperation angewandt wird. Auf der Rechten Seite stehen diejenigen, die in den Wirtsgraphen eingefügt werden. Durch den oberen Einbettungsbereich und den Kanten von der Linken bzw. der Rechten Seite dorthin wird angegeben, wie die alten Teile im Wirtsgraph vorhanden sein mußten bzw. wie die neuen Teile in den Wirtsgraphen integriert werden.

Im Beispiel wird die Regel folgendermaßen angewandt: Suche im Wirtsgraphen einen Hyper-Level-Knoten Tr, der die angegebenen Verbindungen zu den Knoten aus dem Einbettungsbereich hat. Lösche diesen und alle angegebenen Kanten. Füge dann die Rechte Seite ein und setze im Wirtsgraphen die neuen Verbindungskanten zwischen der Rechten Seite und dem Einbettungsbereich. Als Ergebnis hat der Wirtsgraph nach der Anwendung eine Form erreicht, die den Fluß zwischen Umwelt und Trachea terminal darstellt und dieser Teil im Prinzip ein simulierbares Modell repräsentiert.

Neben dieser Regel wirken auf die Trachea noch weitere. Daher wird der Hyper-Level-Knoten „Tr" durch einen gleichartigen ersetzt. Somit kann er in weiteren Regeln noch modifiziert werden. (Auf diesen Aspekt wird später nochmals eingegangen). In der zugehörigen Attributierung werden PRE und POST auf „true" gesetzt und die obige Attribute gemäß den Gleichungen 1 belegt. Da auf die Hyperknoten noch weitere Operationen angewandt werden, können deren Attributwerte nicht festgelegt werden. Der Graph der Graphoperation erlaubt in dem Beispiel darzustellen, wie der Fluß in die Trachea realisiert wird.

Mit derartigen Graphoperationen läßt sich allerdings noch nicht das gesamte Modell spezifizieren. Dies leisten zweistufige Graphgrammatiken.

Zwei-Stufen-Graphgrammatiken

Nach den Vorbereitungen des letzten Abschnitts sind wir in der Lage, in einer Graphoperation auf einen Aspekt des Modells einzugehen, ohne auf den Zusammenhang mit anderen Teilen zu verzichten, diesen aber auf das Notwendige reduzieren. Nunmehr wird gezeigt, wie man sich zweistufige Graphgrammatiken (kurz: ZGG) in unserem Zusammenhang zunutze machen kann. Zunächst wird die unter Verzicht auf die konkreten Definitionen informell in ZGGs eingeführt. Die Idee hinter ZGGs ist, Graphoperationen, sog. Metaregeln, solange auf Graphoperationen, sog. Hyperregeln, anzuwenden, bis eine anwendbare Regel entsteht. Eine ZGG besteht dann im wesentlichen aus einem Startgraphen, einer Menge von initialen Hyperregeln und einer Menge von Metaregeln. Die von dieser Grammatik erzeugte Sprache sind die Graphen, die sich aus dem Startgraphen dadurch gewinnen lassen, daß man auf die initialen Hyperregeln eine endliche Folge von Metaregeln anwendet, die zu einem Graphen ohne Hyperknoten führt.

Am Beispiel des physiologischen Modells demonstrieren wir dieses Vorgehen. Da von Anfang an klar ist, daß wir nur ein Modell erhalten wollen, genügt es eine initiale Hyperregel anzugeben. Sie überführt den Startgraphen in die grobe Modellstruktur: Das Modell besteht auf der abstraktesten Ebene aus drei Teilmodellen, die miteinander interagieren. Neben dem Ventilationsteil, der den Transport der Luft in die Lunge beschreibt, existieren das Herz und der Perfusionsteil, der den Gasaustausch von der Lunge ins Blut und vom Blut in die Zellen modelliert. Während sich der Ventilationsteil in Teilmodelle

für das gesamte Volumen, den Sauerstoff und das Kohlendioxid aufspaltet, sind bei der Perfusion nur der Sauerstoff und das Kohlendioxid zu beachten. Dies führt zu folgender Hyperregel, die durch das fett gezeichnete Y-Symbol kenntlich gemacht wird:

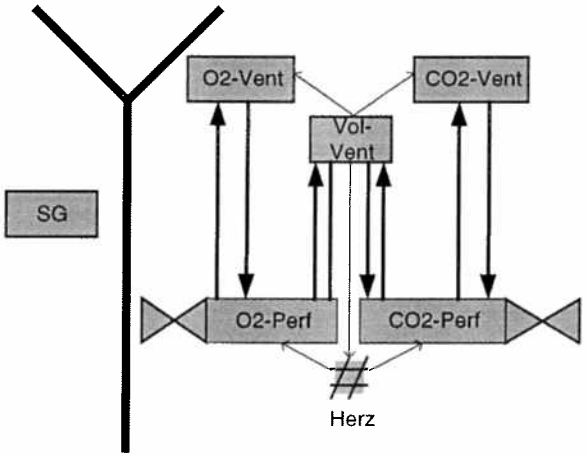


Abbildung 3: Initiale Hyperregel zur Modellstruktur

Schon die Zahl der aus- und eingehenden Kanten macht klar, daß das Volumen-Ventilationsmodell „Vol-Vent" eine zentrale Stelle im Modellentwurf einnimmt. Es steht mit allen weiteren Teilmodellen in irgendeiner Weise in Verbindung. Dies führt auf die Strategie zur Angabe der Metaregeln, die notwendig sind, bis ein Modell-Diagramm entsteht, das nur terminale Knoten hat und somit derart attributiert werden kann, daß es ein simulierbares Modell repräsentiert: Zunächst wird in einer Metaregel der aktuelle Aspekt im „Vol-Vent"-Modell verfeinert. Gibt es entsprechende Stellen oder erfordert er strukturelle Anpassungen in anderen Modellteilen, werden diese in passenden weiteren Metaregeln formuliert.
Dieses Vorgehen wird am Beispiel der ersten Verfeinerungsstufe der Ventilationsmodelle demonstriert. Die Ventilation läßt sich dadurch aufteilen, daß die Atemwege in einen oberen Abschnitt, die Trachea, und einen unteren, der Lunge, aufgeteilt werden und dazwischen ein Fluß stattfindet. Da diese Struktur auch in den Ventilationsmodellen für den Sauerstoff und das Kohlendioxid vorhanden ist, müssen entsprechende Regeln für diese formuliert werden. Hier wird exemplarisch die Metaregel, die am Y-Symbol mit Doppelstrichen zu erkennen ist, angegeben:

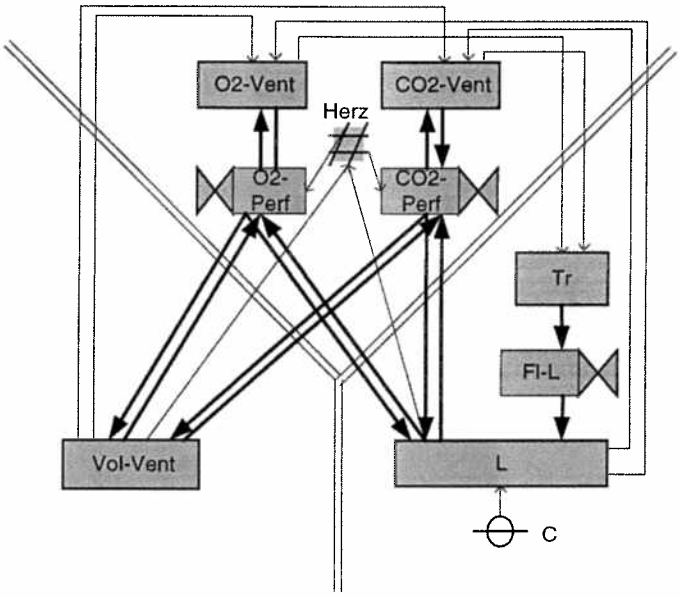


Abbildung 4: Metaregel zur Struktur der Atemwege

Diese Metaregel wird auf die initiale Hyperregel angewandt. Dadurch entsteht eine Hyperregel, die im Volumen-Ventilationsmodell den grundsätzlichen Aufbau der Atemwege zeigt. Für die analogen Regeln in den beiden anderen Ventilations-Modellen ist zu beachten, daß die in der Regel aus Abbildung 4 eingefügten Teile sich im Einbettungsbereich befinden, so daß die Regeln nur angewendet werden können, wenn

das Volumen-Modell bereits konkretisiert worden ist. Insgesamt ergibt sich nach Anwendung dieser Regeln eine Hyperregel die in allen drei Ventilationsmodellen die Struktur von der Rechten Seite der Abbildung 4 hat.

An dieser Stelle werden bereits diejenigen Konstanten-Parameter angegeben, die sich auf die gesamte Lunge beziehen und bei späteren Konkretisierungen der Lunge durch das Einfügen von Kompartimenten benötigt werden. Dies ist einerseits technisch notwendig, da sich bspw. die Compliance eines Kompartiments aus der Lungen-Compliance ergibt. Andererseits ist dieses Vorgehen auch für die Spezifikation des Gesamtmodell von Vorteil: Informationen werden auf derjenigen Abstraktionsebene ins Modell eingeführt, auf der sie auch in der Anwendung angesiedelt sind.

Schließlich zeigt diese Regel, wie Graphgrammatiken beim Erzeugen von Netzen von Nutzen sein können: Verfeinert man ein Netz um einen Knoten, so stimmt dessen Typ mit jenen an den Ein- und Ausgängen des eingefügten Teilnetzes überein. Da Graphoperationen in der Göttlerschen Darstellung einen anschaulichen Zugang bieten, erkennt man auf einen Blick, ob man syntaktisch korrekt gearbeitet hat.

Sind mit Hilfe der eben dargestellten Metaregeln die groben Strukturen der Ventilationsmodelle im Modell, d.h. der daraus entstandenen Hyperregel, vorhanden, kann die Trachea weiter konkretisiert werde, indem man die Regel aus Abbildung 2 als Metaregel interpretiert. Anschließend können Kompartimente eingefügt werden.

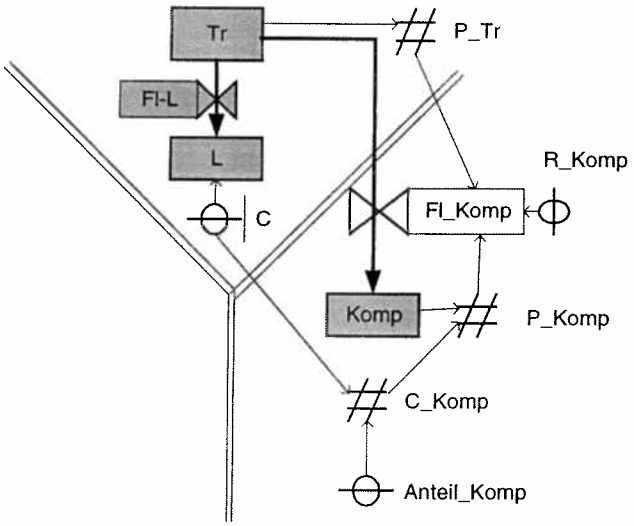


Abbildung 5: Metaregel zum Einfügen eines Kompartiments

Die Regeln zum Einfügen von Kompartimenten, die hier wieder exemplarisch für das Volumen-Teilmodell gezeigt wird, bringt in unserem Zusammenhang drei neue Aspekte. Auf der Anwendungsebene zeigt sie eine unserer zentralen Entwurfsentscheidungen. Wir haben jedem Kompartiment einen Konstanten-Parameter „Anteil-Komp" zugeordnet, der dessen Größe angibt. Dieser liefert die Compliance des Kompartiments und bestimmt damit dessen dynamisches Verhalten. Ferner ist er auch mit dem Herzen verbunden und regelt so den Anteil des Bluts, der in diesem Kompartiment am Gasaustausch teilnimmt. Diese Regeln konnten hier aus Platzgründen nicht angegeben werden. Für die Anwendung von Graphgrammatiken demonstriert Abbildung 5, wie Hyperknoten als Platzhalter verwendet werden. Dadurch daß die Hyperknoten „Fl_L" und „L" im Einbettungsbereich angegeben werden und nicht gelöscht werden, bleiben sie nach der Anwendung vorhanden. Insbesondere ist diese Regel im Prinzip beliebig oft anwendbar. Dies sichert, daß die Anzahl der Kompartimente im Modell frei wählbar ist.

An dieser Stelle hat das Modell eine Struktur erreicht, die bis auf die verbleibenden Hyper-Level-Knoten „Komp", „Tr" und „Fl-L" mit „L" die den gewünschten Aufbau im Ventilationsteil zeigt: Die Umwelt ist mit der Trachea verbunden und diese mit der gewünschten Anzahl von Kompartimenten.

Dann findet eine Regel Anwendung, die die Hyperknoten aus dem Modell entfernt. Eine ähnliche Platzhalterfunktion haben auch der Hyperknoten „Komp" und „Tr" in Abbildung 2. Zwar sind sie prinzipiell schon terminale Knoten, doch ist „Komp" noch nicht mit dem Perfusionsmodell verbunden und „Tr" wurde erst mit der Regel aus Abbildung 5 in Beziehung zu den Kompartimenten gesetzt. Daher werden sie noch nicht-terminal dargestellt. Vielmehr müssen noch Regeln formuliert werden, die dies leisten. Erst dann kann die endgültige Schaltregel in der Attributierung angegeben werden.

Insgesamt kommt man im vorgestellten Beispiel aus der initialen Hyperregel mittels folgender Strategie zu einer anwendbaren Regel:

1. Verfeinere das Volumen-Ventilationsmodell gemäß der Regel aus Abbildung 4.
2. Wende analoge Regeln für den Sauerstoff- und Kohlendioxidteil an.
3. Konkretisiere die Trachea in den drei Ventilationsmodellen (vgl. Abbildung 2).
4. Füge in den Ventilations-Modellen die gewünschte Zahl der Kompartimente ein.
5. Bestimme für die Kompartimente die „Ausgänge" des Herzens
6. und verfahre für die Perfusion entsprechend (nicht angegebene Regeln).
7. Wandle die verbleibenden Hyperknoten in terminale um.

Mit dem letzten Schritt wird die Attributierung der Knoten beendet. Damit hat man aus einer abstrakten Hyperregel, die die Modellstruktur angibt, eine Regel erzeugt, die den Graphen eines Modells konstruiert, das mit Hilfe seiner Attributierung ein simulierbares Modell des Systems der menschlichen Atmung beschreibt. Auf diese Weise ist allerdings nicht nur ein Modell entstanden, sondern es ist gleichzeitig eine Sprache entstanden, die alle in unserem Sinn möglichen Modelle enthält.

Ergebnisse

Die hier vorgeschlagene Methode zur Spezifikation eines Modells zeigt einige Vorteile. Durch die Darstellung der Graphoperationen als Graphen liefert sie ein anschauliches, aber zugleich formales Werkzeug zur Darstellung eines Modells. Insbesondere sieht man schnell durch Vergleich der Linken und Rechten Seiten, ob man syntaktisch korrekte Netze konstruiert hat. Durch das Einführen einer ZGG wird erreicht, daß man nicht nur ein Modell sondern gleich eine ganze Klasse von Modellen konstruieren kann. Diese entspricht den Graphen, die von der ZGG generiert werden können. Die Zweistufigkeit erlaubt es ebenso, in den Metaregeln einzelne Aspekte des komplexen Modells herauszugreifen und diese zu beschreiben, ohne einerseits das gesamte Modell aus den Augen zu verlieren, aber andererseits den Fokus auf den in dieser Regel zentralen Aspekt setzt. Schließlich gestatten die Hyperknoten die Kontrolle darüber zu behalten, ob der Graph ein simulierbares Modell repräsentiert. Erst wenn diese aus dem Graphen entfernt sind, stellt man ein simulierbares Modell dar. Dies führt auf die Grenzen des Ansatzes: Leider fehlt ein Tool, mit dessen Hilfe Graphgrammatiken interaktiv erzeugt werden können und diese dann anwendet. Damit hätte man die Möglichkeit in einem für die Modellbildung typischen Trial-and-Error-Verfahren die generierten Regeln in Simulationen zu überprüfen. Somit bleibt zur Zeit nur der Ansatz, wie er hier angewandt worden ist: Ein erzeugtes Modell nachträglich mit Graphgrammatiken zu spezifizieren.

Literatur

[Baumgarten90] Baumgarten, B.: Petri-Netze
Wissenschaftsverlag; Mannheim 1990.

[Bossel92] Bossel, H.: Modellbildung und Simulation
Vieweg-Verlag; Braunschweig 1992.

[Forrester72] Forrester, J.W.: Grundsätze einer Systemtheorie
Betriebswirtschaftlicher Verlag Dr. Th. Gabler; Wiesbaden 1972.

[Göttler88] Göttler, H.: Graphgrammatiken in der Softwaretechnik,
Informatik-Fachberichte Nr. 178
Springer-Verlag; Berlin 1988.

[Himmelreich95] Himmelreich, B.: Spezifikation von Zustand und Dynamik objektorientierter Datenbanken durch Graphgrammatiken;
Dissertation, Universität-Mainz, Mainz 1995.

[Nunn87] Nunn, J.F.: Applied Respiratory Physiology, Third Edition;
Butterworths, London 1987.

[Reisig82] Reisig, W.: Petrinetze
Springer-Verlag; Berlin 1982.

[Reisig85] Reisig. W.: Systementwurf mit Netzen
Springer-Verlag; Berlin 1985.

[Perl83] Knitelius, U., Perl, J. und Schröder H.-J-: Implementierung interpretierter Netzvergröberungen.
Osnabrücker Schriften zur Mathematik, Reihe I, Heft 13; Osnabrück 1983.

Modellierung grafischer Benutzerschnittstellen für Internetapplets

Dietmar P. F. Möller[1,2)], Hans-Dietrich Doebner[3)], Markus Lamers[2)],
Sascha Aderhold[2)], Mathias Reuter[2)]
[1)]Universität Hamburg, Fachbereich Informatik, Arbeitsbereich Technische Informatiksysteme,
Vogt-Kölln-Str.30, D-22527 Hamburg
[2)]TU Clausthal, Institut für Informatik, Julius Alberts Str. 4, D-38678 Clausthal-Zellerfeld
[3)]TU Clausthal, Arnold Sommerfeld Institut für Mathematische Physik, Leibnizstr. 24, D-38678 Clausthal

1 Einleitung

Bei Durchsicht der auf Web-Servern eingerichteten Informationsseiten fallen einem deutliche Unterschiede hinsichtlich Gestaltung und Schwerpunktlegung der dargebotenen Informationen auf, wobei vielfach gegen elementare Regeln zur Präsentation komplexer Informationen verstoßen wird. So wird die Information vielfach nicht so aufbereitet, das sie intuitiv verständlich ist, bzw. daß Navigieren in der in Webstruktur transparent bleibt oder zumindest erleichtert wird. Die Feststellung "lost in space" ist dabei eine häufige Realität und frustrierend für den Nutzer. Um diesen Nachteil zumindest teilweise zu beheben, wurde im Rahmen eines Forschungsprojektes ein erster methodischer Ansatz entwickelt, Internet-geeignete Benutzerschnittstellen, unter Berücksichtigung kognitionswissenschaftlicher Methoden, zu entwerfen. Durch Integration der Erkenntnisse aus unterschiedlichen Bereichen der informationstechnischen, mulitimedialen und studienberatenden elektronischen Informationsdarbietung soll die Verarbeitungseffizienz der Informationsinhalte klientelbezogen gewährleistet werden. Als Ergebnis dieser Arbeiten sollen "Richtlinien" für die Realisation benutzerfreundlicher Navigationskonzepte und Orientierungshilfen abgeleitet werden.

Als zusätzliche Eigenschaft soll in dem zu entwickelnden Informations- und Studiensystem der Benutzerklientel die Möglichkeit gegeben werden, durch einen Dialog mit dem Web-Master interkommunikativ aus der jeweiligen Informationspräsentation heraus mit Hilfe einer sog. "Notizbuchfunktion" (zunächst in schriftlicher Form) zu kommunizieren. Die von der Benutzerklientel gestellten Fragen und Anmerkungen dieser Notizbuchfunktion sollen der übrigen Klientel (nach Sichtung durch den Web-Master) zur Verfügung gestellt werden. Aus dem Aufbau einer derartigen Kommunikationsstrecke versprechen sich die Antragsteller, daß die Klientel in ihrem Lernprozeß unterstützt wird und die Lehrenden auf etwaige Unzulänglichkeiten des Lehr- und Informationsangebotes aufmerksam gemacht werden. Da diese Notizbuchfunktion direkt mit dem Informations- und Studiensystem gekoppelt ist, ist ein direkter Bezug zwischen der Informationsmitteilung und dem Server gegeben, d.h. die Benutzung von weiteren Kommunikationsmitteln entfällt, was u.a. zu einer Akzeptanzerweiterung des zu entwickelnden Informations- und Studiensystems führen sollte.

Damit gewinnt dieses System auch für den Studienstandort Deutschland und die hier studierenden Ausländer, ein langfristig wirkendes Kapital, an Bedeutung, die nicht unterschätzt werden darf. Die sinkende Zahl dieser Goodwill-Träger ist ein Alarmsignal, dem begegnet werden muß. Neben hausgemachten Schwierigkeiten, wie beispielsweise dem international nicht vergleichbaren Studien- und Abschlußsystem, sind es vor allem Sprachbarrieren, die in der Vergangenheit als Hemmschwelle erkannt wurden.

Aber auch die wachsende internationale Konkurrenz der Bildungseinrichtungen, die mit weltweitem Marketing und dienstleistungsorientiertem Serviceangebot Studenten an sich binden, zieht Studenten von Deutschland ab. So zum Beispiel beschlossen 1996 in den USA 21 Regierungschefs der westlichen US-Staaten die Gründung einer virtuellen Universität und unterstützten damit die Idee des Distance Learning bzw. des Virtual Campus als eine Möglichkeit der universitären Ausbildung über große Distanzen hinweg. Die „Western Governors University" ist eine virtuelle Hochschule, die mit Hilfe von Internet und anderen Kommunikationstechniken kosteneffektiv Lehre überträgt und einen akademische Grad vergibt. Damit soll der Zugang zur Lehre einer über weite Gebiete verstreuten Menge von Studenten ermöglicht werden, die auf andere Weise keinen Zugriff hätten. Zum zweiten fördert eine kurze Recherche im WWW sofort hunderte von mehr oder weniger großen oder guten Veranstaltungen zum Thema Tele-Lernen zu Tage und zahlreiche internationale Organisationen und Institutionen beschäftigen sich ebenfalls damit. Deutsche Universitäten sind auf diesem Weltmarkt „Aus- und Weiterbildung" dagegen bisher wenig präsent, was durch das beantragte Projekt zumindest in teilen wettgemacht werden soll.

2 Visuelle Informationsverarbeitung

Bei der Organisation von Objekten zu Einheiten folgt der Mensch in der Regel den aus der Gestalt-Psychologie bekannten Gestaltgesetzen der Wahrnehmungsorganisation. Danach nimmt man beispielsweise eher vier Paare von Linien und nicht etwa acht einzelne Linien wahr. Jedoch reicht es in der visuellen Informationsverarbeitung nicht aus, die visuelle Welt in Objekte zergliedern zu können, man muß vielmehr diese Objekte dann auch noch identifizieren können, was im Kontext der Merkmalsanalyse zur Objekterkennung zu sehen ist. So können z.B. Objekte nach den diesen zugrundeliegenden zylindrischen Formen gegliedert werden, was Kontextinformation in den Zusammenhang zum Mustererkennen stellt. Darüber hinaus muß beachtet werden, das die visuelle Aufmerksamkeit jeweils nur zu einem gewissen Teil des visuellen Feldes korreliert ist, womit der aufmerksamkeitsgesteuerten Nutzerführung bei der Auslegung internetbasierter Informationssysteme eine besondere Bedeutung zukommt.

Die Analyse der existierenden Informationssysteme im Rahmen einer Vorstudie zeigte, daß multimediale Datenbanken, wie z.B. die WWW-Server, CD-Roms und ZIP-Disketten Zugriffsmedien bieten, die schnell genug sind, um einem Benutzer z.B. einen virtuellen Besuch eines Unternehmens oder einer Universität zu gestatten. Oft ist es möglich, sich über verschiedene Berufsbilder und -felder zu informieren und sich anhand der Lehr- und Studienpläne ein Bild des erwünschten Tätigkeitsfeldes zu machen bzw. sich über die verschiedenen Berufe soweit zu informieren, daß der persönlichen Neigung entsprechend Ausbildungs- und Studieninteressen definiert werden können. Außerdem läßt sich speziell ein Studienwunsch mit im virtuellen Campus zu hinterlegenden Studienmaterial, z.B. Vorlesungsskripte oder Praktikumsanleitungen realisieren und erproben. Auch kann sich ein Student in Fachvorlesungen ,,einloggen", um fehlendes Wissen schnell zu erwerben und vorhandenes Wissen weiter auszubauen.

Der Zugriff auf Informationen ist standardisiert und dessen Bedienung leicht erlernbar. Die Darbietungen im Bildungsbereich sind aber oft nicht allgemeinverständlich, wodurch sich im Verbund mit der Komplexität der unterschiedlichen Präsentationen unterschiedlicher universitärer Einrichtungen gerade neue Benutzer leicht verloren fühlen („lost in space") und sich deshalb die angebotenen Informationen nur rudimentär oder gar nicht aneignen. Wesentlicher Schwachpunkt, so zeigen Studien, ist dabei neben der groben Verletzung der durch die Kognitionswissenschaften evaluierten grundlegenden Prinzipien der Mensch-Maschine-Kommunikation, die fehlende Aufmerksamkeitssteuerung eines "unbedarften" Bedieners, die aus dem Fehlen einer immer geltenden generellen Struktur der Informationsstrukturierung resultiert. Solange die Struktur der WWW-Server, die CD-Rom und ZIP-Disketten diese Mängel aufweisen, ist nur mit einer beschränkten Ausschöpfung dieses Informationspotentiales zu rechnen. Die ,,artgerechte" und in sich immer konsistente Gestaltung der Front-End-Oberfläche ist daher eines der dringendsten Probleme, bevor diese neuen Techniken im universitären und nicht-universitären Betrieb standardmäßig eingesetzt werden können. Dies gilt auch für ein Metainformationssystem mit integrierten virtuellen Campi für die allgemeine Wissensaquisition, falls derselbe Lerneffekt erzielt werden soll, wie dies bei einem „realen" Besuch einer Lehrveranstaltung oder bei einer ,,klassischen" Wissensaquisition der Fall ist. Dabei muß ein besonderes Augenmerk darauf gerichtet sein, daß die Benutzer des Metainformationssystems sich aus dem System heraus an den Betreiber des Systems mit Anmerkungen und Fragen wenden können und diese Informationen entsprechend der übrigen Klientel zur Verfügung stehen. Nur so ist eine laufend, sich adaptiv an die nötigen Informationsstrukturen moderner Informationssysteme anpassende, Informationsdarbietung zu gewährleisten.

Die Umsetzung des Einhaltens von Standardauslegungsregeln, die den ,,mental-overload" der Benutzer schon in der reinen Wissensdarbietungsphase minimieren, stellt einen weiteren unbedingten Schritt bei der Akzeptanzerweiterung und dem lehrbezogenen Einsatz dieser Informationssysteme dar. Da das grundlegende Wissen größtenteils in den verschiedenen informationstheoretischen Wissenschaftsrichtungen bereits erworben wurde, sollte dieses nunmehr zusammengefaßt und anhand der Realisierung eines leicht verständlichen Prototyps gezeigt werden, wie die Umsetzung dieser Erkenntnisse zu verbesserten Servern führt. Bewußt sollte dabei auch auf den Inhalt vorhandener Informationssysteme zurückgegriffen werden und die praxisorientierte Auslegung eines Prototyps dadurch umgesetzt werden, daß Strukturen bereits bewährter Informationssysteme übernommen werden sollen.

3 Wissensrepräsentation

Die Entwicklung kognitionswissenschaftlich begründeter Wissensrepräsentaten kann anhand der Mehr-Speicher-Konzeption erfolgen, welche ein sog. sensorisches oder auch Ultrakurzzeitgedächtnis, ein Kurzzeitgedächtnis und ein Langzeitgedächtnis berücksichtigt. Um komplexe mentale Bilder aus Einzelteilen zusammenzusetzen, reicht die Kapazität des Kurzzeitgedächtnisses in der Regel nicht aus, weshalb man bei Internetbasierten Applets die Einzelinformationen in größere Einheiten zusammenführen, den sog. Chunks, zusammenführen sollte. Chunks sind dabei eine Einheit der Wissensrepräsentation und werden aus einer gewissen Anzahl primitiver Einheiten (Propositionen) zusammengesetzt. Sie bilden aber gleichzeitig die Basiseinheiten komplexerer Gebilde. So gesehen bilden sie eine Form der hierarchischen Struktur mentaler Vorstellungen.

Darüber hinaus können spezifische Erfahrungen und statt dessen die Kategorisierung durch Merkmale und Kennzeichen einer allgemeinen Erfahrungsklasse als konzeptuelles Wissen eingeführt werden, welches durch die modernen objektorientierten und verbundenen objektrelationalen Paradigmen realisiert werden kann.

Ausgangspunkt der Entwicklung des vorgeschlagenen adaptiven multimedialen Lehr- und Studiensystems im Kontext eines virtuellen Campus ist die kritische und vergleichende Analyse der vorhandenen Studiensysteme und ihrer anzusprechenden Klientel durch Benutzermodelle (User-Models, Human Performance Models) sowie der Zusammenhänge von Eigenschaften der Bedienoberfläche (Informationsdichte, Informationsrelevanz) und deren erreichter Akzeptanz-status. Daraus können Kernfragen zu folgenden Problembereichen formuliert werden:

1. Ursache für Fehlbedienungen,
2. Ursachen des „Verlorengehens" in den Informationssystemen,
3. Möglichkeiten der Informationsaufbereitung und -darstellung zur Vermeidung von Fehlhandlungen, Fehlschlüssen und des sog. „mental overloads".

Im Bereich 1 werden anhand von bekannten Fehlbedienungen sowie einer Fehlertaxonomie die Situationen definiert, unter denen Fehlbedienungen auftreten können. Bekannte ergonomische Gestaltungsregeln werden in die Auslegung der Bedienoberfläche sowie in der Präsentation der Informationen integriert.

Im Bereich 2 wird anhand von bestehenden Informationssystemen untersucht, warum so viele Nutzer plötzlich orientierungslos in den Systemen „hängenbleiben". Eine (auch auf bestehende Systeme aufzusetzende) generelle Informationsstruktur soll formuliert und umgesetzt werden, z.B. soll durch ein graphisch ausgelegtes Leit- und Orientierungssystem eine Benutzerführung realisiert werden.

Im Bereich 3 werden die Interface- und Informationspräsentationseigenschaften festgelegt, die in den Untersuchungen variiert und experimentell geprüft werden sollen. Ein Beispiel für eine Eigenschaft der Oberflächengestaltung ist die Menge an Information, die gleichzeitig dargeboten wird; ein weiteres der logische Zusammenhang der dargestellten Inhalte bezogen auf ein bestimmtes Informationsgebiet, sowie der Gebrauch von interkommunikativen Möglichkeiten.

4 Internet Benutzerschnittstelle

Die Art der Interaktion zwischen Internet-Benutzer und Programm ist von entscheidender Bedeutung für die Akzeptanz Internetbasierter Applets. Die dafür erforderliche Mensch-Maschine-Schnittstelle kann einfacher und effizienter organisiert werden, wenn die Informationen visualisiert und damit der direkten Manipulation zugänglich gemacht werden kann, d.h. der Internet-Benutzer soll möglich mit den zur Verfügung stehenden Objekten auch arbeiten können. Zentrales Element der Benutzerschnittstelle ist damit der Dialog, und damit die Dialog- und Maskengestaltung z.B. im Rahmen eines Fensterkonzeptes. Ziel einer guten Organisation des Fensterkonzeptes ist es, dem Benutzer die passenden Informationen anzubieten mit einem minimalen Bedienungsaufwand zur Fenstersteuerung und dabei immer die Übersichtlichkeit zu bewahren, um die Augen- und Kopfbewegungen gering zu halten.

Ausgehend von einer Taxonomie der Handlungsfehler bei rechnergestützten Verfahren wurde im vorgestellten Projekt einer Benutzerschnittstelle für internetbasierte Simulation mit dreidimensionaler Navigation für einen WWW-Server konzipiert und implementiert. Aus den dabei gewonnenen Erfahrungen wird berichtet und die erzielten Ergebnisse präsentiert.