Fertigungsnahe Entwurfsunterstützung für die Mikrosystemtechnik

Vom Fachbereich Elektrotechnik und Informatik der Universität Siegen

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von

Dipl.-Ing. Andreas Wagener

1. Gutachter: Prof. Dr. Rainer Brück
2. Gutachter: Prof. Dr. Dietmar Fey
Vorsitzender: Prof. Dr. Madjid Fathi-Torbaghan

Tag der mündlichen Prüfung: 10.02.2005

urn:nbn:de:hbz:467-994

„Why is it so easy to acquire the solutions of past problems
and so difficult to solve current ones?“

(Herbert Marshall McLuhan 1911 - 1980)

Abstract

In microsystem technology, especially in the area of silicon-based non-electronic devices, the design is dominated by the fabrication of the devices. That is not only true for the technology-driven and physical design phases but also for the more abstract phases in the beginning of the design. In contrast to micro electronics there is no Mead/Conway-abstraction possible. That is why available design frameworks are limited to the behavioural-driven and more abstract design.

A new concept of an integrated design support for microsystem technology and a prototypically realisation of that concept is presented. Designing micro-systems with this tool the designer need not longer to be technology expert.

Basing on design methodology in micro electronics and mechanics as well considerations on the design in microsystem technology are discussed. A new methodology for technology-driven design is presented and by the means of a prototype realised. This prototype allows the management of process data as well as the design of fabrication process flows. Also the manufacturability of these flows can be checked. The concept of hierarchical ordered simulation models is introduced to guarantee at least a rough simulation of process flows, even if there is no model for one or more fabrication steps available.

Vorwort

In dieser Arbeit wird aus Gründen der besseren Lesbarkeit darauf verzichtet, Bezeichnungen für Personen, die sowohl weiblich als auch männlich sein können, jeweils explizit aufzuführen. In der Regel werden die männlichen Bezeichner verwendet. Gemeint sind allerdings jeweils beide Geschlechter.

Englische Quellen sind in dieser Arbeit weitgehend ins Deutsche übersetzt worden. Jedoch lassen sich einige Sachverhalte im Englischen prägnanter formulieren, so dass insbesondere bei Zitaten darauf verzichtet wurde, sie zu übersetzen.

Inhaltsverzeichnis

1 Einleitung

Die Mikroelektronik, deren Geburtsstunde im Jahr 1948 durch Bardeen, Brattain und Shockley bereitet wurde, hat seit Beginn der 1980er Jahre besonders im digitalen Bereich eine enorme Entwicklung im Bereich der Entwurfsunterstützung durchgemacht. Heutzutage ist der Entwurf von digitalen Schaltungen größtenteils automatisiert und die Fertigung stellt zumindest für Standardanwendungen keine großen Anforderungen an die Entwickler.

Das erste elektromechanische Mikrosystem wurde 1967 entwickelt [Na67]. Seit dem sind nun über drei Jahrzehnte vergangen. Trotzdem ist die Entwurfsunterstützung in der Mikrosystemtechnik bei weitem nicht so weit wie die der Mikroelektronik. Während Digitalschaltungen im Standardverfahren entworfen und gefertigt werden, stellt nahezu jeder Entwurf eines Mikrosystems die Entwickler von Neuem auf die Probe.

Laut [SGCD02] dauert die Entwicklung eines Mikrosystems je nach Anwendungsgebiet zwischen vier und zehn Jahren von der Konzeptphase bis zur Markteinführung. Bei solchen Zeiträumen sind nicht nur Marktprognosen schwierig. Auch die Entwicklungskosten sind für so lang anhaltende Projekte sehr hoch. Schon 1995 kostete die Entwicklung eines Mikrosystems durchschnittlich mehr als eine Million US Dollar [Ant96a].

Solche Zahlen legen es nahe, über eine sinnvolle Entwurfsunterstützung für die Mikrosystemtechnik analog der in der Mikroelektronik vorhandenen nachzudenken. Anders als in der Mikroelektronik hat man es aber bei der Mikrosystemtechnik mit einer Multidomänen-Disziplin zu tun, bei der die Fertigung nicht so einfach zu abstrahieren ist, wie eben in der Mikroelektronik. Bereits 1995 wurden Stimmen in der Fachwelt laut, die beklagten, dass im Gegensatz zur Mikroelektronik, wo die Fertigungsabstraktion bereits vor über 20 Jahren eingeführt wurde, für den Entwurf von Mikrosystemen ein viel zu großes Spezialwissen über die Fertigung notwendig ist [Ant96a]. Bis heute hat sich dar-

an nichts geändert. Je mehr man sich im Entwurf den weniger abstrakten, also technologienahen Entwurfsphasen nähert, desto mehr Fertigungswissen ist von Nöten. Für die Fertigungsplanung selbst müssen dann explizit Prozessschritte ausgewählt und parametrisiert werden. Auf dem Markt sind für die Unterstützung der technologienahen Entwurfsphasen keine durchgängigen Softwarewerkzeuge erhältlich. Lediglich Insellösungen, die sich ganz spezieller Problemstellungen annehmen können erworben werden.

In dieser Arbeit wird ein Konzept zur Unterstützung des fertigungsnahen Mikrosystementwurfs vorgestellt, das dazu führt, dass für den Entwurf weniger Spezialwissen über die Fertigung notwendig ist. Realisiert wird das durch eine gezielte Sammlung und Aufbereitung von Fertigungsdaten, die dann beim Entwurf zur Verfügung stehen. Dabei steht im Vordergrund, dass die Entwicklung von Mikrosystemen auch von „Nicht-Experten“ durchgeführt werden kann. Teile der hier vorgestellten Ergebnisse sind bereits auf internationalen Konferenzen und Tagungen [PSWH05, SWPH04, WPHB04, PWOB04, WPH04, HWPB03, HW03, WPH03, WH03, WPHB02a, WPHB02b] vorgestellt worden.

Im folgenden Kapitel wird ein Überblick über die Mikrosystemtechnik und die Fertigungsprozesse gegeben. Anschließend wird in Kapitel 3 auf grundlegende Aspekte des Entwurfs und deren Umsetzung in der Mikroelektronik und Mechanik eingegangen. Motiviert werden die Betrachtungen durch die Nähe der Mikroelektronik und Mechanik zur Mikrosystemtechnik.

In Kapitel 4 werden dann Untersuchungen zur Entwurfsmethodik in der Mikrosystemtechnik angestellt, die mit der Vorstellung einer Entwurfsmethodik für den fertigungsnahen Entwurf schließen. In den beiden darauf folgenden Kapiteln werden die Anforderungen für die Umsetzung der Methodik und mögliche Lösungen präsentiert.

Kapitel 7 stellt eine innovative Softwarearchitektur vor, die besonders geeignet ist, die in den beiden vorherigen Kapiteln geforderten Eigenschaften einer Entwurfssoftware zu gewährleisten.

In Kapitel 8 wird dann die Entwicklungsumgebung PRINCE vorgestellt, die im Rahmen dieser Arbeit entwickelt und implementiert wurde. PRINCE erfüllt die Anforderungen, die durch die fertigungsnahe Entwurfsmethodik definiert werden.

Abschließend fasst Kapitel 9 alle in dieser Arbeit vorgestellten Erkenntnisse noch einmal zusammen und gibt einen Ausblick auf weitere Arbeiten, die in diesem Kontext erfolgen oder erfolgen könnten.

2 Mikrosystemtechnik

Die Mikrosystemtechnik gilt als Schlüsseltechnologie für dieses Jahrhundert. Sie beschäftigt sich mit dem Entwurf, der Fertigung und der Anwendung von sogenannten Mikrosystemen[1]. Synonym wird der Begriff auch für die Fertigung selbst, der Mikrotechnik, verwendet. Wie der Name schon vermuten lässt, beschäftigt sich die Mikrosystemtechnik mit Systemen, deren Abmessungen typischerweise im Mikrometerbereich liegen. Charakteristisch für Mikrosysteme ist der oft dreigeteilte Aufbau. Sensorik, Aktorik und Auswerteelektronik sind in der Regel Bestandteile eines Mikrosystems.

Die Sensorik nimmt Signale der Umwelt auf. Die Auswerteelektronik wertet diese Signale aus und entscheidet, wie die Aktorik darauf reagieren soll. Damit wird eine weitere Eigenschaft eines Mikrosystems eingeführt, die auch durch den Begriff des *Systems* ausgedrückt wird. Mikrosysteme bestehen aus mehreren Bausteinen, die unterschiedlicher Natur sein können. Die Sensorik beispielsweise kann sowohl elektrische Phänomene detektieren oder auch auf mechanische Krafteinwirkung ansprechen. Das gleiche gilt auch für die Aktorik. Lediglich die Auswertung geschieht in der Regel über eine elektronische Schaltung. Neben den schon erwähnten Bereichen der Elektronik und der Mechanik können auch noch weitere Domänen involviert sein. Thermik, Chemie, Akkustik, Fluidik und Biologie sind nur ein paar Beispiele.

Die Geburtsstunde der Mikrosystemtechnik kann auf das Jahr 1948 datiert werden. Bardeen, Brattain und Shockley stellten in diesem Jahr den Feldeffekttransistor vor, der die Ära der Mikroelektronik einläutete. Da die Mikroelektronik essentieller Bestandteil der Mikrosystemtechnik ist, kann dieses Datum auch eben als Geburtsstunde derselben angesehen. 1954 wurde es dann

[1] Im amerikanischen Sprachraum spricht man hauptsächlich von MEMS. MEMS steht für *Micro Electrical Mechanical System* und enthält bereits im Namen die beiden am häufigsten auftretenden Domänen in der Mikrosystemtechnik.

möglich durch die Entdeckung des piezoresistiven Effekts, Halbleiter auch außerhalb der Mikroelektronik einzusetzen [GD97]. Das allererste Mikrosystem im eigentlichen Sinne stellten jedoch Nathanson et. al. 1967 vor [Na67]. Beim *Resonant Gate Transistor* kombinierten sie einen Transistor mit einer einfachen mechanischen Konstruktion (siehe Abbildung 2.1). Über einen beweglicher Goldcantilever konnte der Drainstrom des Transistors gesteuert werden.

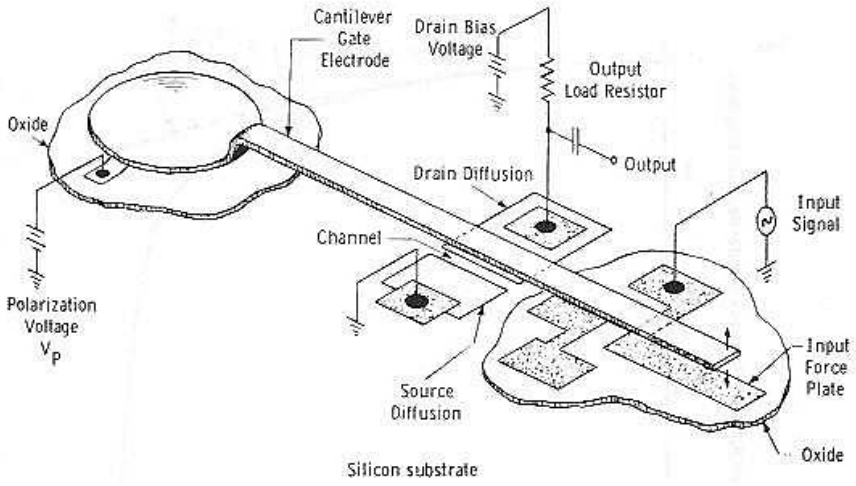


Abbildung 2.1: Der Resonant Gate Transistor [Na67]

Heute sind Mikrosysteme in fast allen Bereichen des Lebens zu finden. Von den Düsen der Tintenstrahldrucker, über Sensoren und Aktoren in modernen Kraftfahrzeugen bis hin zu medizinischen Anwendungen. Mit der Portfolioerweiterung geht auch eine wirtschaftliche Ausdehnung einher. Während im Jahr 2000 das Marktvolumen der Mikrosysteme ca. 30 Milliarden Dollar weltweit ausmachte, sind für das Jahr 2005 mehr als das Doppelte prognostiziert (siehe Abbildung 2.2).

2.1 Fertigung

Mikrosysteme können auf vielfältige Art und Weise hergestellt werden. Viele Faktoren beeinflussen die Wahl des Herstellungsprozesses. Zum einen spielt sicherlich die Strukturgröße des Mikrosystems eine wichtige Rolle. Ein hochintegriertes System mit mechanischen Komponenten im Mikrometer-Bereich

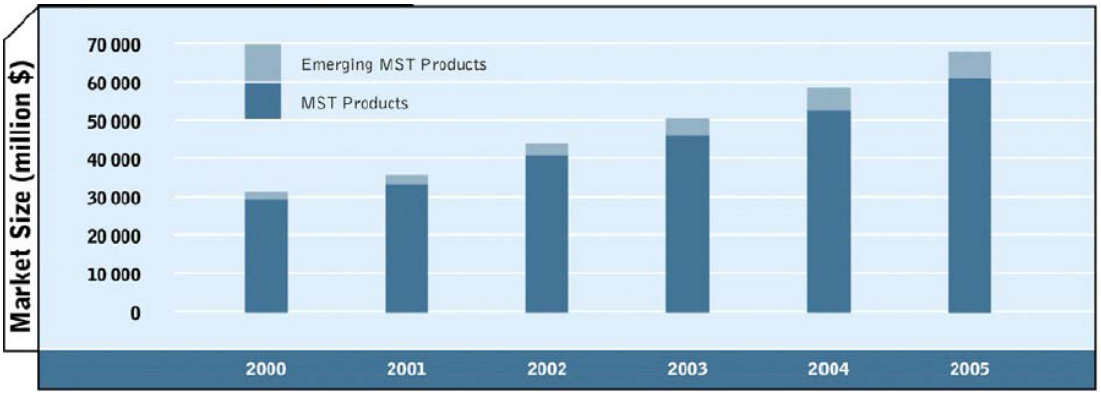


Abbildung 2.2: Marktentwicklung von Mikrosystemen [NEX02]

lässt sich nicht mehr mit Feinwerktechniken herstellen. Beeinflussend auf die Herstellung wirkt sich auch das zukünftige Einsatzgebiet aus. Die wohl dominanteste Rolle jedoch nimmt der Fertigungsprozess an sich bei der Diskussion der Herstellung ein. Bestimmte Strukturen lassen sich teilweise nur mit sehr wenig Spielraum bei der Prozessierung realisieren. Durchgesetzt haben sich in der Mikrosystemtechnik zwei Verfahrensfamilien. Auf der einen Seite steht die Siliziumtechnik. Die Siliziumtechnik in der Mikrosystemtechnik besteht in der Regel aus nativen oder modifizierten Prozessen der Mikroelektronik und ist gerade deshalb sicherlich die verbreitetste Technologiefamilie. Sie bietet sich auch an, wenn man Hybridsysteme bestehend aus Sensoren oder Aktoren und Auswerteelektronik aufbauen möchte. Unter Siliziumtechnik sollen hier auch die abgeleiteten Verfahren mit anderen Halbleitern wie Silizium-Germanium (SiGe) verstanden werden.

Auf der anderen Seite ist hier die LIGA-Technik zu erwähnen. Sie zeichnet sich durch eine große Materialvielfalt und sehr hohe Aspektverhältnisse aus. Das heißt, dass mittels der LIGA-Technik extrem schmale und hohe Strukturen fertigbar sind. Im Gegensatz zur Siliziumtechnik lassen sich Hybridsysteme in der LIGA-Technik jedoch sehr schwer und schon gar nicht auf einem Chip realisieren [Hah98a].

Neben diesen beiden wohl bekanntesten Technologien zur Erzeugung von Mikrosystemen [BRS01] gibt es noch die Feinwerk- und Lasertechnologie. Die Feinwerktechnologie hat ihren Ursprung im Uhrmacherhandwerk und der Kameraherstellung. Eine detaillierte Beschreibung dieser beiden Verfahren ist

in [BRS01] zu finden. Da sie für die hiesigen Betrachtungen wenig Relevanz haben, sei für weitere Details auf die angegebene Literatur verwiesen.

Im Folgenden sollen nun einige Fertigungsverfahren kurz vorgestellt werden. Sie stehen stellvertretend für eine Vielzahl an Fertigungsmöglichkeiten. Eine Aufzählung und Erläuterung aller würde den Rahmen dieser Arbeit bei weitem sprengen und ist für das weitere Verständnis auch nicht nötig. Für eine umfassende Darstellung sei primär auf [Mad02, Sen01] verwiesen. Weiterhin können auch [GD97, FR97] nützlich sein. Im Abschnitt 2.2 werden die Verfahren der Siliziumtechnik vorgestellt. Im Abschnitt 2.3 soll dann ein kunststoffbasiertes Verfahren vorgestellt werden.

2.2 Siliziumtechnik

Die Siliziumtechnik ist wohl die älteste und auch am weitesten verbreitete Technologie in der Mikrosystemtechnik. Silizium vereinigt einige gute Eigenschaften, die wohl sonst nur sehr wenige Materialien inne haben. Zusätzlich ist Silizium seit mehreren Jahrzehnten Grundstoff der Halbleiterindustrie und somit bestens bekannt und verstanden. Anders als der Name vermuten lässt, wird in der Siliziumtechnik nicht nur mit Silizium gearbeitet. Jedoch ist der Wafer, also die Basis des Systems aus Silizium. Heutzutage werden auch andere Materialien als Grundstock verwendet. Zu erwähnen wären beispielsweise Gallium-Arsenid (GaAs) und Silizium-Germanium-Verbindungen (SiGe). Die Grundprinzipien der Fertigung ähneln sich jedoch in der Regel.

Die einzelnen Fertigungsschritte lassen sich auf verschiedene Weisen klassifizieren. Klassisch ist die Aufteilung nach Oberflächentechnologien (engl. *surface micromachining*) und Substrat- oder Bulktechnologien. Zu den Oberflächentechnologien werden dann solche Verfahren zusammengefasst, die in erster Linie darauf abzielen, Schichten auf den Wafer abzuscheiden, zu verändern und zu strukturieren. Die Bulktechnologien beziehen sich dann auf den Wafer selbst, gehen also mehr in die Tiefe. Da jedoch einige Verfahren sowohl in den Oberflächentechnologien als auch in den Bulktechnologien Verwendung finden, soll hier eine Klassifizierung an Hand der Technologieprinzipien erfolgen. Das geschieht natürlich auch mit dem Hintergedanken, dass

in der später in Kapitel 8 vorgestellten neu entwickelten Software genau diese Art der Prozessklassifizierung umgesetzt ist.

2.2.1 Abscheiden und Aufbauen

CVD

CVD-Verfahren (engl. *Chemical Vapor Deposition*, chemische Dampfabscheidung) sind Verfahren, bei denen das Abscheidematerial in einem Trägergas gelöst auf dem Substrat abgeschieden wird. In einem Reaktor umströmt das Gasgemisch das Substrat bei einer Temperatur von 400° bis 1250° C. Es zerfällt in niedermolekulare Bestandteile und reagiert mit dem Substrat. Das Trägergas dient zur Verdünnung der Konzentration, da sich bei zu hohen Konzentrationen bereits im Gas Molekülketten bilden würden. Dies ist zumindest bei monokristallinen Strukturen nicht erwünscht. Die CVD-Verfahren lassen sich noch weiter unterteilen. Man unterscheidet nach Verfahren bei Umgebungsdruck (APCVD), Unterdruck (LPCVD) und plasmaunterstützten Verfahren (PECVD). Das plasmaunterstützte Verfahren findet dort Anwendung, wo mit relativ geringen Temperaturen gearbeitet werden muss [GD97].

Epitaxie

Soll die monokristalline Schicht des Substrats durch die abgeschiedene neue Schicht fortgeführt werden, kann die Epitaxie verwendet werden. In der Regel werden bei der Epitaxie CVD-Verfahren (s. o.) verwendet. Bei den Epitaxie-Verfahren unterscheidet man nach Homo- und Hetero-Epitaxie. Die Homo-Epitaxie dient der Abscheidung eines dem Substrat gleichen Materials, z. B. Silizium auf Silizium. Die Hetero-Epitaxie widmet sich der Abscheidung von Substrat-fremden Materialien. Beide Verfahren erzeugen aber eine monokristalline Schicht, die die Kristallstruktur des Substrats fortführt. Das setzt voraus, dass die Gitterstruktur des abgeschiedenen Materials dem des Substrats ähnelt. Andernfalls kann auf Grund intrinsischer Spannungen kein Einkristallfilm entstehen [MB93]. Eine Dotierung der neuen monokristallinen Schicht ist ebenfalls möglich durch Beimischung von entsprechenden Gasen.

PVD

Zu der Gruppe der PVD-Verfahren (PVD für engl. *Physical Vapor Deposition*) zählen alle Prozesse, die durch physikalische Verfahren das aufzubringende Material in die Gasphase versetzen und anschließend auf dem Substrat kondensieren lassen. Die PVD-Verfahren eignen sich besonders für die Herstellung leitfähiger Schichten. Dazu stehen verschiedene Verfahren zur Verfügung. Beim Aufdampfen wird mittels Hitze das Schichtmaterial verdampft. Es scheidet sich dann auf dem Substrat wieder ab. Dieses Verfahren kann mit Hilfe eines Elektronenstrahls erweitert und präzisiert werden. Das Material wird durch einen steuerbaren Elektronenstrahl aufgeheizt und scheidet sich wieder auf dem Substrat ab. Da der Elektronenstrahl durch Magneten abgelenkt werden kann, steht es frei, mehrere Materialproben anzusteuern. Dadurch erhält man Legierungen als Abscheidematerial mit exakt vorhersagbaren Mischungsverhältnissen [MB93].

Das wohl bedeutendste physikalische Verfahren (nach [GD97]) ist das Sputtern. Sputtern steht hier für Zerstäuben und wird insbesondere dort eingesetzt, wo es auf eine gute Schichthaftung ankommt. Beim Sputtern wird das neue Schichtmaterial als so genanntes Target verwendet. Zwischen dem Substrat und dem Target wird ein Plasma entzündet, dessen Ionen Atome bzw. Moleküle aus dem Target schlagen. Diese werden dann in Richtung Substrat beschleunigt und bleiben auf dem Substrat haften. Mit diesem Verfahren sind nur geringe Aufwachsraten zu erreichen. Verbessern lässt sich das durch eine Überlagerung eines Magnetfeldes. Man spricht dann auch von Magnetron-Sputtern. Auf diese Weise lassen sich dann Abscheideraten von einem Mikrometer pro Minute erreichen [GD97].

Spin-On-Verfahren

Die Spin-On-Verfahren (*Spin-On*, engl. für Aufschleudern) werden hauptsächlich für das Aufbringen von Fotolacken benutzt. Diese können dann lithografisch strukturiert werden. In die Mitte des Wafers wird das flüssige Material aufgetropft und durch gleichmäßige Rotation des Wafers durch die Zentrifugalkraft über denselben verteilt. Es bildet sich eine mikrometerdünne gleichmäßige Schicht auf dem gesamten Wafer. Durch ein anschließendes Trocknen

(*Softbake*) werden die Lösungsmittel herausgetrieben [Hah98a]. Zu beachten ist, dass in der Regel nicht das gesamte Lösungsmittel ausgetrieben wird. Ein kleiner Rest bleibt oft noch enthalten. Dieser kann später zu Problemen führen, wenn andere Prozessschritte unter erhöhter Temperatur gefahren werden.

2.2.2 Verändern

Dotieren

Die Dotierung (engl. *doping*) von Halbleitermaterialien dient der gezielten Einbringung von Störstellen in die Gitterstruktur des Kristalls. Als Materialien für die Dotierung von Silizium dienen beispielsweise Bor, Phosphor und Arsen. Sie erzeugen durch fehlende bzw. überschüssige Elektronen eine erhöhte Elektronenbeweglichkeit im dotierten Silizium. Eine Dotierung ist auf mehrfache Weise möglich. Einerseits kann man direkt bei der Abscheidung einer Schicht dafür sorgen, dass das abgeschiedene Material dotiert ist. Dies ist beispielsweise über die Beimischung von Gasen möglich.

Auf der anderen Seite gibt es noch die Möglichkeit der Ionenimplantation und Diffusion. Zuerst wird das Material in geeigneter Konzentration oberflächennah auf den Wafers mittels der Ionenimplantation eingebracht. Dabei werden die Teilchen so weit beschleunigt, dass sie in Form eines Ionenstrahls auf den Wafer treffen. Dort lagern sie sich nahe der Oberfläche an. Die Feinjustierung des dotierten Bereichs geschieht dann über die Diffusion. Dabei wird der Wafer erhitzt, was bewirkt, dass die Dotierung wegen des Konzentrationsgefälles weiter in den Wafer wandert. Gleichzeitig heilen Gitterfehler im Kristall, die durch den Ionenbeschuss entstanden sind, wieder aus. Auf diese Weise lässt sich ein sehr genaues Dotierprofil fast beliebig im Wafer positionieren.

Oxidieren

Oxidieren kann man auf zwei Weisen. Einerseits kann man thermisch trocken oxidieren. Dabei wird der Wafer in eine Sauerstoffatmosphäre bei 850° bis 1150° C gebracht [Sen01]. Bei diesem Verfahren reagiert der Sauerstoff mit dem

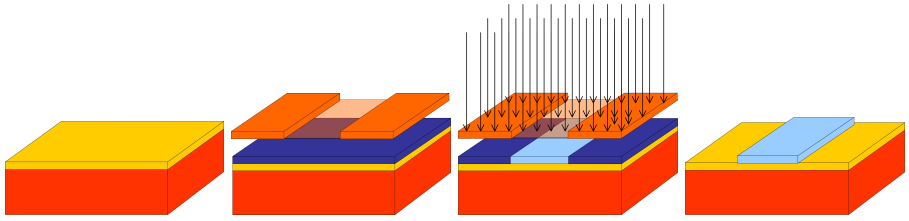
Silizium zu Siliziumoxid. Das Oxid wächst bis zu 40% in die Siliziumschicht hinein. Zusätzlich wächst es auch auf. Wegen der hohen Temperaturen können bereits vorhandene Schichten beschädigt oder zerstört werden [Hah98a]. Die Aufwachsrate nimmt mit steigender Oxiddicke immer weiter ab, da die Sauerstoffatome durch eine immer dicker werdende Oxidschicht wandern müssen, um das Silizium zu oxidieren. Deshalb lassen sich nackte Siliziumwafer leichter und schneller oxidieren als Wafer, die bereits anoxidiert sind [Sen01].

Nassoxidieren wird dann angewendet, wenn die Oxidschicht schnell erstellt und nicht besonders hochwertig sein muss, beispielsweise bei Opferschichten. Zwar ist das Oxid nicht so hochwertig wie ein Trockenoxid, dafür kann man aber leicht Schichtdicken bis zu 1,5 μm erreichen.

Lithografie

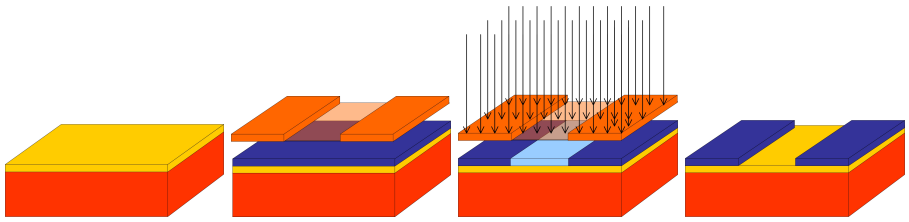
Lithografie bedeutet wörtlich übersetzt soviel wie Steinzeichnung und kommt aus dem Altgriechischen ($\lambda\iota\vartheta o\varsigma$ = der Stein, Fels und $\gamma\rho\alpha\varphi\varepsilon\iota\nu$ = schreiben, zeichnen, malen). Im heutigen Sprachgebrauch jedoch wird der Begriff Lithografie für den Vorgang des Belichtens benutzt. Er beschreibt den Vorgang, bei dem Strahlung für die Übertragung von zweidimensionalen Strukturen auf einer sogenannten Maske in eine dreidimensionale Schicht aus strahlungsempfindlichen Material (Fotolack oder Resist) genutzt wird. Man kann Lithografie auf unterschiedlichste Arten betreiben. Zum einen kann man die Strahlungsquelle von sichtbarem Licht bis hin zur Röntgenstrahlung (siehe auch Abschnitt 2.3) variieren. Das bewirkt, dass die abbildbaren Strukturen kleiner werden, hat aber auch eine enormen Anstieg der Strahlungsenergie zur Folge. Zum anderen kann man die Position der Maske zwischen Strahlungsquelle und Resist verändern, worauf hier aber nicht näher eingegangen werden soll. In der Siliziumtechnik hat sich die Benutzung von sichtbarem und ultraviolettem Licht in Kombination mit lichtempfindlichen Fotolacken etabliert. Bei den Fotolacken unterscheidet man nach Negativ- und Positivlack. Beim Negativlack verändert sich das bestrahlte Material so, dass es beim anschließenden Entwickeln stehen bleibt (siehe Abbildung 2.3). Beim Positivlack wird dagegen das bestrahlte Material entfernt (siehe Abbildung 2.4).

Die Lithografie besteht in der Regel aus mehreren Einzelschritten. Zuerst wird der Lack aufgeschleudert. Dann wird er mittels eines Prebakes etwas gehärtet.



(a) (b) (c) (d)

Abbildung 2.3: Lithografie mit Negativlack



(a) (b) (c) (d)

Abbildung 2.4: Lithografie mit Positivlack

Es folgt die soeben beschriebene Lithografie. Anschließend wird ein Postbake zur endgültigen Festigung des Lacks durchgeführt. Mit einem speziellen Entwickler können dann die nicht benötigten Strukturen entfernt werden. Da der Lack in der Regel resistent gegen diverse Ätzverfahren ist, kann er als Maske für die Strukturierung der darunter liegenden Schichten verwendet werden.

2.2.3 Entfernen

Ätzen

Beim Ätzen unterscheidet man Trocken- und Nassätzverfahren. Die Trockenätzverfahren werden alle bei sehr geringem Druck prozessiert. Das gewährleistet eine hohe Ionenbeweglichkeit. Sie ist für ein gutes Ätzresultat verantwortlich. Beim Trockenätzen wird ionisiertes Gas verwendet. Bei den chemischen Verfahren werden mittels Plasma stark aktive Radikale erzeugt, die die Oberfläche des Materials ablösen. Dies geschieht durch das Eingehen von Bindungen zwischen den Radikalen und den Materialatomen. Die physikalischen Verfahren beschleunigen die Ionen. Sie prallen dann mit großer kinetischer Energie auf die Oberfläche.

Bei den nasschemischen Verfahren kommt der Wafer in Kontakt mit einer Ätzlösung. Dies kann durch Besprühen oder durch Eintauchen geschehen. Sie sind in der Regel kostengünstiger als die Trockenätzverfahren, da die benötigten Apparaturen weniger aufwendig sind [Hah98a]. Man unterscheidet nach isotropen und anisotropen Verfahren. Erstere ätzen gleichmäßig in alle Richtungen, während die anisotropen Ätzlösungen bevorzugt in eine Richtung ätzen. Anisotrope Ätzverfahren haben sich in der Mikromechanik zu einer Schlüsseltechnologie entwickelt [MB93,GD97]. Bei diesen Verfahren macht man sich den Effekt zu nutze, dass spezielle Ätzmittel entlang der Hauptkristallebenen das Silizium unterschiedlich schnell abtragen. Auf diese Weise lassen sich senkrechte Wände oder V-förmige Gräben realisieren.

Lift-Off Technik

Lift-Off Techniken werden beispielsweise dort eingesetzt, wo man eine Strukturierung durch Plasma-Ätzen nicht erreichen kann. Das ist z. B. bei Metallen

der Fall [Sen01]. Beim Lift-Off wird eine Lackschicht auf den Wafer aufge-

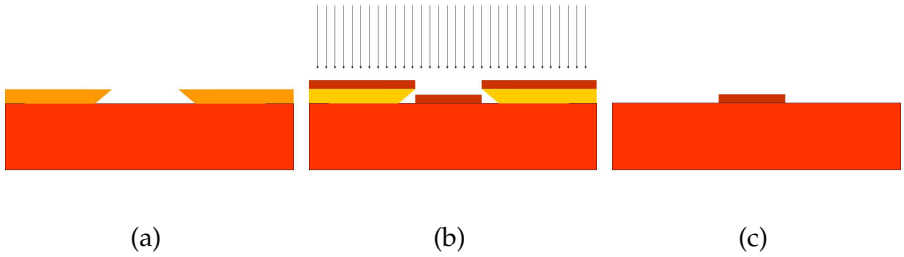


Abbildung 2.5: Lift-Off Technik am Beispiel

bracht, belichtet und strukturiert. Dabei ist darauf zu achten, dass die Lackkanten negativ sind, damit das Metall, das anschließend abgeschieden wird, nicht auch die Kanten des Lacks bedeckt. Das ist wichtig, da sonst das Metall eine durchgängige Schicht bildet, die keinen Entwickler zum Lack durchlässt. Das Metall wird über die gesamte Fläche abgeschieden. Das Metall ist also sowohl auf dem Lack als auch auf den entwickelten Bereichen zu finden. Nun entfernt man die Lackschicht, was zur Folge hat, dass auch die Metallschicht, die auf dem Lack liegt, einfach abfällt. Auf diese Weise hat man eine strukturierte Metallschicht geschaffen (siehe auch Abbildung 2.5).

Reinigung

Vor einigen Prozessschritten muss die Waferoberfläche gereinigt werden. Das ist beispielsweise notwendig, wenn Hochtemperatur-Prozessschritte, wie Oxidation, CVD und Diffusionen durchgeführt werden sollen [Sen01]. Die gängige Art zu reinigen ist die *RCA-Reinigung*[2]. Diese Reinigung besteht aus mehreren verschiedenen Schritten. Man beginnt in der Regel mit dem Entfernen von organischen Materialien. Dafür verwendet man eine starke Säure, wie beispielsweise sieben Teile Phosphorsäure (H_3PO_4) auf drei Teile Wasserstoffperoxid (H_2O_2). Dieser Einzelschritt ist auch besser unter dem Namen *Piranha-Cleaning* bekannt. Übrig gebliebene organische Stoffe werden dann mit einer 5:1:1 Mischung aus Wasser, Wasserstoffperoxid und Ammoniak (NH_3) beseitigt. Weil dabei auf Silizium eine dünne Oxidschicht aufwächst, wird bei

[2]RCA steht hier für das Unternehmen Radio Corporation of America. Dort entwickelte Werner Kern 1965 dieses Standardverfahren zur Reinigung von kontaminierten Wafern.

nackten Siliziumwafern mit einer verdünnten Flusssäure (HF) das Oxid wieder entfernt. Zum Schluss werden dann mit einer 6:1:1 Mischung aus Wasser, Salzsäure und Wasserstoffperoxid Ionenverunreinigungen abgebaut.

2.3 LIGA

Das LIGA-Verfahren wurde Anfang der achtziger Jahre des letzten Jahrhunderts am damaligen Kernforschungszentrum Karlsruhe (jetzt Forschungszentrum Karlsruhe) entwickelt. Ursprünglich war es für die Herstellung von Diffusionsdüsen für die Anreicherung von Uran gedacht. Doch schnell erkannte man, dass es sich auch für die Herstellung anderer kleiner Produkte eignet [FR97]. Der Name LIGA setzt sich zusammen aus den Wörtern Lithografie, Galvanoformung und Abformung, die auch gleichzeitig die drei Hauptschritte der LIGA-Fertigung beschreiben. Mit dem LIGA-Verfahren lassen sich sowohl Produkte aus Kunststoff als auch aus Keramiken und Metall fertigen. Vorteile der LIGA-Technik gegenüber der Siliziumtechnik sind zum einen das sehr große Aspektverhältnis von 1000:1 und die sehr günstige Massenproduktion. Allerdings ist die Herstellung der Form sehr teuer, da die Lithografie mittels Synchrotronstrahlung erfolgen muss. Auch ist der LIGA-Prozess nicht CMOS-kompatibel, so dass Elektronik immer extern realisiert werden muss. Abbildung 2.6 gibt einen Überblick über die einzelnen Schritte des Verfahrens. Die folgende Beschreibung des LIGA-Verfahrens basiert auf [MM02, FR97].

2.3.1 Maskenerstellung

Beim LIGA-Verfahren wird, wie schon erwähnt, Synchrotronstrahlung zur Belichtung eingesetzt. Bei der Synchrotronstrahlung handelt es sich um sehr harte und energiereiche, aber auch parallele Röntgenstrahlung. Sie muss sehr aufwendig in Synchrotronringen erzeugt werden, weshalb die Herstellung des Masters (s.u.) für die Abformung sehr teuer ist. Wegen der Härte der Strahlung müssen die Materialien für die Maske spezielle Eigenschaften haben. Man verwendet als Absorber in der Regel Gold oder Tantal. Beide Elemente schirmen Röntgenstrahlung ausreichend ab. Als transparenten Träger wird Titanium bzw. Beryllium eingesetzt. Auf den Träger wird eine Schicht aus Poly-

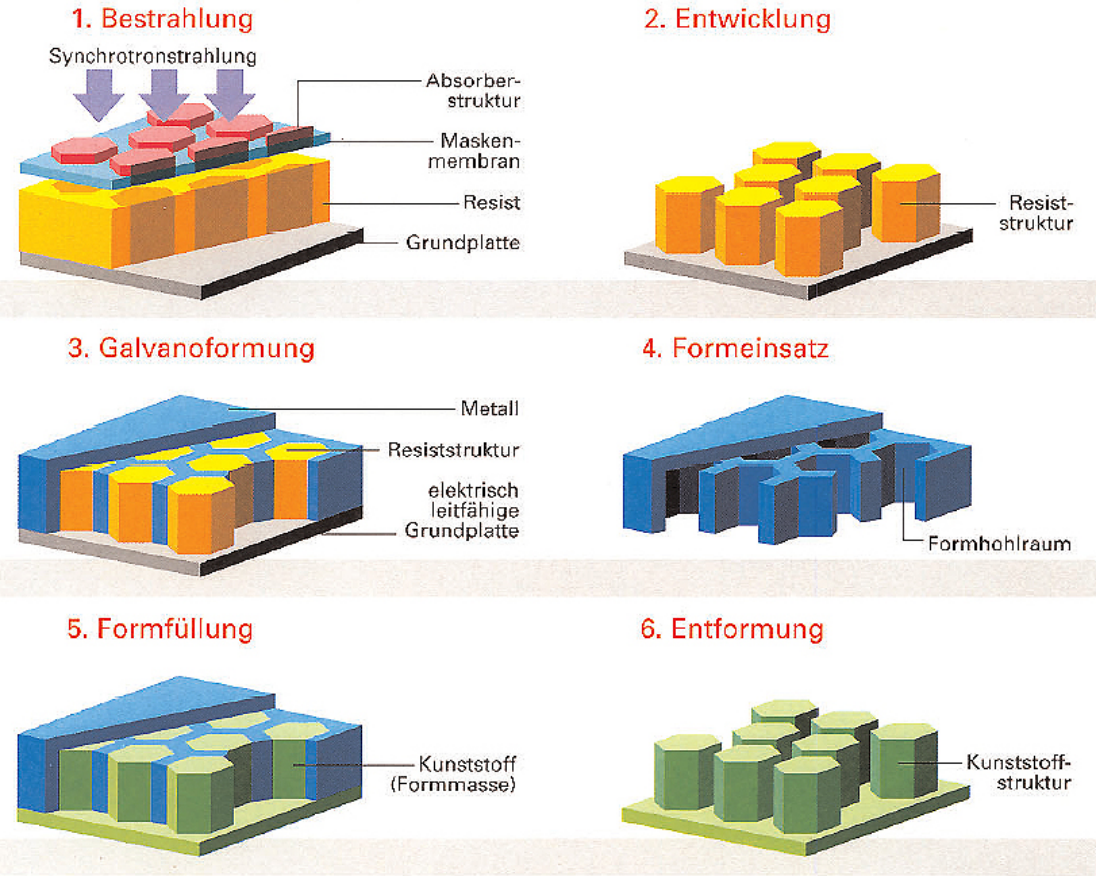


Abbildung 2.6: Prozessfolge beim LIGA-Verfahren [Ins04b]

methylmethacrylat (PMMA), das besser unter dem Namen Plexiglas bekannt ist, aufgebracht. Mittels der Röntgenstrahllithografie (s. u.) wird das PMMA strukturiert. Anschließend wächst man das Absorbermaterial galvanisch auf und entfernt das PMMA wieder. Somit wird die Maske analog der beiden ersten Schritte des LIGA-Verfahrens erzeugt.

2.3.2 Lithografie

Die Lithografie beim LIGA-Prozess ähnelt sehr der bereits in Abschnitt 2.2 beschriebenen Lithografie. Auf einen elektrisch leitfähigen Träger wird eine Resistschicht aufgebracht. Die Dicke der Schicht hängt direkt von der gewünschten Strukturhöhe ab. Der elektrisch leitfähige Untergrund ist für die Galvanoformung nötig (s. u.). Mittels der Maske und der Synchrotronstrahlung wird das Resist belichtet. Dies kann bei PMMA als Resist bis zu acht Stunden dauern [FR97]. Nach der Belichtung kann das Resist entwickelt werden. Übrig bleibt ein dreidimensionales Abbild der Maske.

2.3.3 Galvanoformung

Durch die Entwicklung des Resists gibt es nun Bereiche, in denen das leitfähige Trägersubstrat zum Vorschein kommt, und Bereiche, in denen es vom Resist bedeckt wird. In den nicht bedeckten Bereichen wächst man Metall galvanisch auf. Lässt man es über das Resist hinaus aufwachsen, so erhält man eine neue Form, den Master, die langlebiger als die Kunststoffform ist. Das Leitersubstrat wird entfernt und das Resist, das als Maskierung gedient hat, kann ebenfalls entfernt werden. Verhindert man das Überwachsen des Resists, kann man nach Ablösen des Leitersubstrats die Struktur bereits als Mikrostruktur nutzen. Will man jedoch die Struktur als Form verwenden, so muss sie noch abgeformt werden.

2.3.4 Abformung

Der Grund für die günstige Herstellung von Mikrostrukturen mittels des LIGA-Verfahrens ist die relativ einfache Kunststoffabformung. Mit der durch

die Galvanoformung erhaltenen Negativstruktur kann man nun beinahe beliebig viele neue Strukturen abformen. Das kann auf verschiedene Arten und Weisen gemacht werden. Zum Einen kann man den Master als Stempel verwenden und ihn in ein weicheres Material drücken. Auf diese Weise formt der Master ein Abbild von sich selbst in das Zielmaterial. Zum Anderen kann man aber auch die Hohlräume des Masters mit Spritz- oder Schlickerguss füllen und anschließend aushärten.

3 Entwurfsmethoden

Die Mikrosystemtechnik ist ein Schmelztiegel vieler unterschiedlicher Disziplinen. Wie bereits in Kapitel 2 beschrieben, bestehen Mikrosysteme nicht nur aus elektronischen und mechanischen Komponenten. In der Tat trifft man auch auf optische, thermische, fluidische und andere Systeme. Das bedeutet, dass man sowohl mit den klassischen Naturwissenschaften als auch mit den Ingenieurwissenschaften zu tun hat. Jede einzelne der involvierten Disziplinen hat ihre eigene Methodik für den Entwurf beziehungsweise für die Konstruktion einzelner Systemkomponenten. Die Methoden wurden über viele Jahre entwickelt und speziell auf die jeweilige Problemstellung angepasst. Alle diese Methoden basieren jedoch auf den gleichen Prinzipien, die nun vorgestellt werden.

Zu Beginn soll der Entwurf selbst betrachtet werden.

Entwurf: Ein Entwurf bezeichnet in der Regel die Schritte, die notwendig sind, um aus einer Spezifikation die relevanten Daten für die Fertigung zu generieren [Brü96]. Ein Entwurf besteht häufig aus mehreren Phasen, weshalb man auch allgemein von einem Entwurfsprozess spricht [Hah98a].

Nach [Ull02] kann man fünf Arten des Entwurfs unterscheiden.

Entwurf durch Auswahl (Selection Design): Beim Entwurf durch Auswahl selektiert der Ingenieur ein ähnliches Produkt aus einer Liste, Bibliothek oder einem Katalog aus. Das mag simpel klingen, allerdings kann man sich vorstellen, dass er mit steigender Auswahl vor der berühmten *Qual der Wahl* steht. Entscheidungsgrundlage ist die genaue Spezifikation. Anschließend vergleicht er das gewählte Produkt nochmals mit den gewünschten Eigenschaften. Man könnte auch vom *Entwurf durch Wiederverwendung* sprechen.

Entwurf durch Konfiguration (Configuration Design): Der Entwurf durch Konfiguration ist etwas komplexer als der durch Auswahl. Alle Komponenten für den Entwurf liegen vor. Die Aufgabe besteht darin, diese Komponenten richtig zusammenzusetzen. Ein Beispiel hierfür wäre ein Notebook, dass entsprechend der Vorgaben aufgebaut werden soll. Alle Komponenten stehen in Größe und Gewicht fest. Alleinig die ideale Anordnung auf engstem Raum unter Berücksichtigung von Anschlüssen und Wärmeabfuhr stellt das Problem dar. Natürlich können auch Konfigurationsentwürfe vorliegen, bei denen die Komponenten noch in verschiedene Ausprägungen vorkommen können. Bei dem Notebook-Beispiel könnte beispielsweise der Akku in verschiedenen Abmessungen erhältlich sein. Dadurch steigt der Freiheitsgrad und damit auch die Komplexität des Entwurfsproblems.

Entwurf durch Parametrisierung (Parametric Design): Beim Entwurf durch Parametrisierung geht es um die Findung geeigneter Parameter für eine Komponente. Braucht man beispielsweise ein Getriebe, so weiß man in der Regel ganz genau, wie es aufgebaut ist. Die Getriebeübersetzung lässt sich mit Hilfe von Formeln beschreiben. Allerdings gibt es selbst bei gleichbleibender Übersetzung eine Vielzahl an Parametern die geändert werden können, wie Anzahl der Zahnradzähne, Größe der Zahnräder etc. Bei einer gegebenen Übersetzung jedoch gibt es unendlich viele Kombinationen von Zahnrädern, die zu diesem Ergebnis führen. Die Aufgabe besteht darin, eine geeignete zu finden.

Neuentwurf (Original Design): Der Neuentwurf widmet sich dem gänzlich neuen Entwurf. Ein Neuentwurf findet dann statt, wenn keine bekannte Komponente existiert, die als Lösungsansatz dienen kann. Während die oben beschriebenen Entwurfsprobleme sich in irgend einer Art und Weise durch Regeln, Gleichungen oder andere Schemata beschreiben lassen, kann der Neuentwurf eben nicht so weit reduziert werden. Das macht das Neue aus. Im Grunde genommen sind die obigen Entwürfe Unterkategorien des Neuentwurfs.

Entwurf durch Überarbeitung (Redesign): Die meisten Entwürfe in der Industrie sind Redesigns [Ull02]. Das bedeutet, dass ein bestehender Entwurf überarbeitet wird, um neue Anforderungen zu erfüllen. Die

Überarbeitung an sich kann trivialer Natur sein, wie beispielsweise die Änderung des Erscheinungsbilds. Man spricht aber auch von Redesign, wenn ein Produkt oder System komplett umgewandelt wird.

Im Folgenden werden nun die generellen Prinzipien wie Abstraktion, Partitionierung und Ebenenübergänge beschrieben. Anschließend folgen Betrachtungen zu zwei der am stärksten eingebunden Domänen, nämlich der Mikroelektronik und der Mechanik.

3.1 Abstraktion und Partitionierung

Unabhängig davon, was man sieht oder beschreibt, bedient man sich der Abstraktion. Die Abstraktion kann dabei helfen, komplexe Zusammenhänge einfach und verständlich darzustellen. Ohne Abstraktion würde man sprichwörtlich von Hölzchen auf Stöckchen kommen. Während man die Abstraktion im normalen Leben intuitiv jeder Zeit anwendet, wird sie beim Entwurf und der Modellierung von technischen Systemen explizit eingesetzt. Der Entwurf von komplexen Systemen kann mittels der Abstraktion in Entwurfsebenen strukturiert werden. Jede dieser Ebenen beinhaltet einen gewissen Grad an Details. Für die jeweilige Ebene unwichtige Informationen werden ausgeblendet, um die Komplexität der Beschreibung auf das Wesentliche zu beschränken.

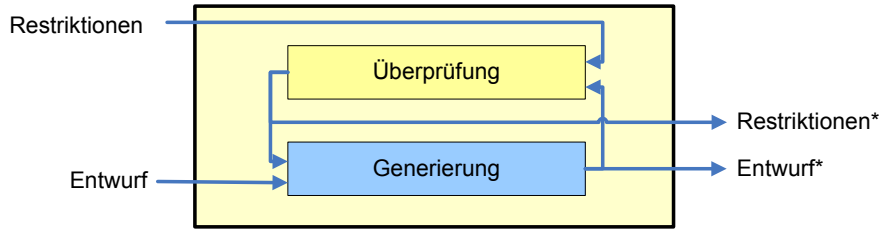


Abbildung 3.1: Kybernetisches Modell nach Rammig (vereinfacht)

Das kybernetische Modell von Rammig [Ram89] kann für eine etwas genauere Betrachtung der Abstraktions- und Entwurfsebenen genutzt werden. Rammig teilt den Entwurf in mehrere Ebenen ein, welche jeweils aus einem generierenden und einem überprüfenden Part bestehen (siehe Abbildung 3.1). Als Eingabe dienen Restriktionen und die Entwurfsanforderungen in einer für die Ebene

passenden Sprache. Restriktionen sind hier Randbedingungen, die einzuhalten sind. Das können beispielsweise in der Elektronik Stromverbrauch oder Chipflächenbedarf sein. Nachdem in einer Ebene Ergebnisse generiert wurden, werden sie zur Überprüfung den Restriktionen gegenüber gestellt. Dies geschieht so lange, bis die Restriktionen erfüllt sind. Als Ausgabe bekommt man einen überarbeiteten Entwurf und eventuell auch veränderte Restriktionen.

Je nach Komplexität des zu entwerfenden Gegenstands kommen unterschiedlich viele Abstraktionsebenen zum Einsatz. Stellt man sich beispielsweise einen aktuellen Mikroprozessor auf Transistorebene vor, so wird schnell klar, dass ohne Abstraktion eine Beherrschung der Details nahezu unmöglich ist. Den gegensätzlichen Prozess zur Abstraktion, also die Erhöhung des Detailierungsgrades, nennt man Spezialisierung. Die Spezialisierung beschreibt den Vorgang der Präzisierung. Mittels der Abstraktion und der Spezialisierung kann man also den Detailierungsgrad eines Entwurfs verändern.

Eine weiteres Hilfsmittel, das heutzutage unerlässlich ist, ist die Partitionierung. Die Partitionierung beschreibt den Vorgang des Aufteilens. Eine große und umfassende Problemstellung kann mit Hilfe der Partitionierung in kleinere und leichter begreifbare Probleme aufgeteilt werden. Bereits Julius Caesar bediente sich vor mehr als 2000 Jahren der Partitionierung. Er teilte unter anderem das besetzte Gallien in drei Provinzen auf, um sie leichter beherrschen zu können [Cae]. Noch vor Caesar nutzte Philipp von Makedonien das gleiche Prinzip. Das geflügelte Wort „Teile und Herrsche“ geht auf ihn zurück.

3.2 Entwurfsstile

Für den Übergang von einer Abstraktionsebene zur nächsten bzw. für die Reihenfolge des Durchlaufs der Abstraktionsebenen gibt es unterschiedliche Ansätze. Das wohl intuitivste Vorgehen ist der *zergliedernde Entwurf*, auch *Top-Down Entwurf* genannt. Man beginnt auf der Entwurfsebene mit dem höchsten Grad an Abstraktion. Das ist in der Regel eine allgemeine Beschreibung des Gegenstands. Die nächste Ebene, die zu wählen ist, ist die mit dem zweithöchsten Abstraktionsgrad. So fährt man fort, bis man zu einem Abstraktionsgrad gelangt ist, der ausreichend für beispielsweise die Fertigung ist. Die

einzelnen Schritte nennt man auch *Synthese* (siehe dazu auch Abbildung 3.2). Die Synthese beschreibt den Vorgang, bei dem man von einer Beschreibung (Verhalten oder Struktur) mit bestimmten Abstraktionsgrad auf eine Beschreibung mit geringerem Abstraktionsgrad schließt.

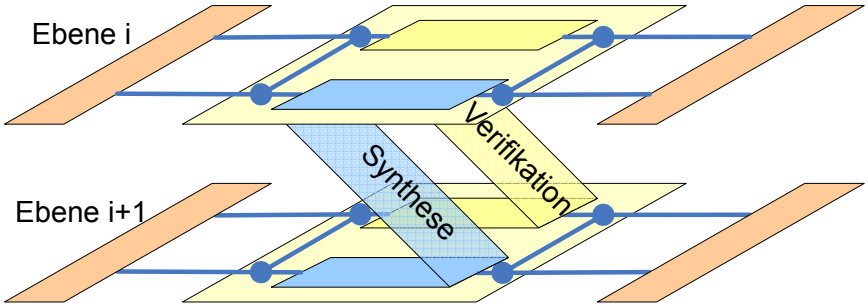


Abbildung 3.2: Übergänge im Entwurf nach Rammig

Rammig stellt die Synthese als Übergänge zwischen zwei Ebenen dar. Mittels der Synthese steigt man im kybernetischen Modell eine Ebene tiefer. Abbildung 3.2 stellt dies auf den ersten Blick gut dar. Allerdings darf man sich nicht davon verwirren lassen, dass in der Abbildung die Synthese vor Beginn der generierenden Aktion der Ebene i zum Ende der generierenden Aktion der Ebene i+1 verläuft. Intuitiver wäre sicherlich die Synthese vom Ende der generierenden Aktion der Ebene i zum Anfang derselben in Ebene i+1 darzustellen. Abbildung 3.3 gibt eine Gesamtübersicht über den Mehrebenenentwurf nach Rammig. Deutlich zu erkennen sind die verschiedenen Ebenen.

Der *aufbauende* oder *Bottom-Up Entwurf* geht den gegensätzlichen Weg. Man beginnt auf der Abstraktionsebene mit dem geringsten Abstraktionsgrad und erhöht dann Schritt um Schritt den Abstraktionsgrad. In der Regel geht das einher mit einer Partitionierung. Das bedeutet, dass Partitionen mit sehr hohem Detailierungsgrad erstellt werden, die dann im nächsten Schritt zusammengeführt werden können. Dieses Vorgehen nennt man auch Analyse oder Verifikation. In der Praxis wird man eher eine Kombination aus beiden Vorgehen vorfinden. Zum einen ist hier der *Meet-In-The-Middle Entwurfsstil* zu erwähnen. Bei dieser Methode führt man sowohl den Top-Down Entwurf als auch den Bottom-Up Entwurf bis zu einem definierten Treffpunkt durch. Zum Anderen ist das *Yoyo-Verfahren* zu nennen. Bei diesem pendelt man wie ein

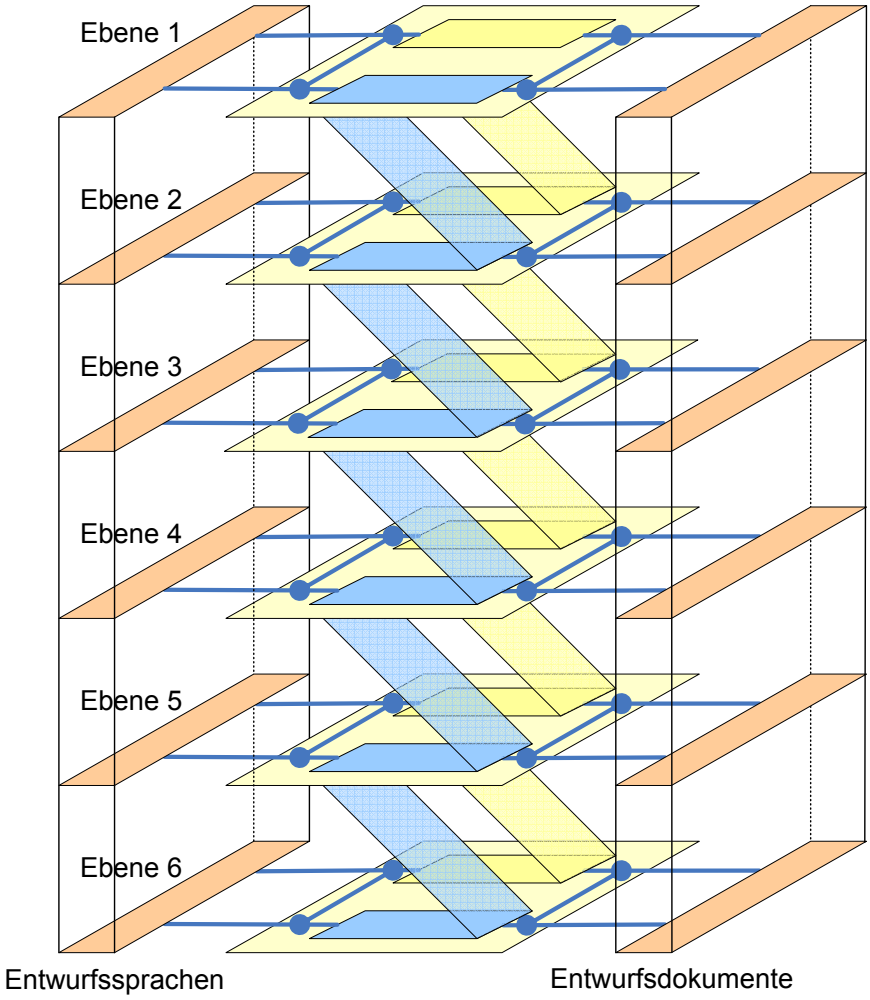


Abbildung 3.3: Gesamtübersicht der Entwurfsebenen nach Rammig

Yoyo zwischen benachbarten Abstraktionsebenen hin und her [Ram89]. Abbildung 3.4 stellt die einzelnen Entwurfsstile grafisch gegenüber.

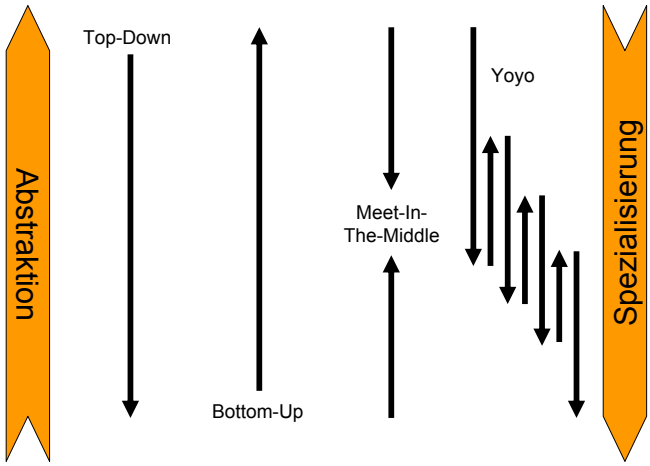


Abbildung 3.4: Entwurfsstile

Bevor nun die unterschiedlichen Methodiken diskutiert werden, sei noch zu erwähnen, dass Änderungen im Entwurf in der Regel auf möglichst hoher Abstraktionsebene erfolgen sollten. Je detaillierter der Entwurf ist, desto stärker wirken sich globale Änderungen auf den Änderungsaufwand aus. Das bedeutet aber auch, wenn man annehmen kann, dass ein Entwurf idealerweise mit dem höchsten Maß an Abstraktion startet, dass Änderungen möglichst früh berücksichtigt werden sollten. Das zeigt, wie wichtig die Spezifikationsphase im Entwurf als initiale Entwurfsphase ist.

3.3 Mikroelektronik

Seit ihrer Geburtsstunde im Jahr 1948 hat die Mikroelektronik ein enormes Wachstum durchgemacht. Im Jahr 1965 beobachtete Gordon Moore, einer der Gründer des Weltkonzerns Intel, die Verdopplung der Transistordichte jedes Jahr [Moo65]. Diese Beobachtung, die bereits vier Jahre nach der ersten planaren integrierten Schaltung gemacht wurde, hat sich bis heute bewahrheitet und ist allgemein unter *Moore's Law* bekannt[1]. Heute werden Chips gefertigt,

[1] Sie wurde lediglich soweit abgeschwächt, dass man heute von einer Verdopplung alle 18 Monate spricht.

die mehr als 100 Millionen Transistoren auf einem Die vereinigen. Systeme, die früher aus vielen Bauteilen und -gruppen zusammengestellt wurden, werden heute als sogenannte *System-On-Chip (SOC)* hergestellt. Laut der *Semiconductors Industry Association (SIA)* ist zumindest noch dieses Jahrzehnt mit dem gleichen rasanten Wachstum der Transistorendichte zu rechnen. Die physikalischen Grenzen sind noch lange nicht erreicht.

Zwar verspricht eine solche Entwicklung ungeahnte Möglichkeiten, allerdings zeigen sich aber auch immer mehr Probleme. Die Softwareunterstützung des Entwurfs kann mit dem von der Technologie vorgelegten Tempo immer weniger mithalten. Man spricht vom sogenannten *Design Gap*.

Design Gap Mit Design Gap wird das Phänomen beschrieben, dass die Entwurfsunterstützung nicht mehr mit der Entwicklung der Technologie Schritt halten kann. Durch die zunehmende Miniaturisierung der Bauteile treten Phänomene auf, die von der verfügbaren Software nicht berücksichtigt werden. Besonders in der analogen Mikroelektronik und im Mixed-Signal Bereich tritt dieses Phänomen vermehrt auf. Das liegt zum einen daran, dass die Analogtechnik noch keine so gute Softwareunterstützung wie die Digitaltechnik besitzt, zum anderen aber auch sicherlich an der ihr eigenen Inhomogenität. Während in der Digitaltechnik fast ausnahmslos Transistoren vorkommen, die allein in ihren Dimensionen variiert werden, greift die Analogtechnik auf eine Vielzahl unterschiedlicher Bauteile zurück.

Aber nicht nur die Analog- und Mixed-Signal-Bereiche haben mit weniger guter Entwurfs- und damit Softwareunterstützung zu kämpfen. Durch die zunehmende Miniaturisierung der digitalen Baugruppen, werden auch Digital-Designer immer mehr mit Problemen konfrontiert, die vormals den Analog-Entwicklern vorbehalten waren. So lässt sich prognostizieren, dass die Unterschiede beim Entwurf von höchstintegrierten digitalen und analogen Schaltungen immer mehr verwischen.

Im Folgenden werden sowohl für die digitale Mikroelektronik als auch für die analoge und den Mixed-Signal Bereich die gängigen Entwurfsmethodiken vorgestellt. Beschränkt werden soll die Vorstellung auf die allgemein bekannten Modelle. Sicherlich gibt es noch eine Vielzahl von Variationen und kom-

plexerer Darstellungen. Sie tragen hier jedoch weniger zum Verständnis der Problematik bei.

3.3.1 Digitale Mikroelektronik

1979 stellten Carver Mead vom California Institute of Technology (*Caltech*) und Lynn Conway von Xerox Corp. ihr Buch *Introduction to VLSI systems* [MC80] vor. Es sollte eine Lücke in der verfügbaren Literaturlandschaft schließen. Sie hatten festgestellt, dass Chip-Architekten selten an der Spezifikation der ICs teilhatten, mit denen sie arbeiteten. Auch an den Universitäten wurde zwischen IC-Entwurf und Systemarchitektur unterschieden [MC80]. Mit diesem Werk legten sie den Grundstein für ein durchgehendes Verständnis beider Bereiche und für den modernen VLSI-Entwurf. Das gemeinhin als *Mead-Conway Methode* oder *Mead-Conway Abstraktion* bekannte Vorgehen trennt den Entwurf des Systems von der Fertigung. Dazu werden Fertigungsaspekte auf geometrische Regeln abstrahiert, die beim Entwurf eingehalten werden müssen. Die Regeln werden so konservativ gestaltet, dass selbst worst-case Szenarien problemlos gemeistert werden können. Der Entwickler muss beim Entwurf nur auf die Einhaltung der Regeln achten und kann sich sicher sein, dass sein Entwurf auch fertigbar ist.

In den 1980er Jahren kam es dann in der Mikroelektronik zur Design-Krise. Die Fortschritte in der Technologie nahmen so rasant zu und erlaubten so komplexe Chips, dass der Entwurf eines Chips genauso viel Zeit in Anspruch nahm, wie die geplante Lebensdauer sein sollte [GT88]. Der Ruf nach adäquater Software-Unterstützung wurde immer lauter. Es entstanden die ersten *Silicon-Compiler*. Sie erlaubten den Entwurf auf immer höheren Abstraktionsstufen, beschleunigten den Entwurf und verbesserten die Entwurfsqualität durch *correctness by construction* [GT88]. Jedoch hatten auch diese Silicon-Compiler ihre Schwächen. Zum einen brauchte man für jede Technologie (CMOS, GaAs, Bipolar...) eigene Compiler, zum anderen war die Abstraktion von Änderungen in der Fertigung mittels Entwurfsregeln noch nicht so ausgereift. Dazu Gajski [GT88]: „*Some compilers must be fine-tuned when process design rules change.*“

Mehr als ein Jahrzehnt später werden digitale Schaltungen bereits von Studenten im Grundstudium selbstständig entworfen. Die fast durchgängige Au-

tomatisierung des Digitalentwurfs und die Trennung der Technologiedaten vom Entwurf machen dies möglich. Die Fertigung von digitalen Systemen geschieht heutzutage in der Regel auf CMOS-Basis, deshalb unterscheiden sich die Fertigungsverfahren nicht wesentlich. Das liegt auch daran, dass im Grunde genommen ein Bauteil, nämlich der Transistor, immer wieder in unterschiedlichen Dimensionen zum Einsatz kommt. Letzteres ermöglicht auch die relativ einfache Migration von Entwürfen von einer Fertigungslinie auf eine andere. Deshalb ist es auch möglich die Besonderheiten in besagte Design Rules zu packen und automatisch prüfen zu lassen. So kann auch mit geringen Kenntnissen über die eigentliche Fertigung des zukünftigen Chips der Designer den IC entwerfen. Standardsysteme lassen sich schon gänzlich ohne Kenntnisse der Fertigung mittels Hardwarebeschreibungssprachen realisieren. Die gängigsten sind Verilog und VHDL. Mit ihnen kann jeder sein System quasi *programmieren*. Anschließend wird es mit Hilfe geeigneter Synthesewerkzeuge im Top-Down Entwurf synthetisiert. Das Ergebnis ist ein fertiger Maskensatz.

Die strikte Trennung von Entwurf und Fertigung in der digitalen Mikroelektronik hat es ermöglicht, einen strukturierten und weitgehend automatisierten Entwurfsprozess zu realisieren [Hub96]. Diese Trennung ist unter anderem erst wegen der soeben geschilderten Sachverhalte möglich. Zusätzlich weist die digitale Mikroelektronik weitere Besonderheiten auf, die dafür verantwortlich zeichnen, dass der normale Digitalentwurf heute kein großes Problem mehr darstellt. Besonders zu erwähnen ist hier die Möglichkeit ein digitales System mit Hilfe genau einer Logikfamilie (NOR oder NAND) vollständig beschreiben und entwerfen zu können. Somit kann die „Bauteilebene“ beim Entwurf um eine Abstraktionsebene angehoben werden. Anstatt mit Transistoren arbeiten zu müssen, kommt der Designer nur noch mit NOR- bzw. NAND-Gattern in Berührung und kann seinen Entwurf vollständig beschreiben. Die Gatter sind in der Regel parametrisierbar, was die Anzahl der Eingänge betrifft. Die notwendige Dimensionierung der Transistoren innerhalb eines solchen Gatters, beispielsweise beim Hinzufügen eines weiteren Eingangs, übernimmt die Software auf Basis der hinterlegten technologieabhängigen Modelle.

Bereits 1983 veröffentlichten Gajski und Kuhn ihr Y-Diagramm [GK83], das, 1985 von Walker und Thomas [WT85] überarbeitet, den Digital-Entwurf gut

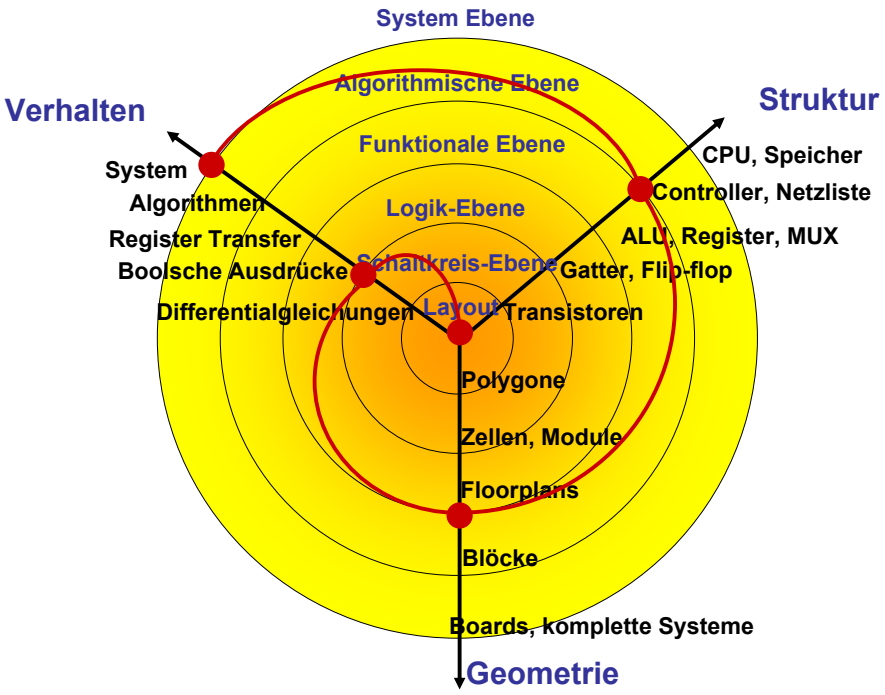


Abbildung 3.5: Y-Diagramm

veranschaulicht und erklärt. Das Y-Diagramm (siehe Abbildung 3.5) ist aufgebaut aus konzentrischen Kreisen, die die verschiedenen Abstraktionsebenen repräsentieren, und drei Y-artig angeordneten, namensgebenden Achsen. Die Achsen stellen die drei verschiedenen Sichten, nämlich Verhaltens-, Struktur- und Geometriesicht, dar. In der Verhaltenssicht wird der Entwurf als eine Art *Black-Box* mit Eingängen und Ausgängen gesehen. Hier interessiert weniger der Aufbau als mehr die Funktion dieser Black-Box. Die Struktursicht dient als Brücke zwischen der Verhaltens- und der Geometriesicht. Sie bildet die Verhaltensbeschreibung auf ein oder mehrere Komponenten ab. Je nachdem welche Nebenbedingungen bezüglich Kosten, Fläche oder Timing einzuhalten sind, können unterschiedliche Komponenten für eine Verhaltensbeschreibung zum Zuge kommen [GDWL92]. Die geometrische Sicht geht einen Schritt weiter. Sie bildet die Komponenten der Struktursicht auf ihre physischen Charakteristika wie Abmessung und Anordnung der Komponenten und der Verdrahtung dazwischen ab. Sie gibt Auskunft über die Lage jedes einzelnen Pins, über den gesamten Strombedarf, Hitzeentwicklung und Größe des Chips [GVNG94].

Der Abstraktionsgrad der Kreise nimmt von außen nach innen ab. Der äußer-

ste Ring beginnt mit der Systemebene. Es folgen die algorithmische Ebene, die funktionale Ebene, die Logik-Ebene und die Schaltkreis-Ebene. Im Folgenden sollen die Ebenen kurz vorgestellt werden.

Systemebene: Auf der Systemebene wird die Schaltung in Bezug auf ihr Verhalten durch ihre grundlegenden Eigenschaften mittels Algorithmen, Flussdiagramme o. ä. beschrieben. Über Signale oder deren detailliertes Verhalten wird hier noch keine Aussage gemacht. Die Struktur beschränkt sich auf gröbere Blöcke, beispielsweise Speicher, Prozessoren, Controller und Bussysteme. Die Geometrie des Systems wird in Teilflächen aufgeteilt, wobei Funktionen generell auch auf mehreren Chips verteilt werden können.

Algorithmische Ebene: Das Verhalten der Schaltung wird in der algorithmischen Ebene durch nebenläufige Prozesse repräsentiert. Die Prozesse bestehen aus Funktionen, Prozeduren oder Kontrollstrukturen. Die Struktur ergibt sich aus einzelnen Blöcken, die mittels Signalen miteinander kommunizieren können. Aus Geometriesicht wird die Grobeinteilung der Systemebene nun in einen feineren, sogenannten Floorplan[2] überführt.

Funktionale Ebene: Die Funktionale Ebene wird auch Register-Transfer-Ebene genannt. Das Verhalten wird durch Transfers zwischen Registern und einer Menge mathematischer Operationen dargestellt. Der strukturelle Aufbau verknüpft die einzelnen Blöcke höherer Ebenen mit Signalen. Hier treten ALUs, Register und Speicher auf. Dadurch entsteht die Verbindungsstruktur. Aus geometrischer Sicht wird das Floorplanning noch verfeinert. Als Beschreibung dienen nun Makrozellen und Blöcke.

Logik-Ebene: Das Verhalten auf Logik-Ebene wird durch logische Verknüpfungen und deren Zeitverhalten veranschaulicht. Zur Beschreibung dienen Boole'sche Gleichungen und endliche Automaten. Durch Anwendung der Verknüpfungen auf die Eingangssignale ergibt sich der Signalverlauf an den Ausgängen. Die Grundelemente (AND-, OR-Gatter, Flip-Flops usw.) bilden die Struktur auf der Logik-Ebene. Die Auflistung der

[2] Ein Floorplan beschreibt die Lage der einzelnen Strukturen auf dem Schaltkreis.

Komponenten und die Angabe der Verbindungsstrukturen ergibt zusammen die sogenannte Netzliste. Zellen und Gatter bilden zusammen die Geometriesicht.

Schaltkreis-Ebene: Das Verhalten auf Schaltkreis-Ebene wird durch mehrwertige diskrete Gleichungen modelliert. Die Struktur wird durch Netzlisten elektrischer Bauelemente, z. B. Transistoren, beschrieben. Auf diese Weise werden die einzelnen Module nicht mehr durch logische Funktionen mit einfachen Verzögerungen, sondern durch den tatsächlichen physikalischen Aufbau repräsentiert. Aus geometrischer Sicht erhält man ein Layout, das durch Polygonzüge, die die unterschiedlichen Materialien und Schichten darstellen, beschrieben wird.

Eine Methodik wird im Y-Diagramm durch Angabe von Schnittpunkten der Ebenen mit den Achsen angegeben. Das Y-Diagramm selbst legt kein explizites Vorgehen fest, jedoch hat sich eine spiralförmige top-down Methodik, ausgehend vom Verhalten über die Struktur bis hin zur Geometrie, etabliert (siehe Abbildung 3.5). Die Übergänge von einem Schnittpunkt zum nächsten werden Syntheseschritte genannt. Neben der High-Level-Synthese, die den Entwurf von der Struktur- oder algorithmischen Ebene zur funktionalen Ebene überführt, spricht man noch von Logik-Synthese und Layout-Synthese. Erstere überführt den Entwurf zur Schaltkreisebene, letztere dann in ein Layout. Sie alle werden heutzutage von mächtigen Softwarepaketen unterstützt. Das Endergebnis des Digitalentwurfs ist in der Regel ein Maskensatz, der den IC ausreichend für die Fertigung beschreibt. Wie schon erwähnt, sind die prozessrelevanten Informationen bereits über die Design-Rules in den Entwurf eingeflossen und somit im Maskensatz berücksichtigt. Auf die Entwicklung des Fertigungsprozesses wird eben aus diesen Gründen beim Y-Diagramm nicht eingegangen. Für detailliertere Beschreibungen des Entwurfs sei auf die Originalliteratur [GK83, WT85, GT88, GDWL92, GVNG94] verwiesen.

Natürlich gibt es für den VLSI-Entwurf noch andere Vorgehensmodelle. Allerdings hat sich keines so durchgesetzt, wie das Y-Diagramm. Es wurden Erweiterungen des Y-Diagramms zum X-Diagramm in [Ram89] vorgestellt, die noch eine weitere Achse einführt, die den Test explizit berücksichtigen soll; das sogenannte Rugby-Modell [JKH99], das beispielsweise auf Timing-Probleme näher eingeht oder der Design-Cube [EHM96], der allerdings für ein

eingängiges Entwurfsvorgehen viel zu kompliziert ist. Auf diese alle soll nicht weiter eingegangen werden, da sie in der Praxis eher untergeordnete Rollen spielen. Für Details sei auf die angegebene Literatur verwiesen.

3.3.2 Analoge und Mixed-Signal Mikroelektronik

Wie bereits erwähnt, ist der Entwurf von analogen oder mixed-signal Schaltungen oft ungleich komplizierter und weniger unterstützt als der Digital-Entwurf. Das liegt auch sicherlich daran, dass in der analogen Welt nicht alles nur, salopp gesagt, aus Einsen und Nullen besteht. Digitale Systeme arbeiten mit diskreten Werten, während analoge Systeme eine Vielzahl Veränderlicher haben. Zwar sind in letzter Zeit viele analoge Anwendungen durch digitale Signalprozessoren (DSP) ersetzt worden, allerdings ist dies nicht in allen Bereichen machbar. Hoch- und Höchstfrequenzanwendungen, Antennen und andere Schnittstellen zur Umwelt können gar nicht oder nur sehr aufwendig digital ersetzt werden. Auch wird es sich eher um eine zeitliche Verschiebung der Probleme handeln, da bereits oben erwähnt wurde, dass durch die fortschreitende Integration und Miniaturisierung auch die digitale Welt auf analoge Probleme stößt.

Eines der großen Probleme des Analog- und Mixed-Signal-Entwurfs ist das Fehlen geeigneter Softwareunterstützung. Im Gegensatz zum klassischen Digitalentwurf, der weitgehend automatisiert abläuft, müssen Analogentwickler noch Vieles von Hand machen. Die Problematik spitzt sich noch weiter zu, wenn man den Bedarf an analogen Systemen analysiert. Zunehmend werden neue Systeme entwickelt, die größtenteils auf analoger Technik basieren. Der Trend wird nicht zuletzt durch die Nutzung von drahtlosen Kommunikationssystemen wie beispielsweise Mobiltelefone und Funknetzwerke verstärkt. So steigt zwar die Nachfrage, die wiederum eine Beschleunigung des Entwurfsprozesses fordert, jedoch fehlt geeignete Unterstützung.

Beim Entwurf von gemischten digitalen und analogen, also mixed-signal Systemen kommt noch hinzu, dass sie typischerweise in digitale und analoge Bereiche aufgeteilt und getrennt voneinander entworfen werden. Erst sehr spät werden dann beide Teilsysteme wieder zusammengeführt. Fehler führen dann zu sehr zeitintensiven Iterationen durch den jeweiligen Design-Prozess. Der

Flaschenhals liegt aber auch hier bei den analogen Teilsystemen. Wie bereits oben beschrieben, ist der digitale Entwurf sehr gut unterstützt. Unberechenbar ist jedoch der analoge Part. Zwar wurden Pendants und Erweiterungen zu den digitalen Hardwarebeschreibungssprachen entwickelt (Verilog-A, VHDL-AMS), jedoch sind sie bei weitem noch nicht so weit verbreitet wie ihre Verwandten aus der digitalen Welt. Mit diesen Beschreibungssprachen und entsprechender Bibliotheken ist zumindest der Grundstein für einen Top-Down Entwurf auch in der analogen Welt gelegt.

In der analogen und mixed-signal Mikroelektronik gibt es im Gegensatz zur digitalen Welt kein so allgemein bekanntes und akzeptiertes Entwurfsvorgehen wie das Y-Diagramm. Im Jahr 1996 schlugen Forscher der Toshiba Corp. einen Top-Down-Entwurf auch für die analoge Mikroelektronik vor. Sie stellten ein Verfahren vor, dass die Wiederverwendung von Bauteilen ähnlich wie beim digitalen Entwurf unterstützt [MOM96]. Jedoch fehlte eine detaillierte Beschreibung der Methodik. Vielmehr stellten sie ein Internet-basiertes System zur Verwaltung von Bauteilen vor.

Im Jahr 2000 brachte es Ken Kundert von Cadence Design Systems auf den Punkt [Kun00]: „*With mixed-signal designs becoming more complex and time-to-market windows shrinking, designers cannot hope to keep up unless they adopt a more formal process for design and verification: top-down design.*“ Es kann nicht verleugnet werden, dass hier gewisse Parallelen zur Design-Krise Anfang der 1980er Jahre im VLSI-Entwurf offensichtlich werden. Kundert weiter: „*In analog design, the primary culprits behind the poor productivity of those at the bottom of the scale are increasingly complex designs combined with a continued preference for a bottom-up (transistor level) design methodology and the occurrence of simulation late in the design cycle, which leads to errors and respins. There's a huge disparity in productivity between mixed-signal designers who have made the transition to an effective top-down design methodology and use mixed-signal hardware description languages and those who practice bottom-up design and rely solely on spice.*“ Hier wird klar, dass auch die analoge Welt dringend ein strukturiertes Vorgehen braucht und auch umsetzen muss. Sicherlich lässt sich das nicht so einfach umsetzen wie in der digitalen Welt, jedoch gibt es bereits viel versprechende Ansätze. Einen abstrahierten und leicht verständlichen top-down Ablauf beschreiben Gielen und Rutenbar in [GR01] (siehe Abbildung 3.6). In den aufkommenden Design-Werkzeugen findet sich häufig dieser Ablauf wieder [GR01].

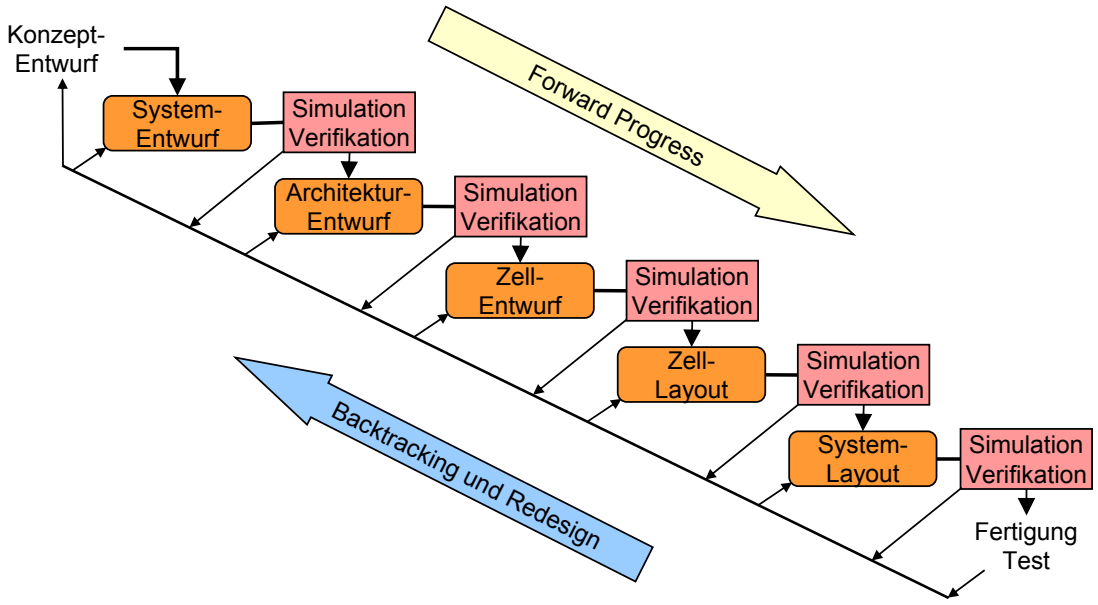


Abbildung 3.6: Analog- und Mixed-Signal-Entwurf nach [GR01]

Konzept-Entwurf: Während der ersten Phase, dem konzeptionellen Entwurf, wird das Gesamtkonzept des Produkts und dessen Spezifikationen bestimmt. Auch werden Kostenschranken und Markteintritt festgelegt. Diese Phase ist besonders wichtig, da sie über Erfolg oder Misserfolg des Produkts entscheidet.

System-Entwurf: Die Phase des Entwurfs des Systems ist die erste Stufe des eigentlichen Entwurfs. Die Gesamtarchitektur mit der zugehörigen Technologie, verwendeten Bibliotheken und Teststrategien wird festgelegt und eventuell aufgeteilt. Es erfolgt die Aufteilung in Hardware und Software mit den zugehörigen Schnittstellen, die in geeigneten Beschreibungssprachen spezifiziert werden. Auf Seiten der Hardware erfolgt das auf Ebene der Verhaltensbeschreibung. Die Partitionierung und Spezifikationen werden mittels detaillierter Cosimulationen validiert.

Architektur-Entwurf: Die Hardware wird in funktionale Blöcke zerlegt, die für das spezifizierte Verhalten von Nöten sind. Das umfasst auch die Auftrennung nach digitalen und analogen Bereichen. Die Teilspezifikationen der einzelnen Blöcke werden festgelegt und mit geeigneten Hardwarebeschreibungssprachen beschrieben. Die nun festgelegte Architektur wird wieder gegen die Spezifikation validiert. Dafür können mixed-signal-

Simulationen genutzt werden, mit denen auch das Zusammenspiel der einzelnen Blöcke untereinander überprüft werden kann.

Zell-Entwurf: Die verschiedenen analogen Blöcke werden entsprechend der Spezifikation und ausgewählten Technologie entworfen. Das Ergebnis ist eine vollständige Beschreibung des ICs auf Bauelementebene. Die Schaltkreisparameter und die Topologie werden festgelegt. Dieser Vorgang wird auch *Sizing* genannt, weil bei diesem Vorgang in der Regel auch die physischen Größen (bezogen z. B. auf die Maske) festgelegt werden. Zu komplexe Bereiche werden nochmals in kleinere Bereiche aufgeteilt und einzeln behandelt. In dieser Phase spielen auch schon Fertigungsprozesse eine Rolle, die beispielsweise einzuhaltende Toleranzen festlegen. Wieder wird das Ergebnis mit Hilfe von Schaltkreissimulatoren validiert.

Zell-Layout: Diese Stufe des Entwurfs zeichnet sich durch die Transformation der Schaltkreisbeschreibung in ein geometrisches Maskenlayout aus. Das Hauptproblem ist hier die Kompromissfindung zwischen genutzter Fläche und elektrischer Leistungsfähigkeit. Berücksichtigt werden müssen auch parasitäre Effekte, die den Entwickler dazu zwingen können, die Leistungsfähigkeit des ICs nach unten korrigieren zu müssen. Anschließend kann aus dem erzeugten Layout wieder eine Beschreibung extrahiert werden, die es ermöglicht, auch die parasitären Effekte bei der Simulation mit einzubeziehen.

System-Layout: Beim System-Layout werden sowohl die analogen als auch die digitalen Bereiche auf dem zukünftigen Chip angeordnet. Zusätzlich erfolgt die Verdrahtung, die nicht nur die Verbindungen der Blöcke untereinander berücksichtigen muss, sondern auch die chipweiten Signale wie Taktgeber oder allgemeine Stromversorgung. Besonders bei mixed-signal Systemen ist es schwierig, parasitäre Effekte vorherzusagen. Elektrisches Übersprechen, Spannungseinbrüche oder Einkopplungen durch das Substrat sind die dominierenden Effekte beim mixed-signal Design. Deshalb ist es besonders wichtig, auf ausreichende Abschirmung zu achten. Teststrukturen und -leitungen werden auch in dieser Phase eingebaut, um ein späteres Testen des Chips zu ermöglichen. Schließlich wird das System mittels Hardware-Software-Cosimulation getestet.

Fertigung und Test: In der letzten Phase werden die Masken hergestellt und der IC gefertigt. Während und nach der Fertigung wird der Chip getestet, um defekte ICs direkt auszusortieren.

3.4 Mechanik

Die Mechanik ist wohl die älteste und bestverstandene Ingenieursdisziplin überhaupt. Seit mehreren Tausend Jahren werden mehr oder weniger komplexe mechanische Systeme entwickelt. Bis zur Jahrhundertwende zum 20. Jahrhundert war es ohne Probleme möglich, als einzelner Entwickler ein System in seiner Gesamtheit in absehbarer Zeit zu entwickeln. Danach jedoch stieg die Komplexität der Systeme stark an. Betrachtet man dies am Beispiel des Automobils, so wird es sehr schnell deutlich. Das erste Auto bestand im Grunde nur aus einer umgebauten Kutsche, die mit einem Motor versehen wurde. Hier lag die Ingenieurleistung sicherlich im Motor und nicht in der Kutsche. Der wurde aber von einer Person entworfen und gebaut. Mit der Zeit nahm die Komplexität der Fahrzeuge zu. Heute kann man sicherlich mit Fug und Recht sagen, dass ein Ingenieur allein kein aktuelles Auto in angemessener Zeit in Gänze entwerfen könnte. Ganz zu schweigen von der Flugzeugindustrie. Die Entwicklung eines Passagierflugzeugs beispielsweise bei Boeing dauert 10.000 Mannjahre [Ull02]. Solche Leistungen sind nur in Teams machbar. Diese Teams benötigen allerdings ein wohldefinierte und ausgeklügelte Entwurfs- und Kommunikationsstrategie.

In der Mechanik profitiert der Entwurf sehr stark von Bibliotheken. Das ist sicherlich ein unschätzbarer Vorteil für die Mechanik. Fast jedes mechanische System besteht aus mehreren Teilen. Diese Teile können in der Regel getrennt von einander entworfen und gefertigt werden. Beim Entwurf des Gesamtsystems entstehen Spezifikationen für jedes benötigte Teil. Mittels dieser Spezifikationen kann jedes Teil für sich hergestellt werden. Solange es die Spezifikation einhält (und der Gesamtentwurf richtig war), wird auch das aus den Teilen zusammengesetzte Gesamtsystem die Spezifikation erfüllen.

Angefacht von den Erfolgen, die man mit strukturierten Vorgehensmodellen in der Mikroelektronik erreicht hatte, veranstaltete die *National Science Foundation* (NSF) der Vereinigten Staaten von Amerika 1994 einen Workshop mit

dem Titel „New Paradigms for Manufacturing“. Ziel dieses Workshops war, herauszufinden, ob man auch in der Mechanik ähnlich gut strukturierte Vorgehensmodelle wie in der digitalen Mikroelektronik finden kann. Jedoch war das Ergebnis ernüchternd. Auf Grund der Diversivität der Mechanik konnte kein allgemeingültiges Vorgehen gefunden werden. Einzig für den damals noch sehr jungen Bereich der Mikromechanik sah man Chancen [Ant96a]. Die Literatur, die sich mit dem Entwurf in der Mechanik beschäftigt, beschränkt sich deshalb mehr auf die übergeordneten Entwurfsstrategien, wie beispielsweise die VDI-Richtlinie 2221, die das Vorgehen beim Entwickeln und Konstruieren im Maschinenbau beschreibt [Kas00], bzw. auf Verfahren, mit denen sich spezielle durch die Spezifikation hervorgerufene Probleme lösen lassen (siehe [Hor02]).

In [Kas00] werden die Ergebnisse einer Untersuchung der verschiedenen Entwurfsabläufe vorgestellt. Die Untersuchung kommt zu dem Ergebnis, dass es zwar verschiedene Vorschläge gibt, diese ähneln sich jedoch sehr. Die grundlegenden Entwurfsphasen sind sogar bei allen Abläufen gleich.

Ein typischer Entwurfsprozess wird als Beispiel in Abbildung 3.7 dargestellt. Bei dieser Vorgehensweise handelt es sich in erster Linie um ein klassisches Top-Down Vorgehen. Allerdings sind Rücksprünge in beliebiger Weite und Anzahl möglich. In dieser Interpretation des Entwurfs in der Mechanik folgt auf jede Tätigkeit ein Ergebnis. Das Ergebnis kann als Eingabe für die nächste Tätigkeitsphase dienen oder zu Rücksprüngen animieren. In der Abbildung sind schon einige Tasks inhaltlich gruppiert. Diese Inhaltsgruppen sollen nun genauer beschrieben werden.

Clarification of the task: Die erste Phase des Entwurfs dient der Analyse der Anforderungen. Die für das zu entwerfende mechanische System notwendigen Funktionen werden analysiert. Dies geschieht unter Beachtung eventueller Nebenbedingungen. Als Ergebnis dieser Tätigkeit erhält man eine Spezifikation.

Concept Generation: Mit Hilfe der Spezifikation kann nun ein Gesamtkonzept entwickelt werden. Dafür muss als erstes das Kernproblem identifiziert werden. Erst wenn man die eigentliche Problematik verstanden hat, kann man sich Gedanken über mögliche funktionale Strukturen machen. Nachdem ein oder mehrere Lösungsprinzipien gefunden wurden,

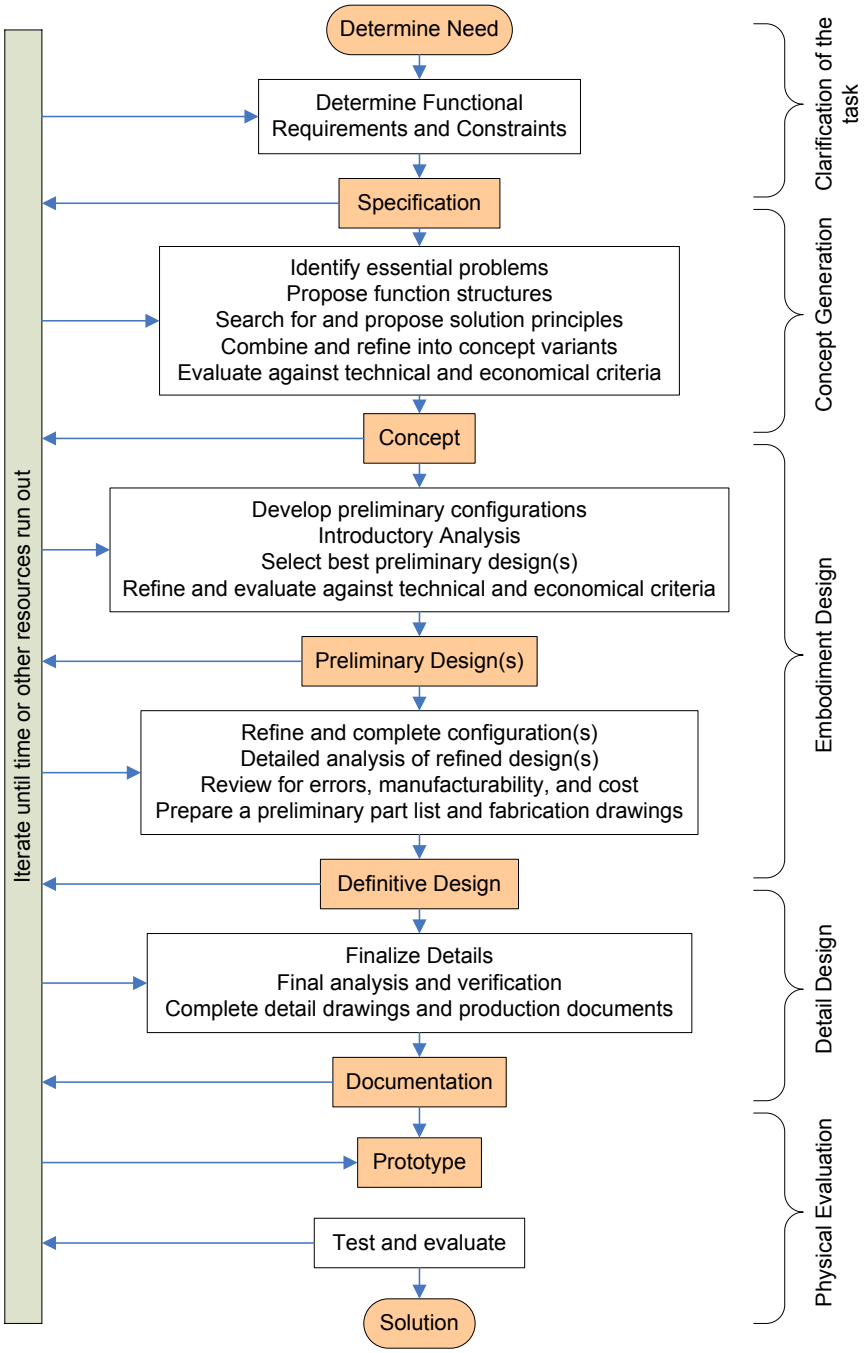


Abbildung 3.7: Entwurfsvorgehen in der Mechanik nach [Ant96a]

werden verschiedene Lösungskonzepte entwickelt und gegen technische und wirtschaftliche Randbedingungen geprüft. Auf diese Weise erhält man ein Konzept.

Embodiment Design: Steht das Konzept, so können vorläufige Konfigurationen des Systems entwickelt werden. Diese Konfigurationen werden einer Voranalyse unterzogen und ein oder mehrere Entwürfe kommen in die engere Auswahl. Diese Entwürfe werden dann wieder auf Einhaltung der technischen und wirtschaftlichen Randbedingungen überprüft. Das Ergebnis ist ein vorläufiger Entwurf. Dieser wird nun ausgearbeitet und die vorläufige Konfiguration wird vervollständigt. Anschließend wird der Entwurf analysiert und auf Fehler untersucht. Weiterhin werden die Fertigbarkeit sowie die Kosten untersucht. Mit der Aufstellung erster Teilelisten und Blaupausen endet diese Phase. Man hat nun einen festen Entwurf vorliegen.

Detail Design: Der vorliegende Entwurf wird nun endgültig gemacht. Letzte Analysen und Prüfungen des Entwurfs sollen für eine eindeutige und fertigbare Lösung garantieren. Die Blaupausen werden so weit verfeinert und detailliert, dass jede Komponente eindeutig beschrieben ist. Das Ergebnis ist eine umfassende Sammlung von Dokumenten, die das gesamte System vollständig beschreiben.

Physical Evaluation: Mit der Dokumentensammlung kann dann die Fertigung beginnen. In der Regel wird erst ein Prototyp gefertigt, der noch einmal gegen alle Anforderungen geprüft werden kann. Erst wenn der Prototyp für zufriedenstellend erklärt wurde, beginnt die eigentliche Fertigung.

3.5 Fazit

In diesem Kapitel wurden die methodischen Grundlagen für die folgenden Betrachtungen des Mikrosystemtentwurfs vorgestellt. Es wurden der Begriff des Entwurfs konkretisiert und die verschiedenen Arten des Entwurfs betrachtet. Besonders wichtig ist das kybernetische Entwurfsmodell von Rammig, da die

Entwurfsmodelle für die Mikrosystemtechnik, die im folgenden Kapitel vorgestellt werden, größtenteils darauf basieren.

Die Betrachtungen zur Mikroelektronik haben gezeigt, dass besonders der Digitalentwurf methodisch sehr gut erfasst ist. Beim Analogentwurf genauso wie beim mixed-signal-Entwurf hat man jedoch auch noch heute mit großen Problemen zu kämpfen. Wie sich noch zeigen wird, sind diese Probleme ähnlich den Problemen in der Mikrosystemtechnik. Die Mechanik kann nicht mit solch detaillierten Vorgehensmodellen aufwarten. Das liegt sicherlich daran, dass der Entwurf in der Mechanik zu einer anderen Entwurfsklasse gehört. In der Mechanik trifft man in der Regel auf einen Baukastenentwurf. Das heißt, dass der Designer sich aus einem sehr großen Fundus von bereits definierten Bauteilen bedienen kann. Die Bauteile werden so wie sie sind oder parametrisiert in den Entwurf eingebracht. Sowohl die Fertigung als auch mikroskopische Wechselwirkungen, wie sie in der Mikroelektronik oder Mikrosystemtechnik zu beachten sind, haben meist keinen Einfluss auf den Entwurf in der Mechanik.

Sowohl die Mikroelektronik als auch die Mechanik spielen eine große Rolle in der Mikrosystemtechnik. Im folgenden Kapitel wird der Mikrosystementwurf im Detail betrachtet, wobei die bereits in diesem Kapitel vorgestellten Disziplinen und deren Betrachtungen mit in die Diskussion einfließen.

4 Mikrosystementwurf

„The primary method for creating a MEMS mask-layout today is trial-and-error“. Diesen Satz schrieb Antonsson 1995 in seinem Beitrag zum NSF-Workshop *„Structured Design Methods for MEMS“* [Ant96b]. Sein Statement liegt nun fast ein Jahrzehnt zurück und man könnte annehmen, dass sich seit dem einiges im Bereich des Mikrosystemtentwurfs geändert hat. Dem ist aber beileibe nicht so. Auch wenn sich in den letzten Jahren einiges im Bereich der Methodik und deren softwaremäßigen Unterstützung getan hat, zeigt die Resonanz sowohl aus der Industrie wie auch aus dem akademischen Umfeld, dass die Unterstützung unzureichend, ja teilweise sogar mangelhaft ist [Wag01, LN04].

Die in den letzten Jahren auf den Markt gekommenen Werkzeuge sind in der Regel Spezialwerkzeuge, die auf bestimmte Problemstellungen zugeschnitten sind. Eine durchgängige Unterstützung wie sie beispielsweise in weiten Teilen der Mikroelektronik vorzufinden ist, ist zur Zeit nicht in Sicht. Für eine durchgängige Unterstützung benötigt man eine Strategie, eine Methodik für den Entwurf. Eine solche soll in diesem Kapitel erarbeitet werden.

Im Folgenden wird eine mögliche Klassifizierung von Mikrosystemen eingeführt. Anschließend werden die verschiedenen Entwurfsebenen des Mikrosystementwurfs erläutert. Dann wird gezeigt, dass herkömmliche Entwurfsmodelle nicht so einfach auf die Mikrosystemtechnik übertragen werden können. Es folgt eine Vorstellung des verhaltensnahen und des fertigungsnahen Entwurfs.

4.1 Klassifizierung von Mikrosystemen

Mikrosysteme lassen sich in drei Klassen einteilen. Das sind Technologie-Demonstratoren, Forschungssysteme und kommerzielle Systeme [Sen01].

Technologie-Demonstratoren: In der Mikrosystemtechnik werden immer wieder neue Technologien und Verfahren entwickelt. Das Ziel der meisten Bemühungen ist dabei, bereits verwendete Verfahren zu verbessern bzw. abzulösen oder überhaupt erst Lösungen realisierbar zu machen. Ein Technologie-Demonstrator ist ein Mikrosystem, bei dem eine neue Technologie oder ein neues Verfahren zur Fertigung eingesetzt wird. Oft ist der einzige Sinn dieses Systems, zu zeigen, dass die Technologie oder das Verfahren funktioniert und einsetzbar ist. Dabei geht es weniger um das System als Anwendung, sondern mehr um die Herstellung. Es ist also Mittel zum Zweck. Solche Demonstratoren werden meistens in Forschungseinrichtungen hergestellt.

Forschungssysteme: Bei den Forschungssystemen handelt es sich um sogenannte Vorausentwicklungen. Das heißt, man versucht Anwendungen für die Zukunft zu finden. Ein Thema solcher Forschungssysteme könnte beispielsweise sein, ein heutiges System, das auf herkömmliche makroskopische Art und Weise hergestellt wird, durch ein Mikrosystem zu ersetzen. Deshalb sind Forschungssysteme in der Regel Prototypen, die nicht zur Anwendung kommen. Die Herstellung dieser Prototypen ist meist auch Thema der Forschung. So können Forschungssysteme auch gleichzeitig Technologie-Demonstratoren sein.

Kommerzielle Systeme: Kommerzielle Mikrosysteme zeichnen sich dadurch aus, dass eine Vielzahl an gleichen Systemen hergestellt wird und ein System dem anderen bis ins Detail gleicht. Die Fertigung ist fest vorgeschrieben. Veränderungen, speziell bei der Fertigung, werden, wenn überhaupt, nur in sehr geringem Umfang vorgenommen. Wie der Name schon sagt, dienen die kommerziellen Systeme dazu, Geld zu verdienen. Um die hohen Entwicklungskosten bei sehr geringen Stückpreisen wieder einzuspielen, müssen sehr große Mengen hergestellt werden. Die Stückzahlen können leicht in den zweistelligen Millionenbereich gehen.

Die obige Einteilung der Mikrosysteme spielt insbesondere bei der Betrachtung der Entwurfsmethoden und der Entwurfsunterstützung eine Rolle, da sich die Vorgehen je nach Entwurfsziel, sei es Technologie-Demonstrator, Forschungssystem oder kommerzielles System, doch unterscheiden können.

Eine weitere Unterscheidung kann man nach dem Initiator der Entwicklung machen. Auf der einen Seite gibt es Mikrosysteme, die auf Grund der Nachfrage am Markt (*market-driven*) entwickelt werden. Auf der anderen Seite stehen solche Systeme, die bereits vorhandene ersetzen sollen und in der Regel besser, schneller, günstiger oder zuverlässiger sind. Das sind Mikrosysteme, die auf Grund der Technologie entwickelt wurden (*technology-driven*). Erstere haben den Vorteil, dass eine klare Nachfrage auf dem Markt existiert. Letztere hingegen müssen sich gegen evtl. schon vorhandene und etablierte Systeme durchsetzen. Insbesondere wenn ein Paradigmenwechsel vorliegt, also die Funktionsweise gänzlich von der bekannten abweicht, spielt immer ein gewisses Risiko bei der Markteinführung mit. Allerdings sagt dazu Senturia in [Sen01]: *„Most truly successful MEMS devices have been paradigm shifts.“*

4.2 Entwurfsebenen

Die Entwurfsebenen oder -phasen der Mikrosystemtechnik lassen sich für einen Überblick in vier Bereiche einteilen. Das sind die Systemebene, die Deviceebene, die physische Ebene und die Prozessebene (siehe auch Abb. 4.1).

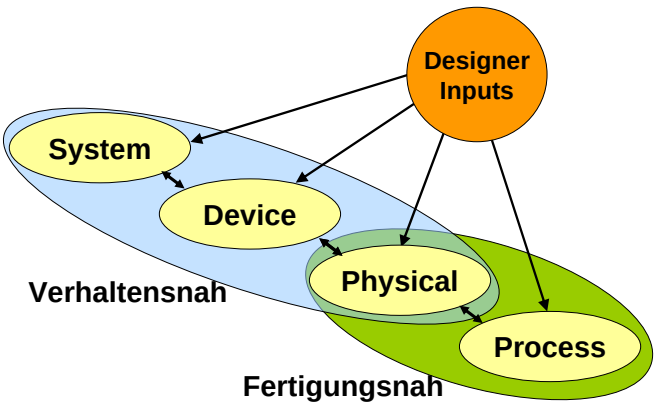


Abbildung 4.1: Übersicht verhaltensnaher und fertigungsnaher Entwurf

Systemebene: Die Systemebene in der Mikrosystemtechnik ist vergleichbar mit der Systemebene der Mikroelektronik. Auf Systemebene wird ein

System als solches beispielsweise in Form von Blockschaltbildern und Verhaltensbeschreibungen beschrieben. Die Systemebene steht in der Abstraktionshierarchie des Entwurfs an oberster Stelle. Sie dient dazu, das Gesamtsystem zu beschreiben. Dazu orientiert man sich an dem Vorgehen in der Regelungstechnik. Das System wird durch informationsflussorientierte Übertragungselemente beschrieben. Diese Übertragungselemente bestehen aus Blöcken mit Ein- und Ausgängen, die jedoch rückwirkungsfrei sind [Lor99].

Für die Systemsimulation sind bei der Modellierung der Komponenten in der Regel Abstriche in Sachen Genauigkeit zu machen. Oftmals wird erst dadurch die Dauer der Simulation erträglich. Die Blöcke beschreiben das Verhalten des Systems, was idealerweise mathematisch geschieht. Allerdings ist das nicht immer möglich. Besonders bei mechanischen Strukturen kann das Verhalten teilweise nur durch Diskretisierung und riesigen Matrizen mit mehreren 10.000 Zeilen und Spalten beschrieben werden [RBH+02]. Um das zu umgehen, kann man sich auf Makromodelle stützen.

Deviceebene: Auf Deviceebene wird das Gesamtsystem in einzelne Subsysteme, den Devices, aufgeteilt. Zur Beschreibung derselben bedient man sich der Makromodelle und analytischer Modelle. Makromodelle sind Blöcke, die auf Vereinfachungen basieren. Sie können aus realen und idealisierten Komponenten zusammen gesetzt sein. Die Simulation von Makromodellen ist zwar langsamer als die der Verhaltensmodelle, aber deutlich schneller als beispielsweise die Simulation auf Netzwerkebene [Kie98].

Um ausreichend genaue Makromodelle für mikromechanische Systeme zu erstellen, muss man in der Modellierungshierarchie in der Regel bis auf die Feldsimulation (s. u.) hinunter. Nach [RBH+02] läuft die Erstellung eines Makromodells wie folgt ab: Ausgangspunkt ist die Beschreibung der Komponente in einem Finite-Elemente Modell. Nach der Gittererzeugung (Diskretisierung) werden wichtige Punkte im Modell ausgewählt. Das sind in der Regel Randpunkte, die das Verhalten des Systems widerspiegeln. Anschließend wird das Modell auf diese ausgewählten Punkte vereinfacht. Abschließend kann das Verhalten auf Basis der Vereinfachung beschrieben werden. Dies kann auch in Hardware-

beschreibungssprachen wie VHDL-AMS oder Verilog-AMS geschehen. Beispiele dafür sind in [TLMP02, Fed03] zu finden.

Kriterien für ein gutes Makromodell hat Senturia in [Sen98] zusammengefasst:

- vorzugsweise analytisch (besser als numerisch); erlauben dem Entwickler Schlussfolgerungen über die Auswirkungen von Änderungen
- korrekte Abhängigkeiten von Geometrie und grundlegenden Eigenschaften
- korrektes Verhalten bei Energieerhaltung und -wandlung
- deckt sowohl quasi-statisches als auch dynamisches Verhalten ab
- einfache Darstellung als Formel, Netzliste oder gekoppelte einfache Differentialgleichungen
- einfache Anbindung an Simulatoren auf Systemebene

Für eine umfassendere Beschreibung der Makromodellbildung sei hier auf [GD97] und [Fed03] verwiesen.

Physische Ebene: Auf der physischen Entwurfsebene trifft man auf die Feldmodellierung und 2D- bzw. 3D-Modellierung des Entwurfs. In dieser Phase ist das spätere Aussehen des Entwurfsgegenstands bereits offensichtlich. Die einzelnen Schichten des Mikrosystems stehen fest und sind detailliert beschrieben, d. h. Material und Materialparameter sind bekannt. Bei der Feldmodellierung werden in der Regel kritische Bereiche des Mikrosystems modelliert und simuliert, was wiederum Rückschlüsse für die Makromodellierung zulässt. Das Ergebnis ist oft eine dreidimensionale Darstellung des Systems.

Prozessebene: Die Prozessebene wird von der Technologie beherrscht. Die für die Herstellung des Mikrosystems notwendigen Prozessschritte werden gefunden, parametrisiert und zu Prozesssequenzen zusammen gefügt. Zusätzlich müssen Masken für eventuelle Lithografieschritte erstellt werden. Die Prozessebene ist die am wenigsten abstrakte Entwurfsphase. Die Kombination aus parametrisierter Prozessfolge und Masken stellt eine eindeutige Beschreibung des Mikrosystems dar. Auf dieser

Ebene entscheidet sich letztendlich, ob ein Entwurf realisierbar also fertigbar ist oder nicht.

Wie schon allgemein im Kapitel 3 über Entwurfsmethodik erläutert wurde, gibt es unterschiedliche Herangehensweisen beim Entwurf. In der Mikrosystemtechnik speziell kann man grundsätzlich zwei globale Entwurfsstrategien feststellen. Zum einen gibt es den verhaltensnahen Entwurf, der die ersten drei der soeben beschriebenen Entwurfsphasen umfasst. Diese Art des Entwurfs ist stark verhaltens- bzw. strukturorientiert und geht deshalb auch die Entwurfsproblematik über eben solche Beschreibungen an. Zum anderen spricht man vom fertigungs- oder technologienahen Entwurf, wenn die beiden letzten Phasen betroffen sind. Der fertigungsnahe Entwurf ist stark an der Technologie und der Modellierung der Fertigung an sich interessiert.

Bevor nun im Detail auf die beiden Entwurfsausprägungen eingegangen wird, soll der Gedanke der Adaption von Entwurfsmodellen diskutiert werden. Die Nähe der Mikrosystemtechnik zu ihren Wurzeln in der Mikroelektronik und Feinmechanik lässt die Vermutung zu, dass der Entwurf zumindest ähnlich der Ursprungsdisziplinen erfolgen kann. Wäre das der Fall, so ließe sich eventuell sogar die Unterstützung dieser Disziplinen in der Mikrosystemtechnik einsetzen.

4.3 Adaption von Entwurfsmodellen?

Bei der Findung und Entwicklung von Vorgehensmodellen für die Mikrosystemtechnik hat man immer wieder versucht, Modelle aus verwandten Bereichen direkt oder leicht verändert zu übernehmen. Besonders die Mikroelektronik, hier speziell der Digitalentwurf, wurde dafür gerne herangezogen. In den 1990er Jahren versuchte man, angespornt durch die Erfolge, die man bei der Mikroelektronik verbuchen konnte, ausgehend vom Digitalentwurf auch für die Mikrosystemtechnik ähnliche Modelle zu entwickeln.

1995 veranstaltete die amerikanische *National Science Foundation* (NSF) einen Workshop zur Findung einer durchgängigen Entwurfsmethodik für die Mikrosystemtechnik (*Structured Design Methods for MEMS*, [Ant96b]).

Bevor nun die Adaption der verschiedenen Entwurfsstrategien diskutiert wird, soll Tabelle 4.1 die Gemeinsamkeiten und Unterschiede zwischen der digitalen Mikroelektronik, Mechanik und Mikrosystemtechnik hervorheben. Aus der Gegenüberstellung kann man das unspektakuläre Ergebnis ziehen, dass die Mikrosystemtechnik sowohl Gemeinsamkeiten mit der Mikroelektronik als auch mit der Mechanik hat.

Obwohl auf den ersten Blick mehr Übereinstimmungen zwischen Mechanik und Mikrosystemtechnik vorhanden sind als zwischen Mikroelektronik und Mikrosystemtechnik, gibt die Betrachtung der Fertigung den Ausschlag.

Die Mechanik hat, wie bereits im vorigen Kapitel gezeigt wurde, keine so detaillierte Methodenlandschaft wie die Mikroelektronik. Insbesondere fehlt ein Pendant zu dem allseits anerkannten Y-Diagramm. So fällt es schwer, eine geeignete Methode zu finden und entsprechend anzupassen. Wie bereits in [Ull02] ausführlich gezeigt wurde, spielt die Fertigung in der Betrachtung des Entwurfs in der Mechanik eine eher unbedeutende Rolle. Dazu auch Gerlach in [GD97]:„*Durch die Komplexität und Miniaturisierung der Mikrosysteme sowie dem hohen Grad der funktionalen und räumlichen Integration kommt dem Entwurf in der Mikrosystemtechnik gegenüber den herkömmlichen Bereichen wie Maschinenbau und Feinwerktechnik eine erheblich größere Bedeutung zu.*“ In [Kas00] findet man noch zwei weitere Gründe, die gegen die Mechanik als Methodenspender sprechen. Zum einen orientiert sich der Entwurf in der Mechanik an klassischen Entwurfsaufgaben, die in der Regel manuell vollzogen werden. Zum anderen sind klassische Mechanikentwürfe eben nur mechanisch und sind nicht geeignet, auf Multidomänenentwürfe verallgemeinert zu werden. Von daher liegt der Schluss nahe, dass es nicht sinnvoll ist, Mechanikmethoden, so sie denn vorhanden sind, auf die Mikrosystemtechnik zu übertragen.

Mikrosysteme, die durch lithografiebasierte Verfahren hergestellt werden, haben in der Regel mehr Gemeinsamkeiten mit der Mikroelektronik als mit der Mechanik. Wie bereits in Kapitel 2.1 gezeigt wurde, sind ein Großteil der Fertigungsprozesse in der Mikrosystemtechnik modifizierte Verfahren aus der Mikroelektronik. Auch kann die Mikroelektronik als eine Untergruppe der Mikrosystemtechnik angesehen werden, die eben nur die elektronische Domäne abdeckt. Das und die oben genannten Argumente gegen die Übernahme der Mechanikmethoden legen es nahe, eher die Methoden der Mikroelektronik als

VLSI Systems	**Macro-mechanical Systems**	**MEMS**
Single energy domain	Multiple coupled energy domains	Multiple coupled energy domains
Small set of primitives; decomposition is easy	Wide range of unstructured non-modular elements; hierarchical functional decomposition is not easy	The range of elements is not as broad as macro systems; scope for limited decomposition
Elements are clearly distinguishable functionally and topologically	Intrinsically shared topological boundaries. No direct mapping between function and form	Same as macro systems; topological segmentation is equally hard (e.g., a fluid volume bounded by moving parts)
Simple interconnection rules (KVL & KCL) among the decomposed elements	Interconnection rules are complex	Same as macro systems
Geometry of physical artifacts is not a big issue in design	Geometry of artifacts is intrinsically tied to the function they perform. Kinematics plays a big role	Same as macro systems, but kinematic issues are not as complex at present. Predominantly monolithic compliant structures
Manufactured with planar lithography	Wide range of manufacturing techniques including 3-D machining	Same as VLSI systems

Tabelle 4.1: Gegenüberstellung VLSI, Mechanik, Mikrosystemtechnik [AS96]

Ausgangsbasis für die Entwicklung von Vorgehensmodellen in der Mikrosystemtechnik zu nehmen als die der Mechanik.

Jedoch lassen sich die Modelle der Elektronik nicht so einfach übernehmen, wie es einigerorts versucht wurde. Das liegt primär an der Multifunktionalität und dreidimensionalen Charakteristik der Mikrosysteme. Während in der Mikroelektronik in erster Linie nur zwei Dimensionen eine Rolle spielen und die Schichtfolge nur durch die Masken zum Tragen kommen, spricht man dort auch von 2,5 Dimensionen. Zusätzlich findet man in der Mikroelektronik oft Manhattan-Design, also rechtwinklige Strukturen, höchstens jedoch 45° Strukturen vor. In der Mikrosystemtechnik kann man diese Vereinfachungen nicht machen, da sich Mikrosysteme im Gegensatz zu elektronischen Schaltungen weder in ein rechtwinkliges Raster, noch auf eine Folge von Masken mit fester Prozessfolge reduzieren lassen.

Der große Unterschied zwischen Digitaltechnik und allgemeiner Mikrosystemtechnik liegt darin, dass in der Digitaltechnik nahezu jede beliebige Funktion mit nur einer fixen Fertigungsfolge erreicht werden kann. Die Fertigungsfolge als Variable entfällt also. Im Gegensatz dazu ist die Fertigungsfolge eine sehr wichtige Variable in der Mikrosystemtechnik. Ohne die Variation derselben könnten viele Funktionen gar nicht realisiert werden.

Trotz dieser Überlegungen findet man häufiger Ansätze, die versuchen, das Y-Modell der Mikroelektronik auf die Mikrosystemtechnik abzubilden. Ein Beispiel dafür ist in Abbildung 4.2 zu sehen. Jedoch sind solche Versuche nicht besonders geeignet. Das Y-Modell fußt auf der Grundlage, dass eine strikte Trennung von Entwurf und Fertigung besteht. Es modelliert daher nur die Entwurfsphasen, die nötig sind, um ein zweidimensionales Layout für die Masken zu erstellen. Auf die eigentliche Fertigungsplanung geht es eben deshalb nicht ein. Das Y-Modell kann daher, wenn überhaupt, nur als Leitfaden für die sehr frühen und verhaltensnahen Entwurfsphasen in der Mikrosystemtechnik herhalten.

In der Mikroelektronik ist der Schritt von der abstrakten Beschreibung zur Fertigung relativ unkritisch (*„There is a clean separation between the processing done during wafer fabrication and the design effort that creates the pattern to be implemented.'"* [MC80]). In der Mikrosystemtechnik jedoch macht besonders der multifunktionelle Charakter eines Systems die Umsetzung des abstrakten Entwurfs

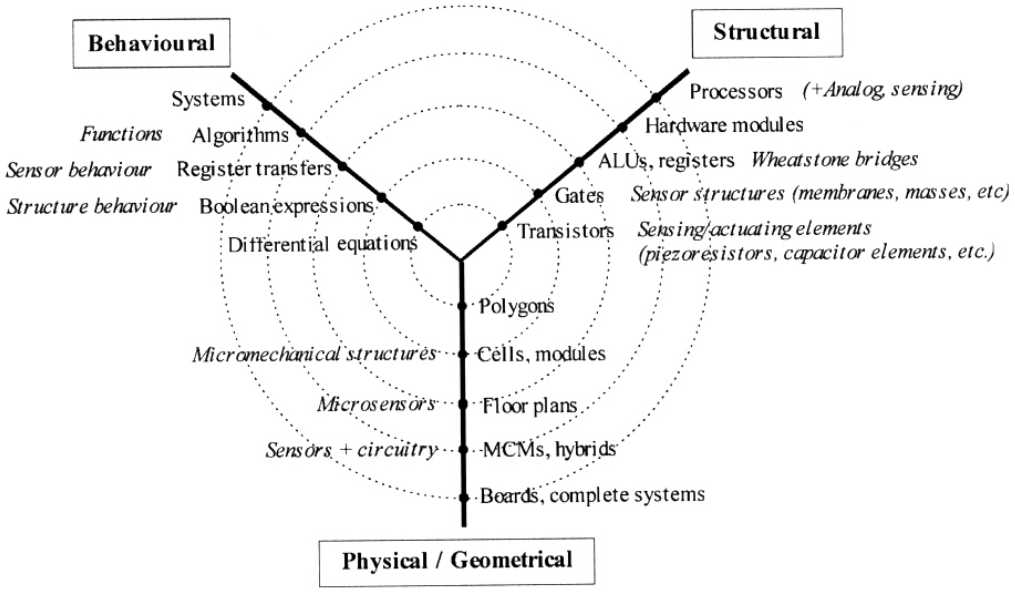


Abbildung 4.2: Y-Diagramm angewendet auf die Mikrosystemtechnik [Bau99]

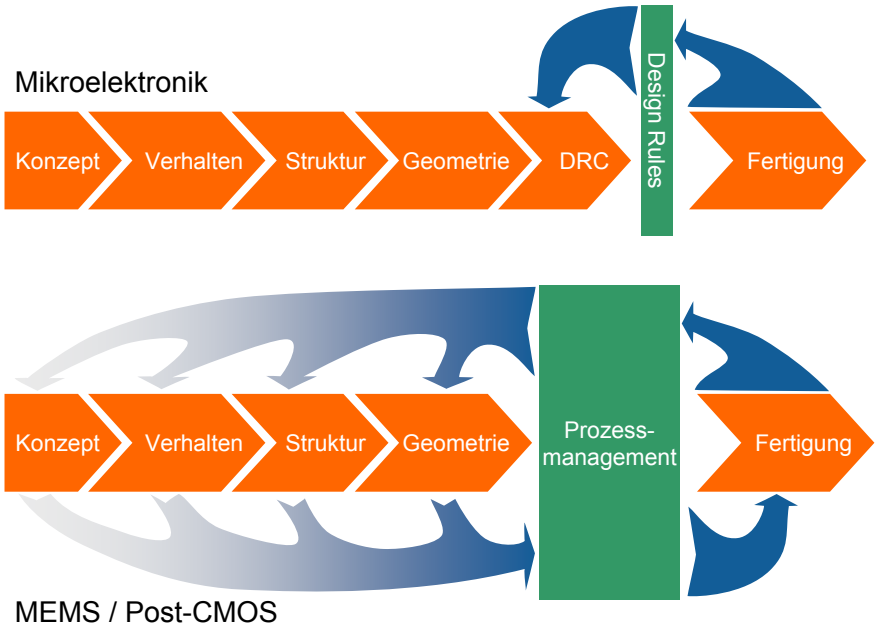


Abbildung 4.3: Einfluss der Fertigung in der Mikroelektronik und Mikrosystemtechnik

in ein physisches Modell sehr schwierig. Das verdeutlicht auch Abbildung 4.3. Wie man dort deutlich sieht, sind die einzelnen Entwurfsphasen bis zur Fertigungsplanung an sich in der Mikroelektronik und der Mikrosystemtechnik annähernd gleich. Während jedoch der Einfluss der Fertigung in der Mikroelektronik deutlich und scharf eingegrenzt ist (siehe dazu auch Kapitel 3.3), ist in der Mikrosystemtechnik eher das Gegenteil der Fall.

Wie schon mehrfach erwähnt wurde, profitiert die Mikroelektronik davon, dass die Fertigung sehr invariabel ist und durch Entwurfsregeln abstrahiert werden kann. Es existiert also eine klar definierte Schnittstelle zwischen Entwurf und Fertigung. In der Mikrosystemtechnik jedoch kann man die Fertigung, die sich von System zu System ändern kann, nicht allgemein durch die Abstraktion mittels Entwurfsregeln beschreiben. Die Berücksichtigung der Fertigung beim Entwurf des Systems muss vom Entwerfer selbst bewerkstelligt werden. Möchte man also erfolgreich ein Mikrosystem, das nicht nur aus Elektronik besteht, entwerfen und später auch fertigen, so muss mit abnehmender Abstraktion mit zunehmenden Einfluss der Fertigung gerechnet und dieser auch berücksichtigt werden.

Aus diesen Überlegungen kommt man zu dem Schluss, dass weder die Mechanik noch die Mikroelektronik als Methodenspender dienen können. Lediglich Anlehnungen an deren Methodik sind in Betracht zu ziehen.

4.4 Das Brezel-Modell

Das Brezel-Modell des Mikrosystementwurfs [WH03] wurde im Rahmen dieser Arbeit speziell entwickelt, um eine Gesamtübersicht über den Entwurf zu erhalten und das Zusammenspiel von verhaltensnahen und fertigungsnahen Entwurf zu verdeutlichen. Das Brezelmodell stellt entsprechend des kybernetischen Modells von Rammig generierende und überprüfende Aktivitäten dar. Das Brezelmodell kennt fünf Zustände. Initialer Zustand ist eine Anforderung, die in der Regel in Form einer Spezifikation vorliegt. Die Struktur beschreibt das Mikrosystem auf Basis von Verhaltens- oder Strukturbeschreibungen. Das 3D-Modell wiederum stellt das System grafisch dreidimensional dar. Die Prozessfolge beschreibt die Fertigungsanweisung. Finaler Zustand ist das fertige Mikrosystem.

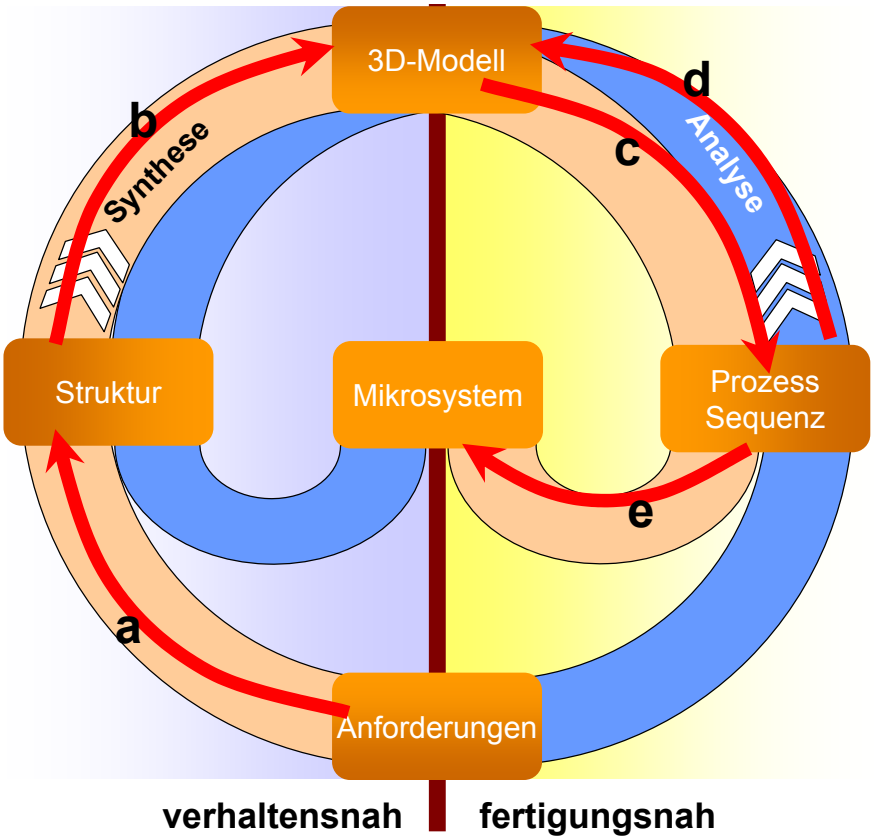


Abbildung 4.4: Brezel-Modell des Mikrosystementwurfs

Zum fertigen Mikrosystem kann man auf unterschiedliche Weise gelangen. Einmal über das Verhalten, also über primär generierende Aktivitäten, oder über die Fertigung mit primär überprüfenden Aktivitäten. Generierende Aktivitäten sind in Abbildung 4.4 orange dargestellt und überprüfende blau. Ebenfalls darstellbar sind die unterschiedlichen Entwurfsstrategien, wie Meet-in-the-middle oder Yoyo.

Das Brezelmodell kann an der Mittelvertikalen in verhaltensnahen und fertigungsnahen Entwurf aufgeteilt werden. Links findet sich der verhaltensnahe Entwurf und rechts der fertigungsnahe Entwurf wieder. In Abbildung 4.4 ist ein mögliches Entwurfsszenario aufgetragen. In den Schritten a und b werden die Funktion und die funktionalen Schichten ermittelt. Anschließend kann durch abwechselnde Folge der Schritte c und d die Fertigungsanweisung erstellt werden. Hier dargestellt ist die Yoyo-Strategie mit abwechselnd generierenden (Schritt c) und überprüfenden (Schritt d) Phasen. Ein detaillierteres Bild über die beiden Entwurfsstrategien geben die folgenden Betrachtungen.

4.5 Verhaltensnaher Entwurf

Beim verhaltensnahen Entwurf handelt es sich um ein klassisches Top-Down Vorgehen. Der Schwerpunkt dieses Entwurfsvorgehens liegt auf der abstrakten Modellierung des Systems bis zu einer dreidimensionalen Darstellung. Die Fertigung wird, wenn überhaupt, nur am Rande bzw. am Ende des Entwurfs betrachtet. Angeregt wurde diese Art des Entwurfs durch die Erfolge, die man in der Mikroelektronik mit eben dieser Entwurfsart verzeichnen konnte. Bereits 1995 schlug Fedder in [Fed96] ein solches Vorgehen auch für die Mikrosystemtechnik vor. In der Fachwelt spielt der verhaltensnahe Entwurf (engl. *behavioral design*) und seine Unterstützung eine viel größere Rolle als der fertigungsnahe Entwurf. Hier sind besonders Fedder [MF97, MF98, Fed99, Fed00, Fed03], Senturia [Sen01, Sen98] und Antonsson [Ant01, MA00, Ant96b] zu erwähnen.

In Abbildung 4.5 ist der verhaltensnahe Entwurf schematisch dargestellt. Der verhaltensnahe Entwurf beginnt mit der Spezifikation des Gesamtsystems. Das Gesamtsystem wird dann partitioniert in beispielsweise mechanische, elektrische und elektromechanische Komponenten. Anschließend kann eine

Netzliste (engl. *Schematic*) für das System erstellt werden. Eine solche Netzliste sieht ähnlich einer SPICE-Netzliste aus (siehe auch Abbildung 4.6). Jedoch besitzt sie auch Repräsentanten für die nichtelektrischen Komponenten. Mit dieser Netzliste lässt sich das Verhalten des Systems darstellen. Vorteil einer solchen Darstellung ist, neben der Verhaltensmodellierung, dass sie benutzt werden kann, um beispielsweise Auswerteelektroniken zu entwickeln und direkt an das Verhalten des zukünftigen Systems anzupassen. Die Netzliste wird dann in eine zweidimensionale Sicht des Systems überführt. Aus dieser kann dann eine dreidimensionale Darstellung erzeugt werden. Die 3D-Darstellung wird mit einem Gitter versehen und mittels Finite-Element-Methode (FEM) simuliert.

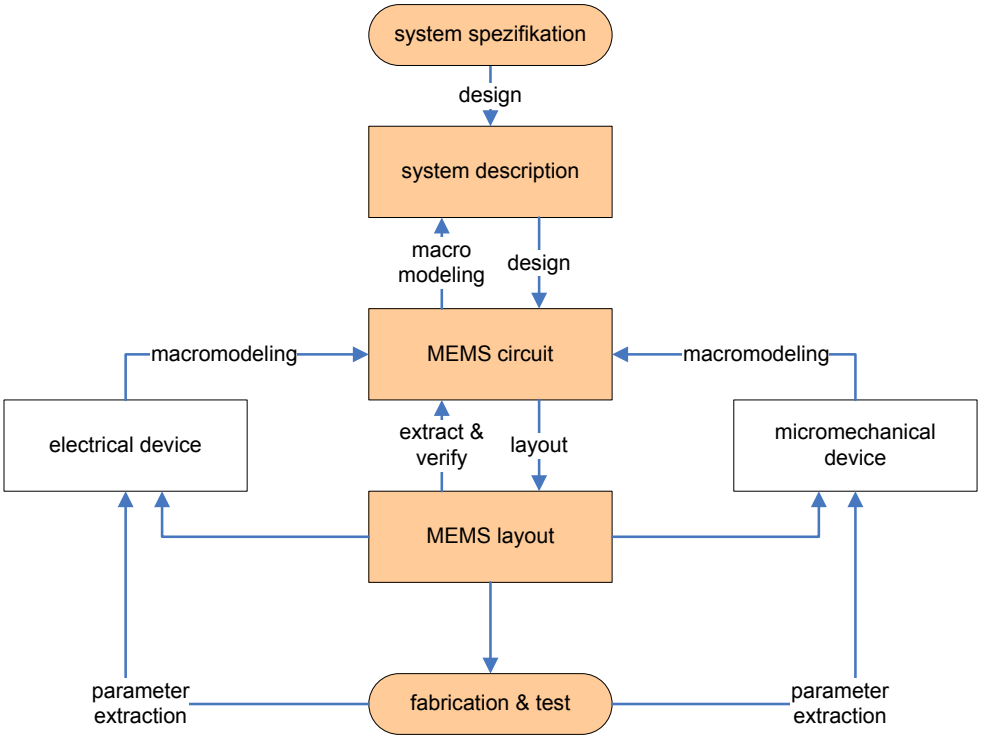


Abbildung 4.5: Verhaltensnaher Entwurf [Fed99]

Mit Hilfe der FEM lässt sich auch das Verhalten des entworfenen Systems simulieren. Aus den Ergebnissen der Simulation können dann verschiedene Parameter extrahiert werden, so dass ein Abgleich gegen die Netzliste erfolgen kann. Eventuell sind Verfeinerungen nötig, so dass es zu mehreren Iterationen kommen kann, bevor die 3D-Struktur dem gewünschten Mikrosys-

tem entspricht. Die Fertigung wird bei diesem Entwurfsstil nicht explizit angesprochen. Sie wirkt indirekt über Entwurfsregeln und technologiebezogene Modelle auf den Entwurf ein.

Eine solche Methodik ist heute bereits in begrenztem Umfang umsetzbar. Die Mikroelektronik profitiert davon, dass die Komplexität des Entwurfs durch eine Hierarchie von Komponenten und Blöcken entschärft wird. Die heutigen EDA-Tools (EDA = *Electronic Design Automation*) unterstützen diese Hierarchisierung. Transistoren können zu Gattern, Gatter zu Funktionsblöcken und Funktionsblöcke zu ganzen Systemen gruppiert werden. Das ermöglicht dem Entwickler einen schnellen Wechsel zwischen den unterschiedlichen Sichten und Abstraktionsebenen und beschleunigt damit den Entwurf enorm. Wie eine solche Hierarchisierung auch in der Mikrosystemtechnik aussehen kann, ist in Abbildung 4.7 zu sehen. Sie stellt die Dekomposition für ein elektromechanisches Mikrosystem entsprechend des verhaltensnahen Entwurfs dar. Der dargestellte Entwurf wird in drei Sub-Komponenten aufgeteilt, die wiederum aus verschiedenen Basiselementen bestehen. Diese Basiselemente sind Massekörper, Federbalken, elektrostatische Luftspalte und Anker [Fed99].

Im Gegensatz zur Elektronik benötigt man für die Beschreibung der Basis-Komponenten in der Mikrosystemtechnik auch die geometrischen und positionsbestimmenden Parameter. Das rührt daher, dass das mikromechanische Verhalten direkt von der Form, von den Materialien und auch von deren Position abhängt. Hat man diese Parameter zur Hand, so kann man bereits heute das Verhalten z. B. mit Hilfe von Analogsimulatoren und der Finite-Elemente-Methode simulieren. Modelle für die elektrostatischen Kräfte, für die Mechanik und auch die Elektromechanik inklusive Dämpfung sind bereits soweit ausgereift [Fed00].

Das ein solches verhaltensnahes Vorgehen sinnvoll ist, wird allgemein akzeptiert und es existieren bereits Software-Werkzeuge, die dieses unterstützen. Für die Simulation des Verhaltens verwendet man Makromodelle. Diese Modelle können entweder analytischer oder numerischer Natur sein. Numerische Modelle basieren beispielsweise auf Tabellen, während analytische Modelle das Verhalten in Formeln fassen. Wünschenswert sind natürlich letztere. Jedoch ist der Aufwand sehr groß, um ein analytisches Modell zu erstellen [Sen01].

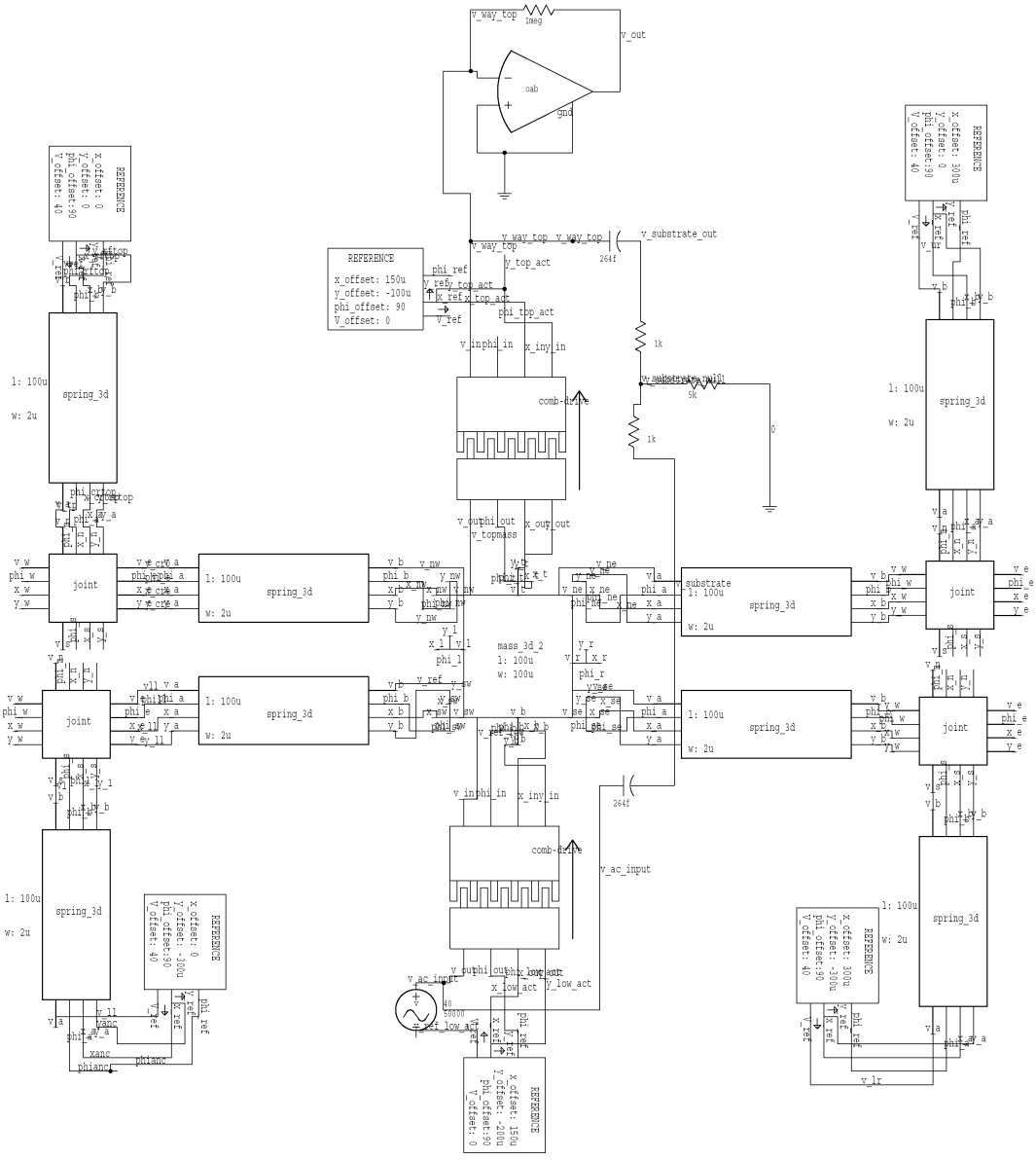


Abbildung 4.6: Netzliste eines Resonators [MF97]

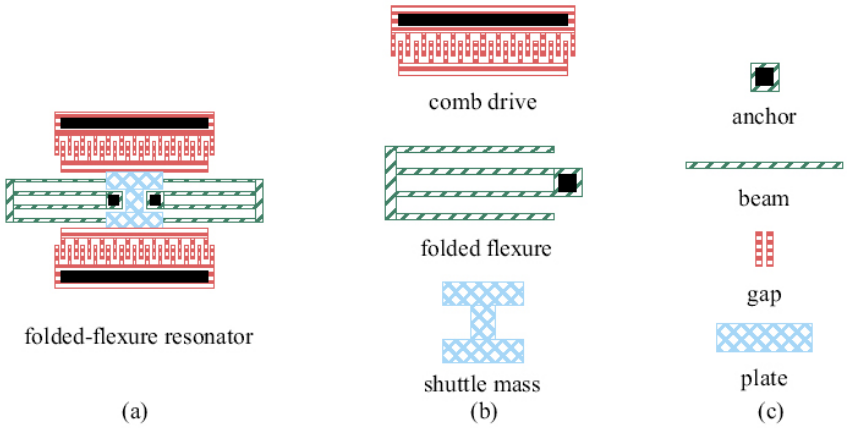


Abbildung 4.7: Hierarchieaufbau für den verhaltensnahen Entwurf [MFB99]

Fast jeder Entwurf in der Mikrosystemtechnik benötigt eine eigene angepasste Prozessfolge. Das erschwert jedoch sichtlich die Etablierung des verhaltensnahen Entwurfs. Die in den Veröffentlichungen [MF97, MF98, Fed99] vorgestellten Beispiele für einen durchgängigen verhaltensnahen Entwurf mit einem gefertigten Mikrosystem als Ergebnis basieren alle auf einer mehr oder weniger fixen Prozessfolge. So ist es möglich, genau wie in der Mikroelektronik, Entwurfsregeln zu definieren und die Modelle mit den nötigen Parametern zu füllen. Allerdings sind die Möglichkeiten der Variation stark eingeschränkt.

Für die Mikrosystemtechnik allgemein können diese Beispiele definitiv nicht stehen. Anwendung findet der verhaltensorientierte Entwurf deshalb hauptsächlich im Bereich der kommerziellen Mikrosysteme, die auf einer festen Prozessfolge basieren. Für Technologiedemonstratoren und Forschungs- und Entwicklungssysteme fehlt für einen verhaltensnahen Entwurf das umfangreiche Wissen über die Fertigung, deren Einfluss nicht zu unterschätzen ist. Für Systeme, die biologische, chemische, optische usw. Komponenten haben, lässt sich die Rückführung auf solche Basiskomponenten wie in Abbildung 4.7 nicht realisieren.

Der große Reiz dieser Methode liegt jedoch darin, dass, sobald das Fertigungswissen in Modelle und Regeln gegossen wurde, auch Entwickler ohne Prozesskenntnisse Mikrosysteme entwerfen können. Zur jetzigen Zeit ist diese Art des Entwurfs nur in den Bereichen sinnvoll, wo eine Prozessfolge unveränderbar vorgegeben ist und die Modelle vorhanden sind, so dass wie in

der Mikroelektronik von der Fertigung mittels beispielsweise Entwurfsregeln abstrahiert werden kann.

Ob für jedes Entwurfsproblem in der Mikrosystemtechnik eine allgemeine Abstraktion von der Fertigung, wie sie in der Mikroelektronik üblich ist, machbar ist, ist jedoch fraglich. Auch in der Mikroelektronik benötigt man fertigungsspezifische Entwurfsregeln. Jedoch variiert dort die Fertigung nicht von System zu System, wie es in der Mikrosystemtechnik leicht vorkommen kann. Denkbar ist aber, dass der verhaltensnahe Entwurf sich für die breite Masse der Mikrosysteme (sogenannte Standardanwendungen) einsetzen lässt. Spezial- und Neuentwicklungen müssen weiterhin fertigungsnah betrachtet werden. Jedoch prophezeiht bereits Fedder in [Fed03]: *„The next generation of MEMS system designers are starting to use composable MEMS models.“* Im Folgenden soll nun pars pro toto die Entwicklungssoftware CoventorWare vorgestellt werden. CoventorWare implementiert die soeben vorgestellte verhaltensnahe Methodik.

4.5.1 CoventorWare

Die Entwicklungsumgebung CoventorWare von der Firma Coventor [Cov04] deckt den verhaltensnahen Entwurf ab. Sie teilt sich auf in die Teilsysteme Architect, Designer, Analyzer und System Builder.

Architect Mittels des Coventor Architects können Netzlisten von Mikrosystemen erstellt werden. Dazu stellt Coventor eine Bibliothek von Komponenten aus den Bereichen Elektromechanik, Optik und Fluidik zur Verfügung. Die Komponenten können in einem gewissen Rahmen parametrisiert werden. Für einen Mikrobalken lassen sich beispielsweise die Länge, die Weite und die Dichte angeben. Die Modelle der vorbereiteten Komponenten sind in der Regel analytischer Natur. Die Komponenten können leicht zu einer Netzliste zusammen gefügt werden. Diese Netzliste kann dann nach unterschiedlichen Kriterien untersucht werden. Dazu werden einige Simulatoren bereitgestellt.

Designer Der Designer dient dazu, ein zweidimensionales Layout und ein dreidimensionales Modell zu erstellen. Das ist generell auf zwei Arten

möglich. Zum einen lassen sich die Netzlisten aus dem Architect direkt in den Designer importieren und in ein zweidimensionales Layout umwandeln. Zum anderen kann das Layout auch selbst erstellt oder aus anderen Werkzeugen im GDSII-, CIF-, oder DXF-Format importiert werden. Der Designer besitzt eine eigene Materialdatenbank. Diese wird dazu genutzt, um aus dem zweidimensionalen Layout ein dreidimensionales Modell zu erzeugen. Zusätzlich müssen allerdings hierzu schon Angaben zu den Fertigungsschritten vorliegen.

Analyzer Im Analyzer wird das dreidimensionale Modell in eine Gitterstruktur überführt. Anschließend kann das Modell mittels der Finite-Element-Methode (FEM, mehr dazu in Kapitel 5.2) untersucht werden. Dazu bietet der Analyzer verschiedene Möglichkeiten der Untersuchung nach beispielsweise elektrischen, fluidischen oder mechanischen Kriterien.

System Builder Der System Builder dient der Extraktion von Makromodellen aus der Finite-Element-Modellierung. Die Ergebnisse, die im Analyzer erzielt wurden, lassen sich mit dem System Builder in den Architect rückkoppeln. Auf diese Weise können einerseits neue Verhaltenmodelle erstellt oder andererseits vorhandene Modelle verbessert werden.

Abbildung 4.8 stellt den Entwurfsablauf mit CoventorWare noch einmal grafisch dar. Mit den vier beschriebenen Teilsystemen deckt Coventor den gesamten verhaltensnahen Entwurf ab. Unbedingt notwendig sind dafür jedoch ausreichende Komponentenbibliotheken. Für die dreidimensionale Modellierung benötigt der Designer bereits die Informationen über die Prozessschritte, die zur Fertigung notwendig sind. Hier sieht man deutlich, dass, obwohl der Designflow in Coventor schon sehr durchgängig ist, eine vollständige Abstraktion von der Fertigung nicht realisiert wurde.

4.6 Fertigungsnaher Entwurf

Der fertigungsnahe Entwurf geht den Entwurf von der Seite der Technologie an. Zwar gibt es, wie im vorhergehenden Abschnitt beschrieben, eine verhaltensorientierte Methodik. Jedoch lässt sie sich auf Grund der fehlenden Tech-

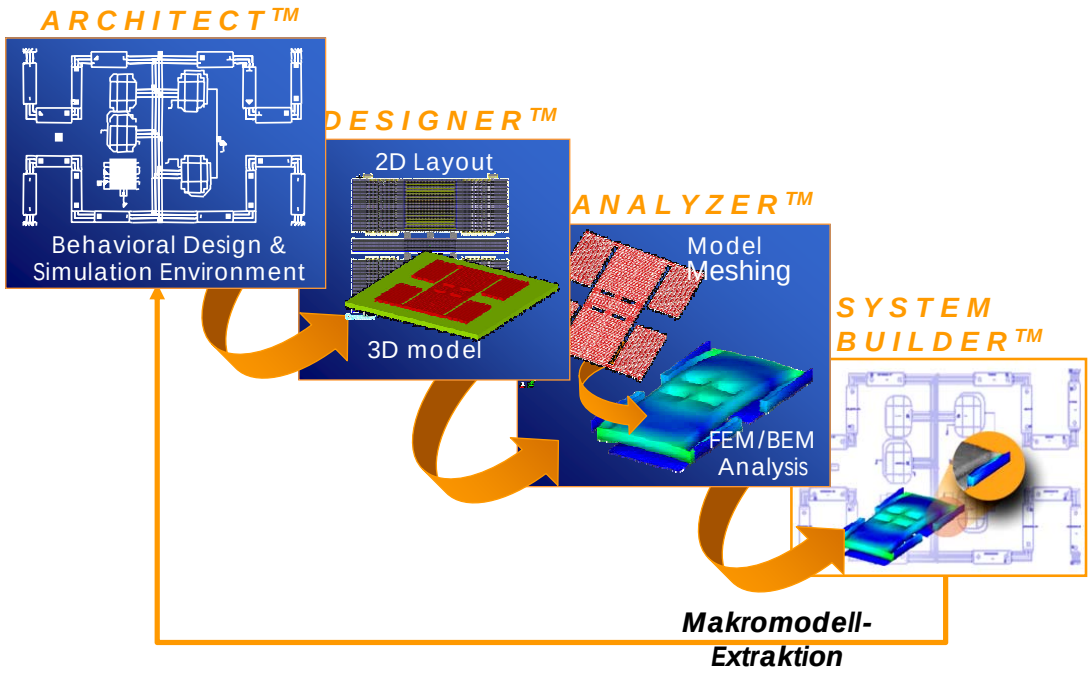


Abbildung 4.8: Verhaltensnahe Methodik mit CoventorWare [Cov04]

nologieabstraktion nur eingeschränkt einsetzen. In der Literatur wird der explizit fertigungsnahe Entwurf eher stiefmütterlich behandelt. Das ist erstaunlich, da immerhin diese Art des Entwurfs in der Industrie zur Zeit vorherrscht und, wenn auch nur teilweise, softwaremäßig unterstützt wird. Eine umfassende Beschreibung ist allerdings in [Brü96] zu finden. Eben dort findet man auch eine Definition des fertigungsnahen Entwurfs:

Fertigungsnaher Entwurf *Die Gesamtheit aller Entwurfsschritte, die unmittelbar die Erzeugung von Fertigungsanweisungen betreffen. [...] Bei technischen Gegenständen beschreiben die Fertigungsanweisungen im Wesentlichen die geometrische Struktur der Komponenten, die topologischen Gegebenheiten, die von diesen eingehalten werden müssen, sowie die Menge aller Parameter, die zur Einstellung und Handhabung der für die Fertigung benötigten Produktionsmittel erforderlich sind.*

Auf die Mikrosystemtechnik angewandt, heißt das, dass der fertigungsnahe Entwurf alle Aspekte der Fertigung, also der Prozessschritte und deren Parametrisierung, betrifft und auch die dreidimensionale Struktur berücksichtigt.

Letztere kann auch durch Querschnitte und Masken- oder Aufsichten ersetzt werden.

Der fertigungsnahe Entwurf ist zur Zeit in der Mikrosystemtechnik das dominante Entwurfsvorgehen. Ähnlich wie in den Anfängen der Mikroelektronik hat sich der fertigungsnahe Entwurf in der Mikrosystemtechnik mehr oder weniger von allein eingestellt. Da in der Mikrosystemtechnik noch keine Trennung von Funktion und Fertigung möglich ist, ist es auch leicht verständlich, dass der fertigungsnahe Entwurf fast überall anzutreffen ist. Schließlich muss die Fertigung für jeden Entwurf betrachtet werden.

4.6.1 Praxisanalyse

Im Rahmen dieser Arbeit wurde eine Analyse der Entwurfsmethodik im fertigungsnahen Bereich durchgeführt. Untersucht wurde das Vorgehen sowohl in der Industrie als auch im akademischen Umfeld [Wag01]. Abbildung 4.9 stellt das Ergebnis dieser Methodenanalyse grafisch dar. Ausgehend von einer Spezifikation werden die Funktionsprinzipien ermittelt. Diese lassen sich dann auf eine oder mehrere funktionale Schichten abbilden. Anschließend wird das Gerüst aus funktionalen Schichten mit „Fleisch“ gefüllt. Hilfsschichten werden eingeführt. Sobald die Schichtfolge steht, werden passende Prozessschritte ermittelt. Für lithografiebasierte Schritte werden zusätzlich noch die Masken erstellt. Für den Fall, dass nicht gesichert ist, dass einer oder mehrere Prozessschritte in dem Kontext prozessierbar sind, werden Schlüsselexperimente durchgeführt. Die Ergebnisse dieser Experimente fließen dann in die Überprüfung der Prozessverträglichkeit ein. In dieser Phase wird die gesamte Prozessfolge auf ihre Prozessierbarkeit überprüft. Das kann theoretisch mittels Expertenwissen geschehen oder praktisch in Form einer prototypenhaften Fertigung. Fällt das Ergebnis negativ aus, so wird der Entwurf überarbeitet. Auf diese Weise iteriert der Entwerfer durch die verschiedenen Phasen des Entwurfs. Fällt die Überprüfung der Fertigbarkeit schließlich positiv aus, so wird eine Chargenkarte erstellt. Eventuell folgt noch eine Bewertung der Fertigungsanweisung nach diversen Kriterien, wie z. B. Kosten, Zeitaufwand oder Materialverbrauch.

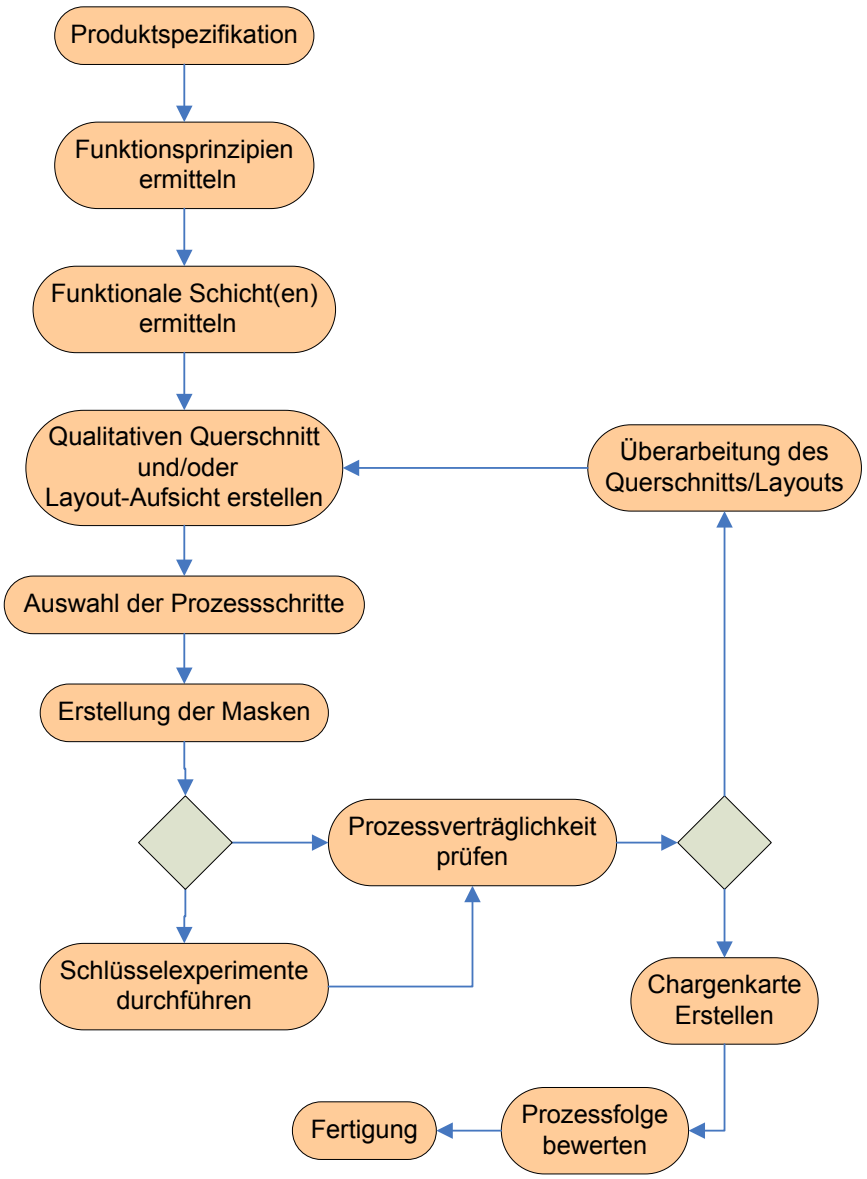


Abbildung 4.9: Aktueller Entwurfsablauf in der Mikrosystemtechnik

4.6.2 Das erweiterte Kreismodell

Das Kreismodell für den fertigungsnahen Mikrosystementwurf wurde bereits Mitte der 1990er Jahre vorgestellt [Brü96, Hah98a] und im Laufe der Jahre am Institut für Mikrosystemtechnik der Universität Siegen zur jetzigen Reife weiterentwickelt. Im Rahmen dieser Arbeit wurde das Kreismodell noch einmal überarbeitet, um die Aspekte aus der Praxisanalyse mit zu berücksichtigen [WH03, WPH03]. Speziell wurde der Aspekt der Maskenerstellung und -modifikation in das Modell eingepflegt. Dieser Aspekt wurde im ursprünglichen Modell nicht berücksichtigt, darf aber nach den Erkenntnissen aus der Praxisanalyse nicht vernachlässigt werden. Der Grund liegt in dem engen Verhältnis von Masken und Prozessschritten. Ändert man beispielsweise die Dauer eines Ätzprozesses, so muss in der Regel auch die Maskierung geändert werden, um Phänomene wie Unterätzen etc. mit zu berücksichtigen. Durch diese Modifikation erhält man ein Entwurfsmodell, das den fertigungsnahen Entwurf in der Mikrosystemtechnik sehr gut abbildet (siehe Abbildung 4.10).

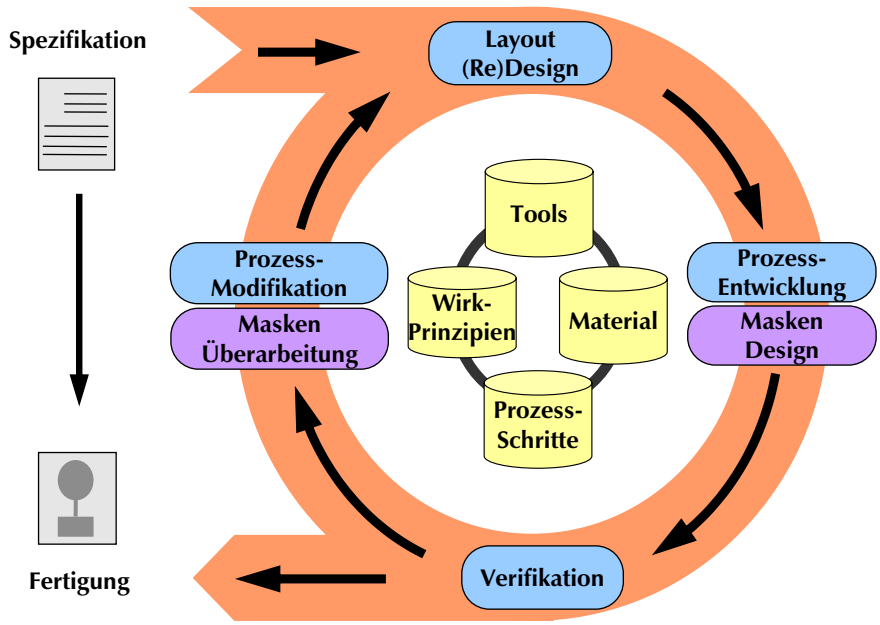


Abbildung 4.10: Erweitertes Kreismodell

Das Kreismodell setzt sich aus vier Phasen des Entwurfs zusammen. Diese Phasen sind Layouterstellung, Prozessfolgen- und Maskenerstellung, Verifi-

kation und Modifikation. Als initiale Eingabe erhält das Kreismodell eine Spezifikation. Die Ausgabe ist eine eindeutige Beschreibung der Fertigung.

Layouterstellung In der Phase Layouterstellung wird das Layout, also die Schichtfolge des Mikrosystems bestimmt. Das Layout kann aus einer dreidimensionalen Schichtbeschreibung bestehen oder sich aus einer Reihe von Aufsichten und Querschnitten zusammensetzen.

Prozessfolgen- und Maskenerstellung Bei der Prozessfolgen- und Maskenerstellung wird für die Layoutbeschreibung eine geeignete Folge von Prozessschritten nebst der evtl. notwendigen Masken zusammengestellt.

Verifikation In der Verifikationsphase wird der Entwurf überprüft. Das heißt, dass kontrolliert wird, ob die gefundene Prozessschrittfolge so fertigbar ist und das initial erstellte Layout als Ergebnis hat.

Modifikation Fällt die Verifikation negativ aus, so muss der Entwurf modifiziert werden. Das kann zum einen heißen, dass das Layout überarbeitet werden muss, oder dass die Prozessfolge bzw. die zugehörigen Masken geändert werden müssen.

Entsprechend dem kybernetischen Entwurfsmodell nach Rammig findet man im Kreismodell erzeugende und überprüfende Entwurfsschritte. Der Übergang von der Layoutphase zur Phase der Prozessschritt- und Maskenerstellung stellt den erzeugenden Schritt dar und soll hier als schichtorientierter Entwurf eingeführt werden.

Schichtorientierter Entwurf Der schichtorientierte Entwurf ist die Entwurfsphase, in der aus einem Layout eine oder mehrere Prozessfolgen nebst der zugehörigen Maskensätze generiert werden. In diesem Zusammenhang spricht man auch von Synthese.

Bei der Betrachtung des schichtorientierten Entwurfs kann man leicht auf den Gedanken kommen, dass es sich dabei nicht um einen erzeugende sondern um eine überprüfenden Aktivität handelt. Allerdings darf man bei der Bestimmung der Aktivitäten nicht außer Acht lassen, dass das Ziel des Mikrosystementwurfs eine Fertigungsanweisung, also eine Prozessfolge nebst Masken ist und nicht eine dreidimensionale Struktur.

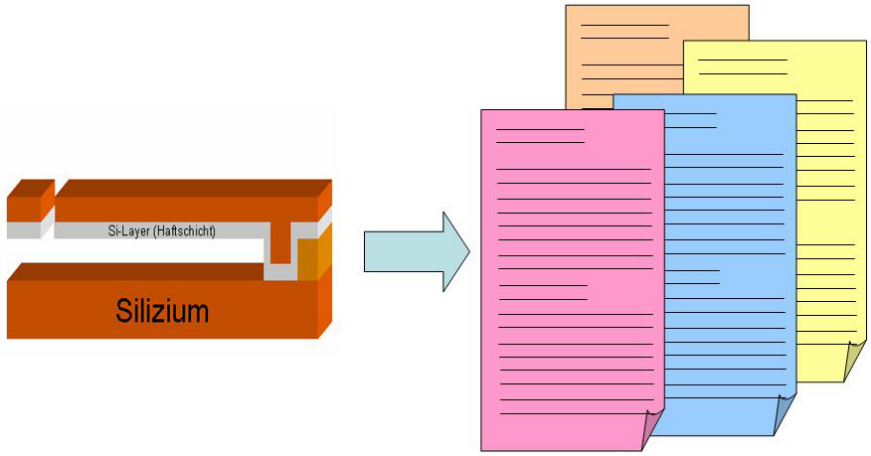


Abbildung 4.11: Schichtorientierter Entwurf

Die Verifikation also die Überprüfung der Prozessfolge gegen das Layout stellt den überprüfenden Schritt dar. Dieser Schritt wird hier als prozessfolgenorientierter Entwurf eingeführt.

Prozessfolgenorientierter Entwurf Der prozessfolgenorientierte Entwurf ist die Entwurfsphase, in der aus einer Prozessfolge und einem zugehörigen Maskensatz ein Layout erstellt wird. Dieser Vorgang wird auch gemeinhin als (Technologie-)Simulation beschrieben.

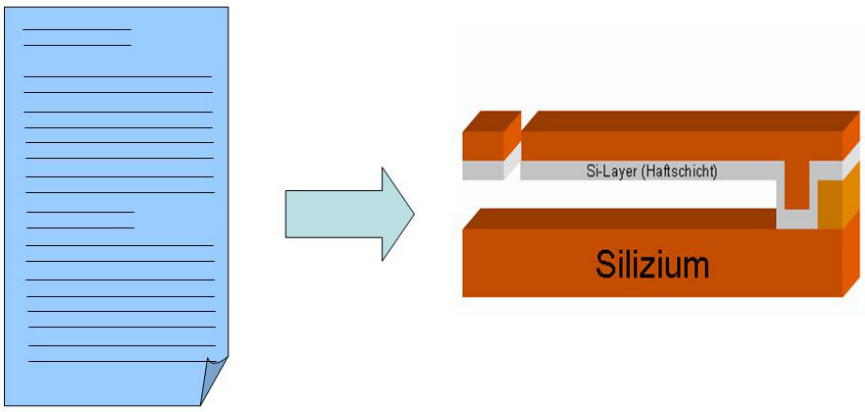


Abbildung 4.12: Prozessfolgenorientierter Entwurf

Für den Entwurf entsprechend des Kreismodells sind einige Voraussetzungen zu erfüllen. Damit der Entwurf gelingt, ist ein umfangreiches Wissen über

die Prozessschritte, die verwendeten Materialien, die Wirkprinzipien und die verfügbaren Werkzeuge notwendig. Im Zentrum des Kreismodells finden sich diese vier Aspekte wieder, die den Entwurf in der Mikrosystemtechnik beeinflussen. Die bildliche Darstellung kann in diesem Fall „wörtlich“ genommen werden. Die Umsetzung des Kreismodells in eine softwaregestützte Entwurfsunterstützung ist ohne die Beachtung dieser zentralen Punkte nicht möglich. Sie werden in den folgenden Kapiteln näher betrachtet. Vorher sollen allerdings vorhandene Entwurfsunterstützungen exemplarisch vorgestellt und deren möglicher Nutzen im Sinne des Kreismodells diskutiert werden. Vorgestellt wird die Software-Suite von Silvaco als Vertreter der kommerziellen Anbieter und die beiden akademischen Systeme MISTIC und LIDO, die als Prototypen bzw. Konzeptstudien angesehen werden können.

4.6.3 Silvaco

Silvaco (**SIL**icon **VA**lley **CO**mpany) wurde 1984 in Santa Clara im Silicon Valley gegründet und hat sich seitdem zu einem der führenden Softwarehäuser für TCAD-Software[1] in der Mikroelektronik entwickelt. Neben TCAD-Lösungen bietet Silvaco noch speziell für den Analog-Entwurf Lösungen im Bereich Verhaltensbeschreibung, Schematic Editoren, Schaltkreissimulatoren auf Spice-Basis, Layout-Editoren und diverse Checkprogramme für den Entwurf an [Sil04]. Hier soll jedoch in erster Linie auf die TCAD-Linie eingegangen werden.

Die TCAD-Schiene und damit der fertigungsnahe Entwurf wird von Silvaco durch die drei Pakete *Virtual Wafer Fab*, *ATHENA* und *ATLAS* unterstützt. Bei ATHENA handelt es sich um einen Prozesssimulator. Mit ATHENA lassen sich einige wichtige Prozessschritttypen wie Dotierung, Deposition und teilweise auch Ätzen simulieren. Die Mächtigkeit des Simulators hängt in erster Linie von den zur Verfügung stehenden Modulen ab. ATHENA an sich bildet nur ein Framework um eine Vielzahl an Simulationsmodulen. Als Eingabe erwartet ATHENA ein sogenanntes Deck-File (Abb. 4.13), das neben den einzelnen zu simulierenden Prozessschritten auch diverse Metadaten über den

[1] TCAD steht für Technology Computer Aided Design und umfasst die Gruppe der Softwaretools, die den Technologieentwurf unterstützen.

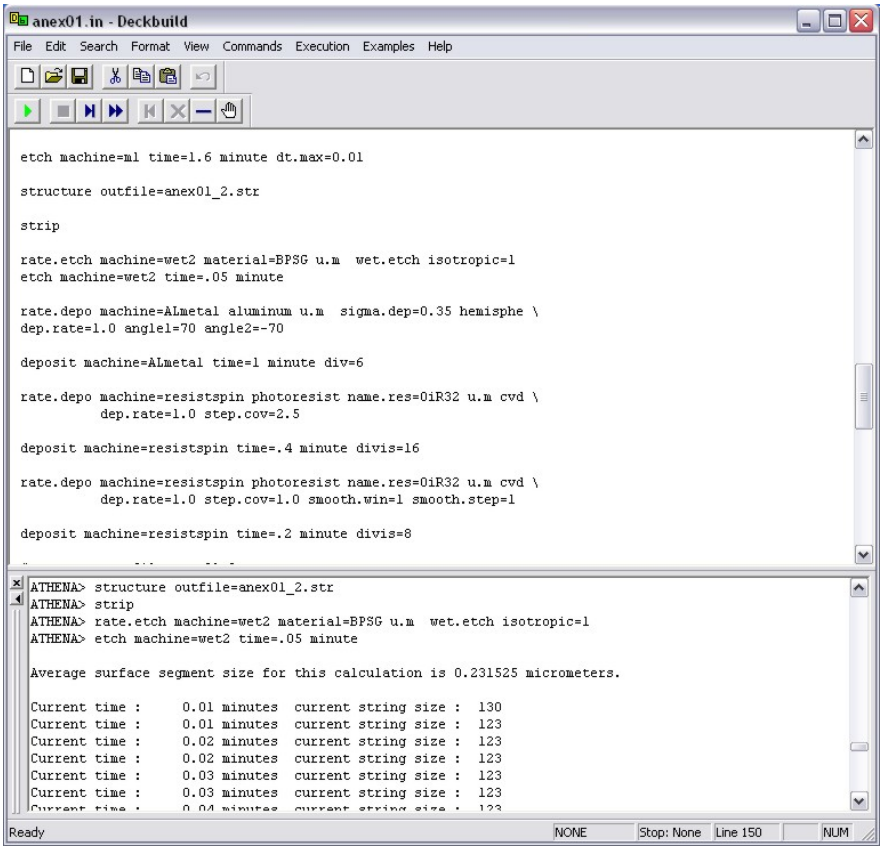


Abbildung 4.13: SILVACO: DeckBuild

Prozessfluss enthält. Das können beispielsweise Angaben über das Gitternetz für die FEM-Simulation sein, Parameter von Schichten, die nicht simuliert werden oder auch die Ausgabedateien. Das Deck-File kann textuell erstellt oder mit einem einfach gehaltenen grafischen Interface zusammengestellt werden. Für die Ansteuerung von ATHENA steht *Deckbuild* zur Verfügung. Deckbuild liest ein Deck-File und gibt es „in Häppchen“ an Athena weiter. Da ATHENA modular aufgebaut ist, können vorhandene Simulationsmodule für einzelne Prozessschritte geladen werden. ATHENA beschränkt sich dabei auf die Simulation der kritischen Prozessschritte, die im Deck-File parametrisiert werden müssen. Die Daten für die Simulationsmodelle sind Veröffentlichungen entnommen und stehen in einer Datei bereit, die die Simulatoren während der Simulation auslesen. Die Referenzen für die einzelnen Modelle können vom Benutzer angezeigt werden. Nach der Simulation eines Prozessflusses mit ATHENA können Aussagen zum Beispiel über die Geometrie, Dotie-

rungsprofile, Stress usw. gemacht werden. ATHENA an sich verfügt über kei-

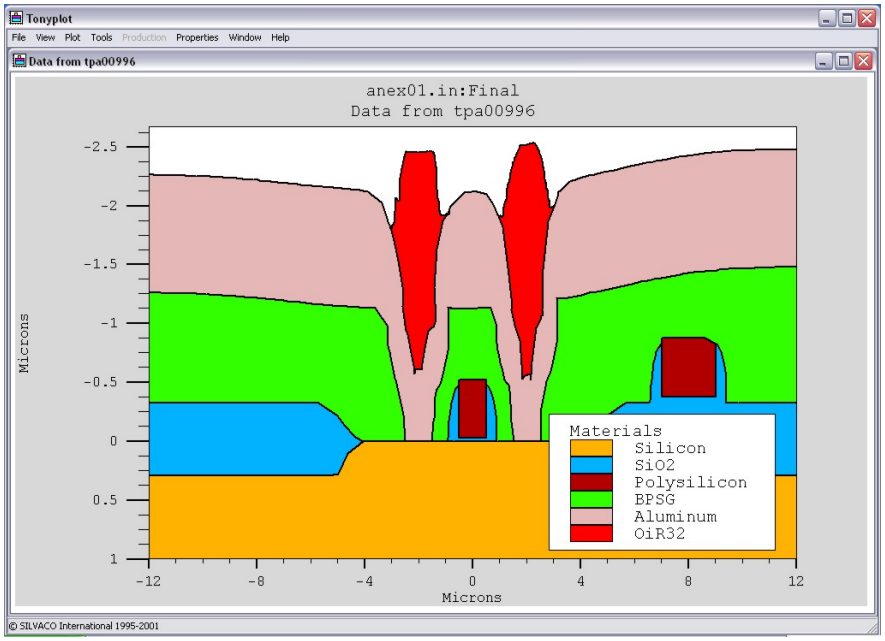
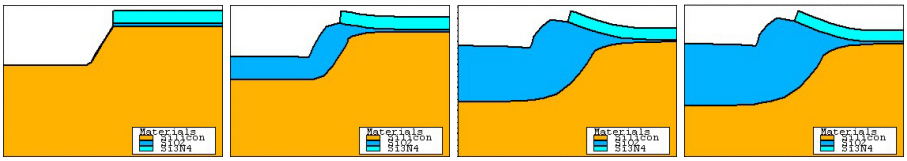


Abbildung 4.14: SILVACO: Tonyplot

ne Möglichkeit der Darstellung der Ergebnisse. Dafür wird ein weiteres Programm benötigt: *Tonyplot* (siehe Abb. 4.14). Tonyplot zeigt die Ergebnisse der Simulation ein- oder zweidimensional an. Eine 3D-Ansicht ist geplant, aber noch nicht verfügbar. Tonyplot stellt eine sehr mächtige Umgebung für die Darstellung dar. So weit die Daten verfügbar sind, kann Tonyplot sowohl diverse Profile anzeigen, aber auch die Simulation als Film abspielen (Abb. 4.15).



(a) (b) (c) (d)

Abbildung 4.15: SILVACO: Simulationsdarstellung als Film

Die Virtual Wafer Fab (VWF) ermöglicht die einfache Durchführung von beispielsweise Parameterstudien. Genau wie Deckbuild stellt VWF nur ein

Framework dar. Für die Simulation greift VWF ebenfalls auf ATHENA zurück. VWF ermöglicht die Aufteilung eines Prozessflusses in beliebig viele Durchläufe mit verschiedenen Parametern oder Prozessschritten. So lassen sich auf einfache Art und Weise beispielsweise die Auswirkung von verschiedenen Drücken auf einen Prozessfluss untersuchen. VWF nimmt dem Ingenieur jedoch keine Arbeit ab. Die Unterschiede der einzelnen Durchläufe müssen manuell eingegeben werden. Auch zeitlich hat man in der Regel keinen Vorteil, es sei denn man verfügt über einen Multiprozessor-Computer und entsprechend viele ATHENA-Lizenzen. So kann man die Funktionalität von VWF wohl am ehesten mit einem vorprogrammierten Batchdurchlauf vergleichen.

ATLAS wiederum ist ein Device-Simulator. Mit ATLAS lassen sich Aussagen über das elektrische, thermische und optische Verhalten eines Mikrosystems machen. Dafür benötigt ATLAS eine genaue Beschreibung des zu untersuchenden Systems. Anders aber als bei ATHENA gibt man hier die fertige Struktur mit den entsprechenden Materialparametern an und nicht die Prozessfolge. Da ATHENA aber genau eine solche Struktur als Ausgabe erzeugt, besteht die Möglichkeit, Ergebnisse direkt aus ATHENA in ATLAS einzulesen. Dann können diverse Parameter extrahiert werden.

Silvaco bietet zum einen die Möglichkeit Prozessflüsse zu simulieren, zum anderen aber auch die Möglichkeit, Strukturen zu untersuchen. Für beides muss allerdings dem Entwickler das Ergebnis mehr oder weniger bekannt sein. Insbesondere muss er sich sehr gut mit den zu simulierenden Prozessschritten auskennen, um das Deck-File zu parametrisieren. Die Benutzerführung an sich zeigt, dass Silvaco schon länger am Markt vertreten ist und die Softwaremodule kontinuierlich entwickelt. Einige Benutzerschnittstellen sind deshalb noch aus den Anfängen der Firma. Ein durchgängiges Bedienkonzept ist darum auch nicht zu erkennen.

Die Silvaco Software eignet sich eher für Prozessfolgenentwickler als für Systementwickler. Letztere können zwar auch mit ATLAS Untersuchungen über ihre gewünschte Struktur anstellen, wie sie aber letztlich gefertigt wird oder ob sie überhaupt gefertigt werden kann, wird nicht hinreichend beantwortet. Die Software besteht in erster Linie aus einer Ansammlung verschiedener Simulatoren. Eine Anpassung der Simulatoren an die eigene Fertigung ist zwar innerhalb bestimmter Grenzen möglich, jedoch nicht standardmäßig vorgesehen. So

kann man die gelieferten Ergebnisse eher als Anhaltspunkte oder Richtlinien als als belastbare Aussagen werten. Die Bedienung verlangt einiges an Einarbeitungszeit. Insbesondere die Parametrisierung der Prozessschritte für die Simulation erfordert die Lektüre des sehr umfassenden Handbuchs. Bereits kleine, aber sehr wirkungsvolle Änderungen, wie die Bereitstellung einer ausgefeilteren grafischen Benutzerschnittstelle, würden die Arbeit mit den Tools um einiges vereinfachen.

4.6.4 MISTIC

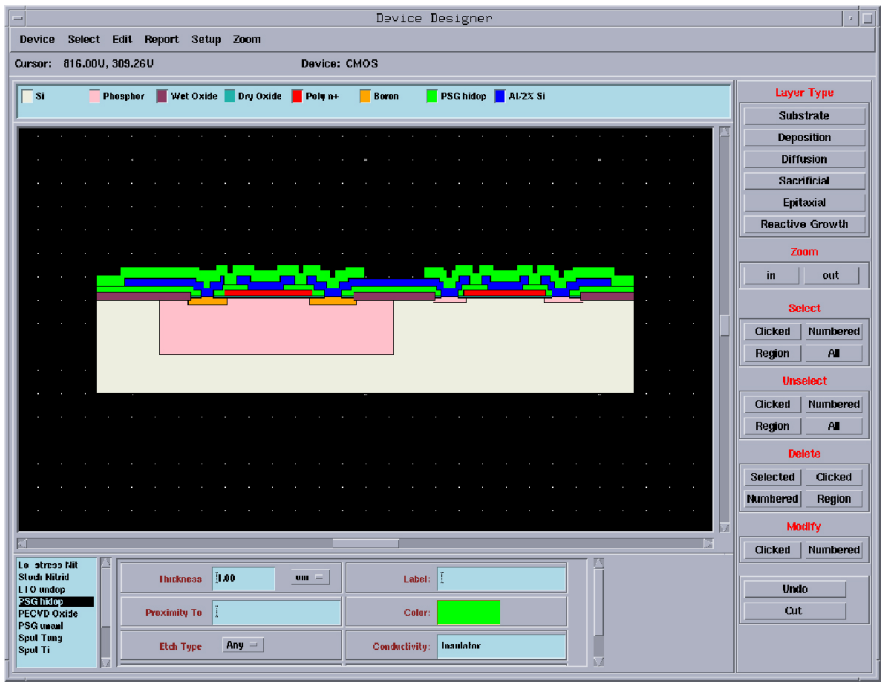


Abbildung 4.16: MISTIC: Schichteneditor

MISTIC (Michigan Synthesis Tools for IC's) wurde an der Universität Michigan Mitte der Neunziger Jahre entwickelt. Es handelt sich um ein viergeteiltes Werkzeug, das automatisch Chargenkarten aus dem Querschnitt eines Objekts für die Dünnschichttechnik und die Mikromechanik erzeugt. Die vier Bestandteile dieser Software sind der Schichteditor (siehe auch Abb. 4.16), die Berechnungseinheit, die Ausgabeeinheit und die Prozessdatenbank. Der Schichteneditor ist die Schnittstelle zum Prozessdesigner. Die Daten für den schichtweisen Querschnittsaufbau entnimmt das System aus einer „Datenbank“. Sie

besteht aus einer Datei, in der die Parameter im Klartext abgelegt sind. Zugriffsbeschränkungen oder Möglichkeiten der Skalierung sind nicht vorgesehen. Beim Schichtaufbau müssen alle Parameter genau spezifiziert werden. Nachträgliche Änderungen sind zwar möglich aber umständlich.

Die Schichten sind nach Prozessen sortiert auswählbar. Sie werden beispielsweise nach Opferschicht, Epitaxieschicht etc. gruppiert. Der Editor lässt nur Rechteck-Strukturen zu. Leider kann das System nur Prozesse bearbeiten, bei denen der Wafer von einer Seite strukturiert wird. Eine Konsistenzprüfung während der Design-Phase wird nicht unterstützt. Wird eine wichtige Schicht vergessen, so fällt dies erstmal nicht auf. Das führt zu hohem und unnötigem Berechnungsaufwand bei der Errechnung der Prozessfolge. Das System berechnet auf Grund der Querschnittsdaten eine oder mehrere passende Prozessfolgen, die als Chargenkarte (siehe Abb. 4.17) ausgegeben werden. Nach welchen Kriterien die Prozessschritte ausgewählt oder verworfen werden, ist nicht ersichtlich. Auch haben Tests gezeigt, dass selbst für einfache Prozesse, bestehend aus Substrat und Oxid, teilweise keine Prozessfolgen gefunden werden. Die für die Berechnung notwendigen Informationen werden aus der „Datenbank“ geholt. Es werden aber auch Parameter errechnet. So z. B. die Temperatur oder die Zeit für die einzelnen Schritte. Für die Ätzungen werden generell die Prozessschritte gewählt, die die höchste Selektivität besitzen. Da der Entwickler nur den Querschnitt angeben muss, werden Implantationen, Ätzschritte oder Lithografieschritte automatisch generiert. Die in MISTIC verwendeten Algorithmen sind in [Zam97] zu finden. Eine ausführliche Beschreibung des Programms, sowie das Programm selbst, ist in [Mas04] zu finden.

4.6.5 LIDO

Das LIDO-System (LIDO ist ein Akronym für **Li**thografiebasierte Design-Tools aus **Do**rtmund) wurde an der Universität Dortmund als automatisches Entwurfsverifikationssystem für lithografiebasierte Mikrotechniken entwickelt [Hah98a]. Im Groben kann man das System in drei Teilsysteme aufteilen. Zum einen der graphische Prozesseditor *LIDO-PEdit* (siehe Abbildung 4.18), der geometrische Entwurfsregelprüfer *LIDO-Check* und für die graphische Ausgabe der Ergebnisse der *LIDO-Manager*. Bei LIDO hat der Entwickler

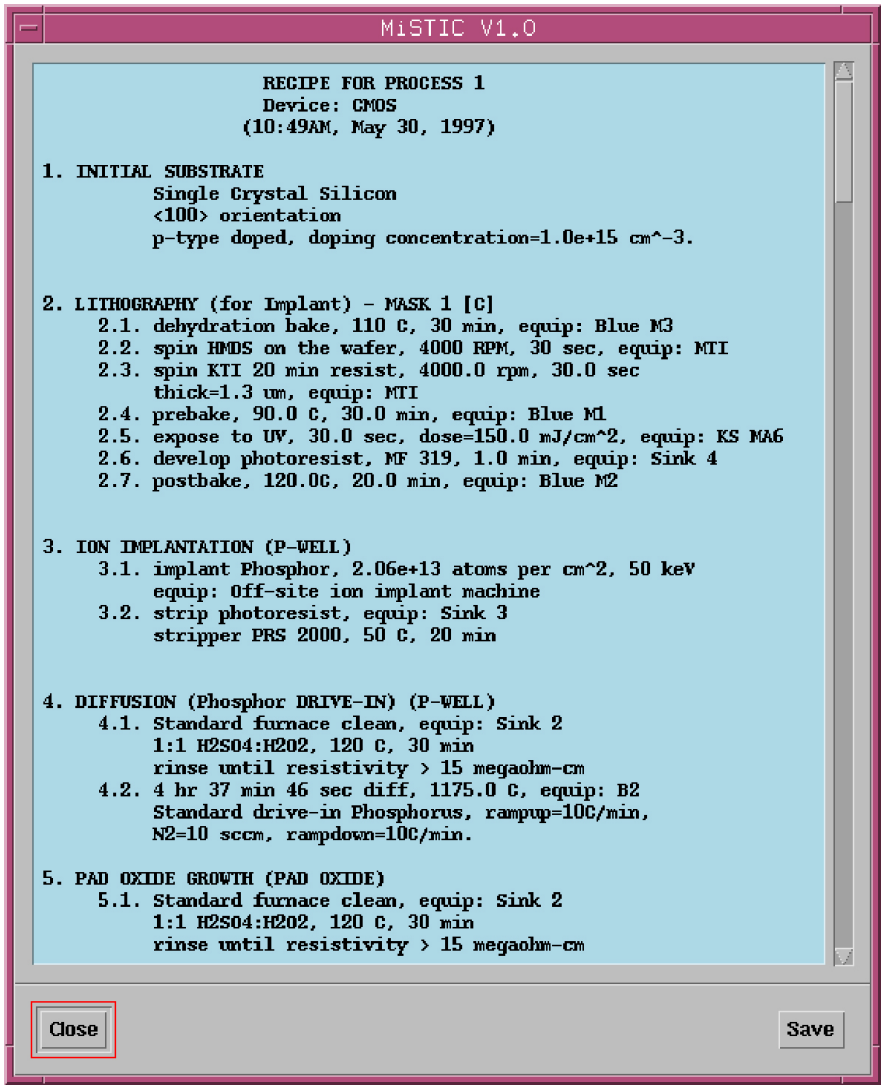


Abbildung 4.17: MISTIC: Chargenkarte

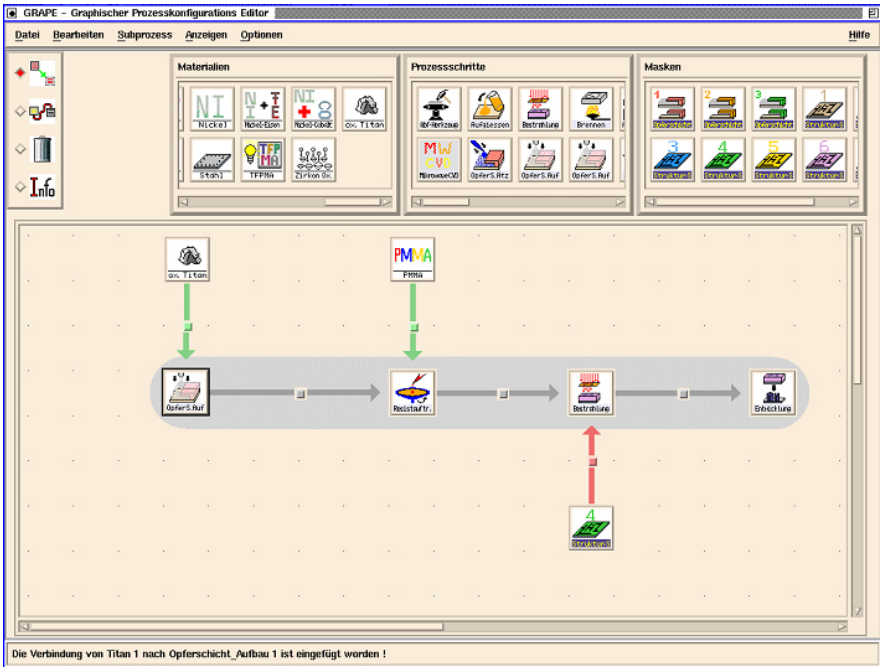


Abbildung 4.18: LIDO: LIDO-PEdit

die Möglichkeit, parallel zu dem Entwurf der Layout-Daten, die im GDSII-Format vorliegen müssen, die Prozessschrittfolge zu entwerfen. Die Daten für den Entwurf der Prozessschrittfolge werden einer Datenbank entnommen. So wird es dem Entwerfer ermöglicht, ein Objekt zu erstellen, auch ohne explizites Fachwissen über die Prozesse zu haben. Die Prozessschritte selbst sind in einer eigens dafür entwickelten, objektorientierten Prozessbeschreibungssprache *„LIDO-PDL“* beschrieben. Eine ausführliche Beschreibung dieser Sprache ist in [Hah98b] zu finden.

Die Prozessschritte und Materialien werden in Form von Piktogrammen zu einer Prozessschrittfolge in LIDO-PEdit graphisch zusammengesetzt (siehe Abbildung 4.18). Die Prozessschritte können hierarchisch zusammengesetzt sein. So lassen sich für den Moment unwichtige Details unter einem Icon verbergen. Dies hat den Vorteil, dass auch komplexe Prozessabläufe einfach verständlich dargestellt werden können. Während des Entwurfs werden die zusammengestellten Prozessschritte auf Konsistenz überprüft, da a priori eine Zusammenstellung beliebiger Piktogramme möglich ist. Eine erfolgreiche Erkennung eines Zyklus ist in Abbildung 4.19 zu sehen. Über eventuell enthaltene Fehler bekommt der Entwerfer eine Rückmeldung. Wenn auch die Layout-Daten

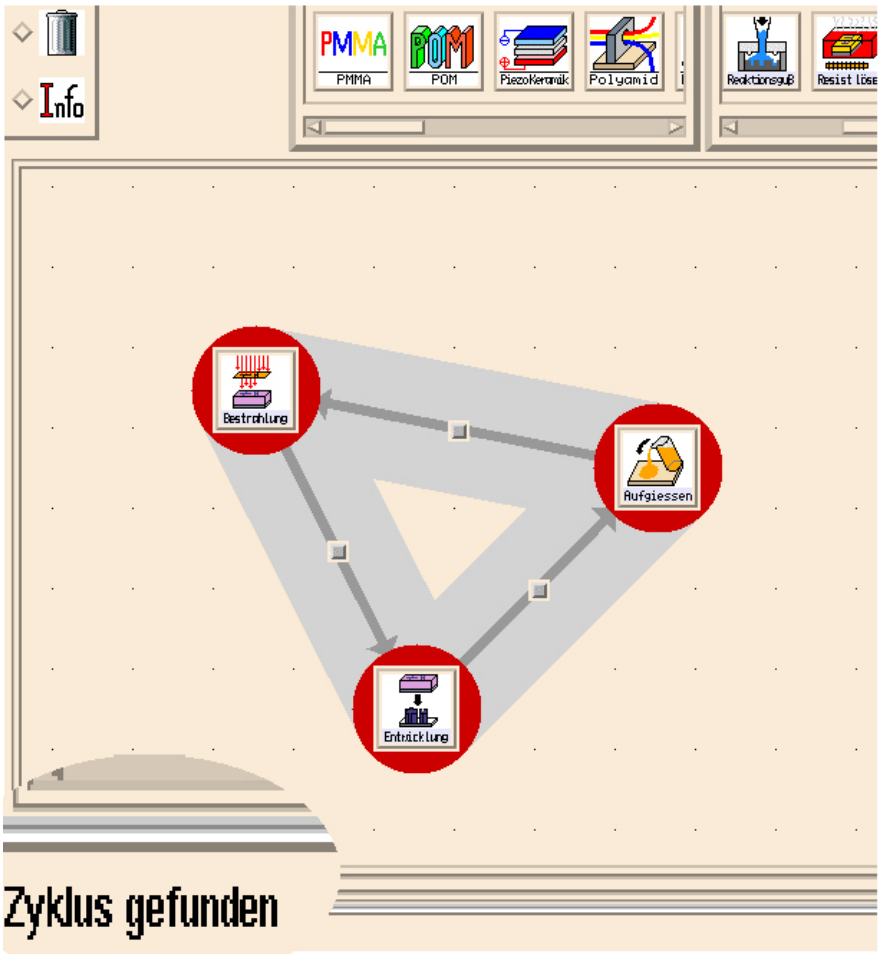


Abbildung 4.19: LIDO: Erkannter Zyklus

vorliegen, werden auch die Layout-Regeln überprüft. Mehr Details des LIDO-Systems sind in [Hah98a] zu finden.

4.7 Fazit

In dem vorliegenden Kapitel wurde der Mikrosystementwurf von der methodischen Seite betrachtet. Es wurde eine mögliche Klassifizierung von Mikrosystemen vorgestellt. Die verschiedenen Ebenen des Entwurfs wurden diskutiert und in die Bereiche verhaltensnah und fertigungsnah aufgeteilt.

Ausgehend von den Methodenbeschreibungen im Kapitel 3 wurde überprüft, ob die Mikroelektronik oder die Mechanik eventuell als Methodenspender für

die Mikrosystemtechnik geeignet ist. Für den Mikrosystementwurf können vorhandene Entwurfsmodelle der Mikroelektronik oder der Mechanik nur bedingt herhalten. Für die Mechanik wurden Beiträge aus der Fachwelt aufgeführt, die eindeutig gegen eine Übernahme der Methoden sprechen. Für die Mikroelektronik wurden ausführlich Gründe gegen eine Adaption diskutiert.

Es wurde das Brezelmodell für den Mikrosystementwurf vorgestellt, das einen guten Überblick über den Entwurfsablauf gibt. Das Brezelmodell lässt sich in verhaltensnahe und fertigungsnahe Bereiche aufteilen. Für den verhaltensnahen Bereich wurde ein detailliertes Entwurfsmodell vorgestellt. Es wurde herausgearbeitet, dass der verhaltensnahe Entwurf nur bedingt einsetzbar ist. Haupteinsatzgebiete sind im Bereich der kommerziellen Mikrosysteme zu suchen.

Für den fertigungsnahen Entwurf wurde eine Praxisanalyse durchgeführt, die zu einer Erweiterung des etablierten Kreismodells geführt hat. Weiterhin wurden die Begriffe des prozessfolgenorientierten und schichtorientierten Entwurfs eingeführt. Eine Untersuchung der vorhandenen Toollandschaft für den fertigungsnahen Entwurf hat gezeigt, dass nur Teilaspekte des Entwurfs unterstützt werden. Mit dem erweiterten Kreismodell ist erstmalig die Möglichkeit geschaffen worden, den fertigungsnahen Entwurf *praxisnah* zu unterstützen. Die verfügbaren Entwurfswerkzeuge unterstützen aber nicht den gesamten Entwurfsablauf sondern in der Regel nur den prozessfolgenorientierten Entwurf (z. B. Silvaco, LIDO), also die Simulation oder nach dem Rammig'schen Modell die überprüfenden Entwurfsschritte. Als einziges schichtorientiertes Werkzeug ist der Prototyp von Mistic zu nennen. Der schichtorientierte Ansatz von Mistic wurde jedoch nicht weiter verfolgt.

Aufgrund des monolithischen Aufbaus der Werkzeuge ist eine Integration derselben zu einem durchgehenden Entwurfsablauf entsprechend dem Kreismodell bestehend aus generierenden und überprüfenden Phasen nicht möglich. Ein weiterer Nachteil der vorgestellten Werkzeuge ist deren Datenhaltung. Bei allen Tools werden ausnahmslos dateibasierte Verfahren eingesetzt. Eine Anpassung oder Erweiterung der Datensätze oder auch der Austausch von Informationen zwischen Werkzeugen oder Standorten ist nur sehr schwer realisierbar.

In den folgenden Kapiteln wird eine Lösung vorgestellt, die die Methodik des

fertigungsnahen Entwurfs entsprechend des erweiterten Kreismodells implementiert.

5 Entwurfsunterstützung

Im letzten Kapitel wurde festgestellt, dass die Entwurfsunterstützung speziell für den fertigungsnahen Entwurf unzureichend ist. Diese Erkenntnis wird gestützt von den Ergebnissen einer Umfrage innerhalb der deutschen Mikrosystem-Industrie, die vom Zentralverband Elektrotechnik- und Elektronikindustrie (ZVEI) durchgeführt wurde [LN04]. Diese Umfrage kommt zu dem Ergebnis, dass zwar Werkzeuge, wie die im letzten Kapitel vorgestellten, eingesetzt werden. Diese Werkzeuge decken den Entwurfsprozess jedoch nur lückenhaft ab, so dass oft händisch gearbeitet werden muss.

Mit dem im vorigen Kapitel vorgestellten erweiterten Kreismodell wurde die Grundlage für eine durchgängige Unterstützung des fertigungsnahen Entwurfs eingeführt. In diesem Kapitel sollen nun die zentralen Unterstützungsmodule des Kreismodells konkretisiert werden.

Für die Unterstützung und erfolgreiche Umsetzung des Kreismodells müssen geeignete Werkzeuge zur Verfügung gestellt werden. Es lassen sich drei Bereiche an Werkzeugunterstützung feststellen. Der erste Bereich ist die Unterstützung bei der Zusammenstellung und Simulation (Verifikation) von Prozessfolgen und Synthese von Schichtfolgen, also eine Unterstützung für den schichtorientierten und prozessfolgenorientierten Entwurf. Die Zusammenstellung und die Simulation werden bereits von einigen Werkzeugen zumindest in Teilen unterstützt. Die Synthese von Schichtfolgen wurde jedoch bis jetzt nur in Form eines Prototypen (MISTIC) realisiert.

Der zweite Bereich betrifft die Überprüfung und Optimierung vorhandener Prozesse. Dieser Bereich wird von den verfügbaren Lösungen nicht abgedeckt. Lediglich die prototypenhafte Implementierung von LIDO bietet eine einfache Überprüfung von Prozessfolgen.

Die Forderung nach einer geeigneten Daten- und Wissensverwaltung bildet den dritten Bereich. Dass die elektronische Haltung der verfügbaren Daten

nicht nur eine Arbeitserleichterung mit sich bringt, sondern auch für die beiden ersten genannten Bereiche der Softwareunterstützung unabdingbar ist, wird in den folgenden Erläuterungen deutlich. Im Folgenden wird ein Überblick über die wichtigsten Aspekte der Entwurfsunterstützung gegeben. Für Details sei auf [Wag01, WPHB02b] verwiesen.

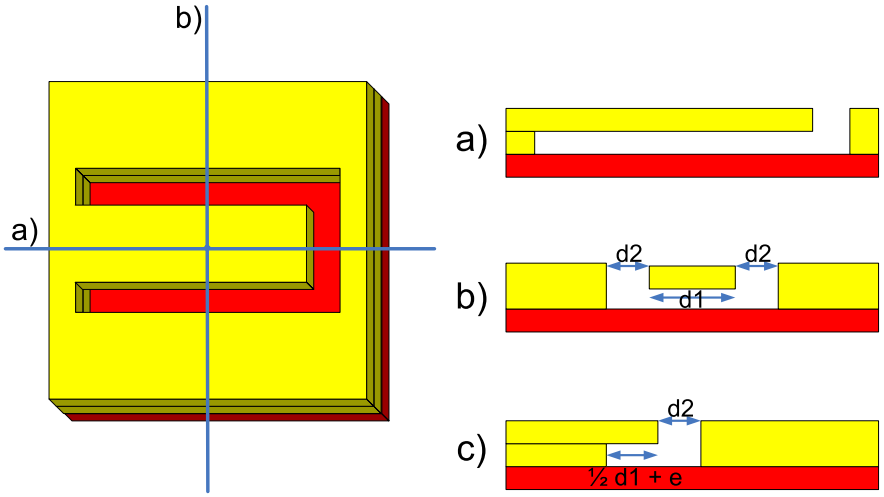
5.1 Charakteristische Querschnitt

Idealerweise werden die Entwürfe für die Mikrostrukturen dreidimensional erstellt. Allerdings haben die Erfahrungen aus der Praxis gezeigt, dass das Erstellen eines dreidimensionalen Entwurfs sehr zeitintensiv ist und wenig unterstützt wird. Üblich ist deshalb eine Kombination von Aufsichten und zweidimensionalen Querschnitten, die während des Entwurfs genutzt werden. Der Verzicht auf die dritte Dimension in den Zeichnungen sorgt dafür, dass diese sehr einfach und schnell modifiziert werden können. Durch die Kombination mehrerer Querschnitte und Aufsichten erhält man de facto die gleichen Informationen, die eine dreidimensionale Darstellung bieten würde. Erst zum Ende, wenn der Entwurf abgeschlossen ist, wird die endgültige Mikrostruktur dreidimensional dargestellt.

Die meisten Werkzeuge im fertigungsnahen Bereich unterstützen allerdings nur den zweidimensionalen Entwurf und arbeiten mit Querschnitten. Um die Eigenheiten einer Mikrostruktur hinreichend zu beschreiben, benötigt man in der Regel mehrere Querschnitte. Die Tools unterstützen aber in der Regel nur einen Querschnitt. Zur Lösung dieses Problems wird im Kontext dieser Arbeit erstmals die Darstellung eines Mikrosystems durch den *charakteristischen Querschnitt* vorgeschlagen.

In Abbildung 5.1a ist ein einfaches dreidimensionales Modell eines Hebels dargestellt. Dieser Hebel ist schmal und lang und nur an der linken Seite fixiert. Für diese Struktur sind zwei Querschnitte angegeben. Querschnitt a) schneidet den Hebel entlang der Längsachse, während Querschnitt b) den Hebel senkrecht der Längsachse schneidet. Betrachtet man nun nur Querschnitt a), so verliert man die Information über die Breite des Hebels. Das gleiche gilt entsprechend für die Länge und die Aufhängung bei Querschnitt b). Möchte man nur anhand des Querschnitts a) die Dauer eines Ätzangriffs bestimmen,

so wird man durch die Länge des Hebels in die Irre geführt. Der Ätzangriff würde zeitlich zu lang dimensioniert, da die Information fehlt, dass ebenfalls von den Seiten geätzt werden kann.



(a) (b)

Abbildung 5.1: Charakteristischer Querschnitt

Querschnitt b) ist für die Dimensionierung des Ätzangriffs besser geeignet, da hier die Breite des Hebels, die in diesem Fall für die Zeitbestimmung relevant ist, in die Betrachtung einfließt. Allerdings entfällt die Information über die eigentliche Intention der Struktur. Weiterhin können viele Werkzeuge mit frei schwebenden Polygonen in den Querschnitten nicht umgehen.

Querschnitt c) zeigt den charakteristischen Querschnitt als mögliche Lösung des Problems. Der charakteristische Querschnitt setzt sich aus mehreren Querschnitten zusammen und vereint die für die Struktur und die Prozessierung relevanten Aspekte. Aus dem Querschnitt a) wird die Information gewonnen, dass es sich um einen Hebel handelt. Querschnitt b) liefert die Breite der Kanäle, die wichtig für die Ätzprozess ist. Weiterhin steuert Querschnitt b) die Breite des Hebels bei. Da die Ätzlösung von drei Seiten den Hebel unterätzen kann, spielt hier der kürzeste Weg die Hauptrolle. Man muss also bei der Berechnung der Ätzzeit nur von der Hälfte der Breite des Hebels ausgehen und einen kleinen Overhead e einkalkulieren, damit die Struktur auch gänzlich un-

terätzt wird. Wäre der Hebel mehr breit als lang, so würde die Länge die dominierende Rolle spielen.

Die Erstellung eines charakteristischen Querschnitts aus einer gegebenen dreidimensionalen Struktur setzt grundlegendes Technologiewissen voraus. Zuerst müssen die charakteristischen Eigenschaften der Struktur durch gezielt gelegte Querschnitte eingefangen werden. Für jede abgebildete Komponente (z. B. Hebel, Graben, Steg, Kaverne) der dreidimensionalen Struktur werden die sich an dieser Komponente kreuzenden Querschnitte betrachtet und die kritischen Abmessungen ermittelt. Diese kritischen Abmessungen lassen sich nur mit Technologiewissen bestimmen. Ein kleines Beispiel soll das verdeutlichen. Für einen langen schmalen Graben ist weniger die Länge als die Schmalheit charakteristisch. Beim Ätzen eines solchen Grabens oder einer Abscheidung innerhalb des Grabens dominiert die Schmalheit die Auswahl und Konfiguration der Technologie. Der charakteristische Querschnitt berücksichtigt hier also die kleinere der beiden gegebenen Abmessungen.

Mit Hilfe des hier vorgestellten Konzepts des charakteristischen Querschnitts ist es möglich, komplexe dreidimensionale Strukturen auf *einen* Querschnitt abzubilden. Eine automatische Erzeugung des charakteristischen Querschnitts ist zur Zeit noch nicht möglich. Auch hängt die Güte des Querschnitts stark vom Fachwissen des Entwicklers ab. Je mehr Wissen vorhanden ist, umso besser kann bei der Erstellung die Technologie berücksichtigt werden. Der charakteristische Querschnitt kann als Eingabe für die Synthese dienen. Es ist allerdings auch sinnvoll bei der Analyse, also der Technologiesimulation einen solchen als Ausgabe zu erzeugen.

5.2 Simulation

Unter dem Begriff der Simulation soll hier die Technologiesimulation verstanden werden. Die Simulation dient, wie bereits im letzten Kapitel erläutert, der Verifikation eines Entwurfs. Dazu wird eine gegebene Prozessfolge simuliert. Das Ergebnis ist in der Regel ein zweidimensionaler Querschnitt durch eine Mikrostruktur. Die Qualität und damit die Aussagekraft einer Simulation hängen stark von der Güte des verwendeten Modells ab.

In der Mikrosystemtechnik trifft man auf verschiedene Methoden für die Simulation der Prozessschritte. Gängig sind Finite-Element-Methoden, Zelluläre Automaten und geometrische Methoden.

Finite-Element-Methode Die Finite-Element-Methode (FEM) ist eine Methode, bei der mittels numerischer Verfahren komplexe partielle Differentialgleichungen gelöst werden. Dabei bedient sich das Verfahren der Diskretisierung und anschließender Approximation zur Lösungsfindung. Mittels der FEM kann das Verhalten fast beliebig komplexer Strukturen sehr realitätsnah simuliert werden, z. B. wenn sie Kraftfeldern oder elektrischen Feldern ausgesetzt werden. Das Problem bei der Simulation von Feldphänomenen ist die Darstellung des Kontinuums. Für eine exakte Darstellung würden Gleichungssysteme mit unendlich vielen Variablen benötigt. Da dies nicht möglich ist, wird das Kontinuum diskretisiert, das heißt, es wird in eine endliche Anzahl Elemente aufgeteilt. Daher auch der Begriff *finite* Elemente. Ein Element ist bei zweidimensionaler Betrachtung ein Polygon und bei dreidimensionaler Sichtweise ein Polyeder mit mindestens drei bzw. vier Knoten. Die Wahl der Elementstruktur ist immer problemabhängig und hat in der Regel direkten Einfluss auf das Ergebnis. In der Praxis sieht diese Diskretisierung so aus, dass eine Gitterstruktur über das Gesamtsystem gelegt wird. Abbildung 5.2 zeigt das Ergebnis einer FEM-Simulation für verschiedene Abscheideprozesse. Details zur Modellierung mittels FEM sind in [ZT00] zu finden. Kommerziell findet die Technologiesimulation mittels FEM Anwendung beispielsweise in den Werkzeugen von Silvaco [Sil04].

Zelluläre Automaten Die Simulation mit zellulären Automaten basiert auf dem Prinzip der Diskretisierung der Struktur und Zeit. Die zu modellierende Struktur wird in eine endliche Anzahl von Zellen aufgeteilt, von denen jede einen diskreten Zustand hat. Der eigentlich kontinuierliche Zeitverlauf wird in Zeitschritte aufgeteilt. Der Zustand einer Zelle kann sich einmal während eines Zeitschritts ändern. Die Zustandsänderung einer Zelle hängt ausschließlich von den Zuständen der Zellen in ihrer Umgebung ab. Das führt zu einer weiteren wichtigen Eigenschaft der zellulären Automaten, nämlich der Lokalität.

Für die Qualität einer Simulation ist neben der Zustandsänderungsfunk-

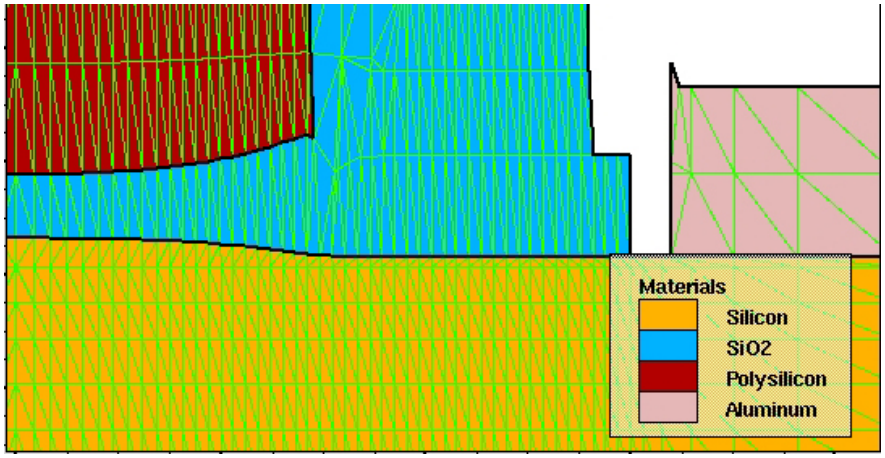


Abbildung 5.2: Simulation mittels FEM

tion auch die Zellularisierung der Struktur von großer Bedeutung. Idealerweise bildet man die Gitterstruktur des Materials nach. Das wird auch in kommerziell verfügbaren Simulatoren gemacht, z. B. wie bei Anise von Intellisense [MHTA98]. Vorteil dieses Vorgehens ist natürlich, dass das Simulationsergebnis sehr realitätsnah ist. Der Nachteil besteht aber darin, dass durch die sehr spezielle Zellanordnung der Simulator nur für eine spezielle Simulation geeignet ist. Allgemeingültiger und sicher auch einfacher zu implementieren sind Zellanordnungen, die auf beispielsweise einer quadratischen oder hexagonalen Struktur basieren. Zwar bilden solche Simulatoren die Realität nicht so genau ab wie solche, die die Gitterstruktur des Materials übernehmen. Dafür sind sie aber für die unterschiedlichsten Materialien einsetzbar [Sch04]. Abbildung 5.3 zeigt als Beispiel das Ergebnis eines Ätzsimulation mittels eines zellulären Automaten und quadratischer Gitterstruktur.

Geometrische Simulation Bei der geometrischen Simulation werden die Vorgänge des zu simulierenden Prozesses mit geometrischen Verfahren bzw. Operationen abgebildet. Beispielsweise lassen sich Depositionsprozesse durch in Richtung des Aufwachsens fortschreitende Ebenen darstellen. Die Aufwachsraten können über die Geschwindigkeit des Ebenenfortschritts simuliert werden. Ähnlich lassen sich Ätzprozesse darstellen. Hier gilt das für die Depositionsprozesse gesagte umgekehrt. Die Grundlagen dieses Verfahrens sind in [Jac62] zu finden. Kommerziell findet die-

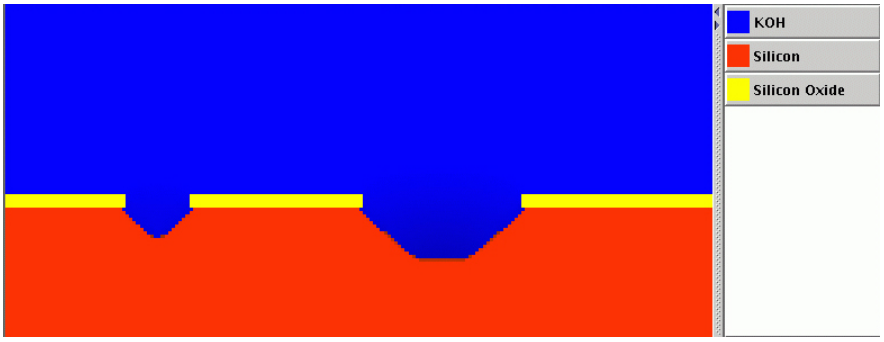


Abbildung 5.3: Simulation mittels Zellulärer Automaten

ses Verfahren z. B. in dem Werkzeug SIMODE [Frü04] Anwendung. In Abbildung 5.4 ist das Ergebnis einer einfachen geometrischen Simulation bestehend aus Depositions- und Ätzschritten zu sehen.

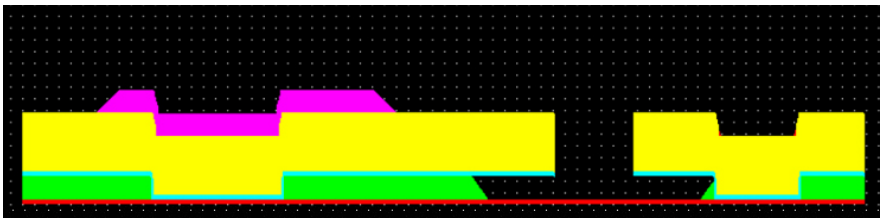


Abbildung 5.4: Geometrische Simulation

5.2.1 Hierarchisierung von Modellen

Idealerweise sollte es für jeden einzelnen Prozessschritt der Mikrosystemtechnik ein angepasstes Simulationsmodell geben. Von diesem Idealzustand ist man allerdings noch weit entfernt. Zum einen liegt das daran, dass die Effekte vieler Prozessschritte noch nicht gänzlich untersucht und verstanden sind. Zum anderen kommen ständig neue Prozessschritte hinzu.

Von Entwicklerseite werden zwei Typen von Prozessmodellen gefordert. Einerseits werden Modelle benötigt, die den Prozessschritt möglichst genau und exakt abbilden können. Andererseits werden Modelle gebraucht, die ein sehr schnelles Ergebnis liefern. Erstere Modelle haben allerdings den Nachteil, dass die Simulation oft sehr zeitaufwendig ist. Letztere simulieren zwar schnell,

dafür müssen aber Abstriche bei der Genauigkeit in Kauf genommen werden. Besonders letztere Modelle dienen einer schnellen Verifikation. Rechtfertigen lassen sich diese Anforderungsunterschiede über den Entwurfsablauf.

Zu Beginn des Entwurfsprozesses ist der Entwurf eher unscharf und noch nicht so elaboriert, wie er zum Ende des Entwurfsprozesses sein sollte. Zu Beginn unterliegt der Entwurf vielen Änderungen. Zum Ende des Entwurfsprozesses hin werden die Änderungen immer geringer. Folglich reichen zu Beginn eines Entwurfsprozesses Simulationsergebnisse zur Überprüfung aus, die das Ergebnis prinzipiell darstellen. Auf zu viel Genauigkeit kann hier zu Gunsten der Geschwindigkeit verzichtet werden. Erst zum Ende des Entwurfsprozess dienen genauere Modelle zur endgültigen Verifikation. Ein solches Vorgehen entspricht auch dem Vorgehen in der Praxis. Zu Beginn dominieren Skizzen und grobe Zeichnungen mit Rechteckstrukturen. Erst zum Ende, wenn ein Großteil des Entwurfs steht, werden detaillierte Zeichnungen erstellt.

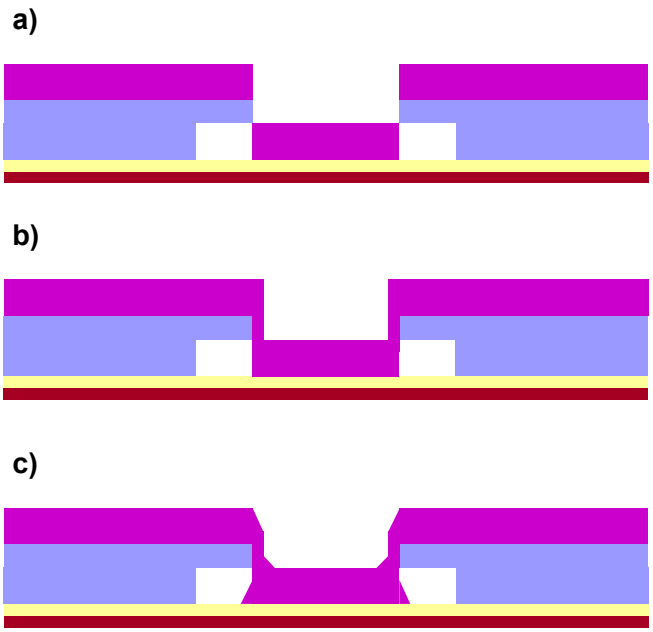


Abbildung 5.5: Hierarchie der Simulationsmodelle am Beispiel der Genauigkeit eines Abscheideprozesses

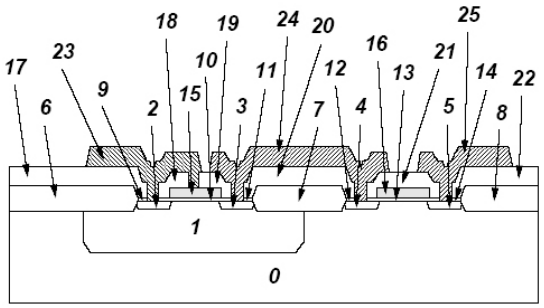
Um die Auswahl immer präziserer Modelle mit dem Fortschreiten des Entwurfs zu ermöglichen, wird hier das Konzept der Hierarchisierung der Modelle erstmals vorgestellt. Abbildung 5.5 zeigt beispielhaft die Auswirkung einer solchen Hierarchisierung. Angenommen, zu Beginn steht nur fest, dass

auf eine bestehende Struktur eine Abscheidung mit bestimmter Dicke vorgenommen werden muss. Welcher Abscheidungsprozess gewählt wird, ist noch unklar. Anstatt nun ein komplexes Modell willkürlich auszuwählen und zeitaufwendig zu simulieren, wählt man ein Modell, dass die Abscheidung prinzipiell darstellt und als Ergebnis beispielsweise Rechteckstrukturen liefert (siehe Abbildung 5.5a). Im Laufe der Entwicklung stehen immer mehr Details des Prozesses fest und ein passenderes Modell kann gewählt werden (Abbildung 5.5b), solange, bis der Prozessschritt endgültig feststeht und damit auch das Modell (Abbildung 5.5c).

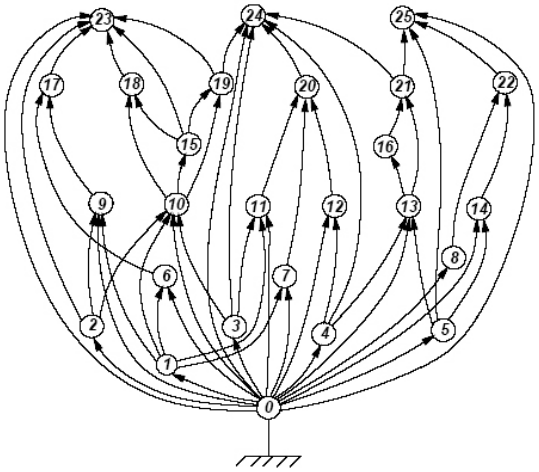
Ebenfalls von Nutzen ist diese Strategie, wenn noch gar kein Modell für den einzusetzenden Prozessschritt existiert. Bis dato kann dann der Prozessfluss in seiner Gesamtheit nicht simuliert werden. Über die Hierarchisierung der Modelle ist es nun aber erstmalig möglich, einen weniger speziellen Prozessschritt, der aber ähnliche Ergebnisse liefert, für eine erste Simulation zu nutzen. Wird beispielsweise ein spezieller LPCVD-Prozess eingesetzt, für den noch kein Modell existiert, so kann für eine erste Verifikation ein Standard-LPCVD-Modell oder lediglich ein allgemeines CVD-Modell verwendet werden. Diese Hierarchisierung hängt eng mit dem Datenhaltungskonzept zusammen, das im folgenden Kapitel vorgestellt wird.

5.3 Synthese

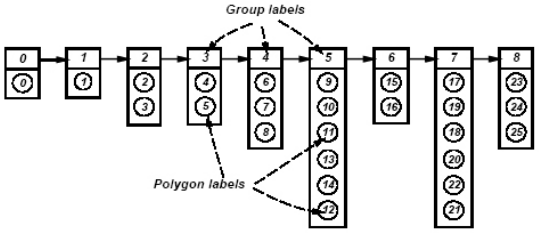
Die technologienahe Synthese in der Mikrosystemtechnik ermittelt für eine gegebene Schichtfolge eine oder mehrere Prozessfolgen. Dazu bedient sich der Synthesealgorithmus aus einem Pool von Prozessschritten und deren Modellen. Im Gegensatz zur Simulation, bei der die Prozessschritte bereits vorgegeben sind und somit die Auswahl der Modelle nicht so schwer fällt, muss bei der Synthese der umgekehrte Weg eingeschlagen werden. Anhand der Schichtfolge, die in der Regel als charakteristischer Querschnitt vorliegt, müssen die verursachenden Effekte nachvollzogen und Prozessschritten zugeordnet werden. Dabei betrachtet man die Schichten nicht separat. Vielmehr muss der Kontext der gesamten Folge in Betracht gezogen werden, da ein Prozessschritt meistens die gesamte Prozessfolge beeinflusst. Kommerzielle Werkzeuge für die Technologiesynthese sind nicht erhältlich. Allerdings wur-



(a)



(b)



(c)

Abbildung 5.6: Dekomposition einer CMOS-Struktur [ZCM99a]

de Mitte der 1990er Jahre an der Universität Michigan MISTIC [Mas04] entwickelt. MISTIC ermöglicht die Synthese von Schichtfolgen auf Basis von Silizium-Dünnschichttechniken. Dazu wurde das Verhalten einer begrenzten Menge an Prozessschritten algorithmisch beschrieben. Diese Beschreibungen dienen dem System als Basis für das Reverse Engineering, denn als solches kann die Technologiesynthese auch gesehen werden.

Als Eingabe dient ein charakteristischer Querschnitt der zu synthetisierenden Struktur. In Abbildung 5.6a ist beispielhaft eine CMOS-Struktur aufgetragen. Die Schichtfolge wird in eine gerichtete Graphdarstellung umgewandelt. Dabei wird jeder einzelne Polygonzug der Struktur zu einem Knoten im Graphen. Die gerichteten Kanten geben die Schichtabfolge wieder. Liegt Schicht a auf Schicht b, so verläuft die gerichtete Kante von b nach a. Die zur Beispielstruktur zugehörige Graphdarstellung ist in Abbildung 5.6b zu sehen. In den folgenden Schritten werden Polygone gleichen Materials gruppiert (siehe Abbildung 5.6c). Den Schichten werden mögliche Prozessschritte zugeordnet und mögliche Ätz- und Lithografieschritte werden eingefügt. Die Prozessparameter werden durch Berechnung der Effekte ermittelt. Durch weitere Berechnungen und Ausschlüsse wird der Ergebnisraum nach und nach verkleinert, bis die möglichen Prozessschritte feststehen.

Je unspezifischer die Angaben zu den einzelnen Schichten zu Beginn der Synthese sind, desto größer ist der anfängliche Ergebnisraum. Damit bleiben mehr Möglichkeiten der Variation der Prozessfolgen, wodurch möglicherweise mehrere Ergebnisse geliefert werden. Diese geben dann allerdings die Möglichkeit der Optimierung. Eine umfassende Darstellung der verwendeten Algorithmen ist in [Zam97, ZCM99a, ZCM99b] zu finden.

5.4 Konsistenzprüfung

Im fertigungsnahen Entwurf lassen sich zwei Bereiche der Verifikation des Entwurfs unterscheiden. Einerseits gibt es die Verifikation anhand der Technologiesimulation, und andererseits die Verifikation mittels eines Konsistenzprüfers. Der Konsistenzprüfer überprüft eine Prozessfolge auf die Einhaltung von Regeln. Das geschieht bereits vor der Technologiesimulation und soll sicherstellen, dass die betroffene Prozessfolge in der gegebenen Form überhaupt

prozessierbar ist und damit die Technologiesimulation ein belastbares Ergebnis liefert. Der Konsistenzprüfer steht also nicht in Konkurrenz zur Technologiesimulation, sondern macht diese überhaupt erst sinnvoll.

Gründe für diese Trennung liegen einmal in der Art der Regeln und in der Mächtigkeit der Simulatoren. Die verfügbaren Simulatoren sind in der Regel Ein-Schritt-Simulatoren. Das heißt, dass sie nur einen einzelnen Prozessschritt simulieren können. Andere Prozessschritte und Schichten werden nur in sehr beschränktem Maß in die Simulation einbezogen. Die Simulatoren agieren sozusagen lokal. Nun ist aber ein Fertigungsprozess in der Regeln nicht lokal eingrenzbar, sondern hat mehr oder weniger Einfluss auf die anderen, bereits vorhandenen Schichten. Über seine eventuell abgeschiedenen Materialien nimmt der Prozessschritt auch auf folgende Prozessschritte Einfluss. Dieser Einfluss ist aber in der Regel in den Simulationsmodellen nicht berücksichtigt. Das liegt daran, dass diese Abhängigkeiten oft nur als Faustregeln oder Wenn-Dann-Regeln vorhanden sind. Solche Regeln können beispielsweise die Haftung verschiedener Materialien aufeinander betreffen, oder aussagen, dass eine Lackschicht nach einem Lithografieschritt auch wieder entfernt werden sollte. Allerdings können sie auch komplexerer Natur sein. Beispielsweise dürfen Gold- und Aluminiummetallisierungen nicht offenliegend in Kontakt mit einer Säure gelangen, da sich dann ein Volta-Element bildet, was wiederum zur Folge hat, dass das unedlere Metall, hier also Aluminium, weggeätzt wird. Solche mehr informellen Regeln, die auf Expertenwissen beruhen, sind sehr schwer in die Simulationsmodelle einzubauen.

Diese Regeln lassen sich aber relativ einfach mit einem Konsistenzprüfer prüfen. Dazu werden die Regeln dem Prozessschritt zugeordnet, der sie einführt. Ein Prozessschritt Lackaufschleudern besitzt dann die Regeln, dass die Lackschicht wieder entfernt werden sollte und, solange die Lackschicht noch nicht entfernt wurde, die Temperatur der Folgeprozessschritte nicht über einen gewissen Wert steigen darf, da sonst die Lackschicht zerstört wird. Unterscheiden lassen sich die Regeln nach Vor- und Nachkonditionierungen, Regeln, die Prozessschritte, Parameter oder Materialien betreffen und bedingte und unbedingte Regeln.

Die Vor- und Nachkonditionierung gibt an, in welche Richtung die Regel sozusagen wirkt. Handelt es sich um eine Vorkonditionierung so schränkt die Regel die Prozessschritte, Parameter oder Materialien ein, die vor der Prozessierung

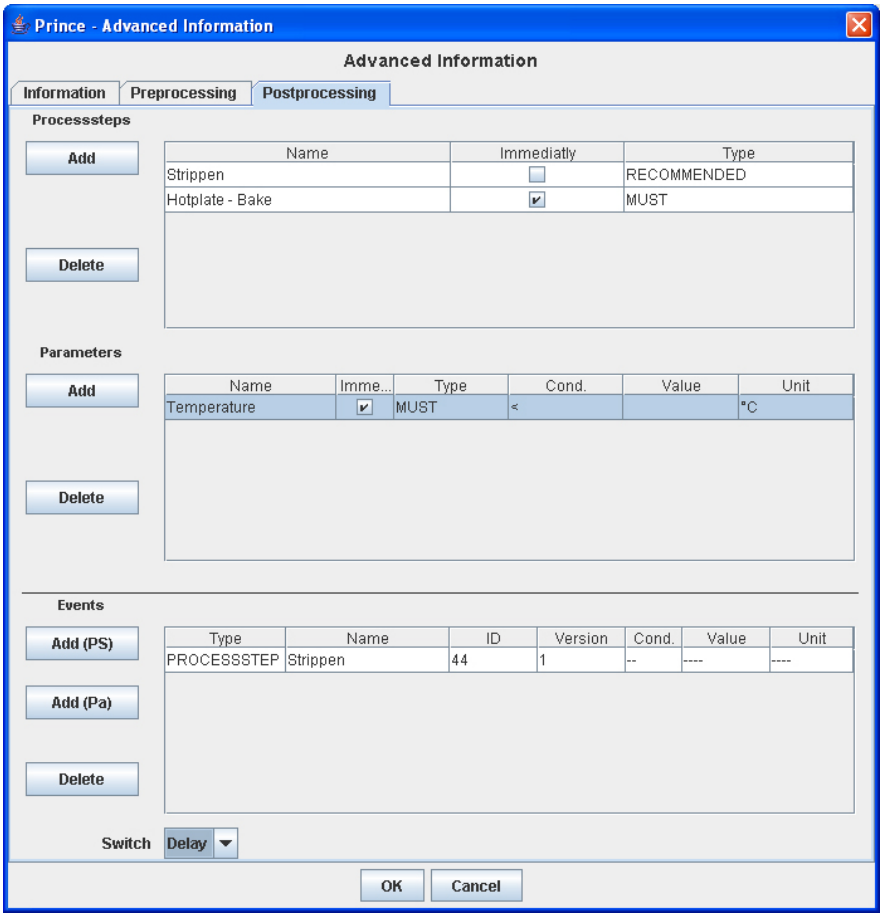


Abbildung 5.7: PRINCE: Vor- und Nachbedingungen

des aktuellen Prozessschritts liegen. Entsprechendes gilt bei der Nachkonditionierung für alle folgenden. Die Regeln lassen sich weiterhin kategorisieren nach unbedingten und bedingten Regeln. Unbedingte Regeln gelten über die gesamte Prozessfolge. Sie werden nicht ungültig im Gegensatz zu bedingten Regeln, die eine Art Gültigkeit haben können. Beispielsweise kann die Regel aufgestellt werden, dass nach dem Aufbringen einer Lackschicht die Temperatur nicht über den Temperaturwert x steigen darf. Dies gilt aber nur solange, wie die Lackschicht noch vorhanden ist. Ist sie bereits entfernt worden, darf die Temperatur höher sein. In Abbildung 5.7 sind beispielhaft ein paar Regeln für einen Prozessschritt angegeben. Alle Regeln lassen sich abschwächen. Standardmäßig gelten die Regeln als auf jeden Fall einzuhaltende. Allerdings kann man die Regeln derart abschwächen, dass ihre Einhaltung nur empfohlen wird.

Das Konzept der Konsistenzprüfung ist nicht neu. Vielmehr wurde es bereits mit dem LIDO-System (siehe vorheriges Kapitel) eingeführt. Neu allerdings ist das Konzept der bedingten Regeln, also die Einführung von Gültigkeiten von Regeln.

Führt man eine Konsistenzprüfung auf einer Prozessfolge durch, so werden die für die einzelnen Prozessschritte gegebenen Regeln der Reihe nach geprüft. Kommt es zu einer Verletzung der Regeln, so wird das dem Entwickler mitgeteilt. Abbildung 5.8 zeigt eine solche Konsistenzprüfung. Die Überprüfung ergab eine Warnung und eine Fehlermeldung. Warnungen resultieren aus Regeln, deren Beachtung empfohlen wird. Im Gegensatz dazu stehen die Regeln, die auf jeden Fall einzuhalten sind. Solche Regeln führen im Fall der Nichtbeachtung zu Fehlermeldungen, was wiederum bedeutet, dass die Prozessfolge in dieser Form nicht fertigbar ist.

5.5 Optimierung

Der erste Schritt vor einer jeden Optimierung ist die Bewertung. Ohne eingehende Bewertung fehlt die Grundlage, um überhaupt die Notwendigkeit einer Optimierung zu sehen. Genauso wie nach unterschiedlichen Kriterien bewertet werden kann, kann auch nach denselben Kriterien optimiert werden. Insbesondere die soeben beschriebene Synthese ermöglicht eine Optimierung, bei-

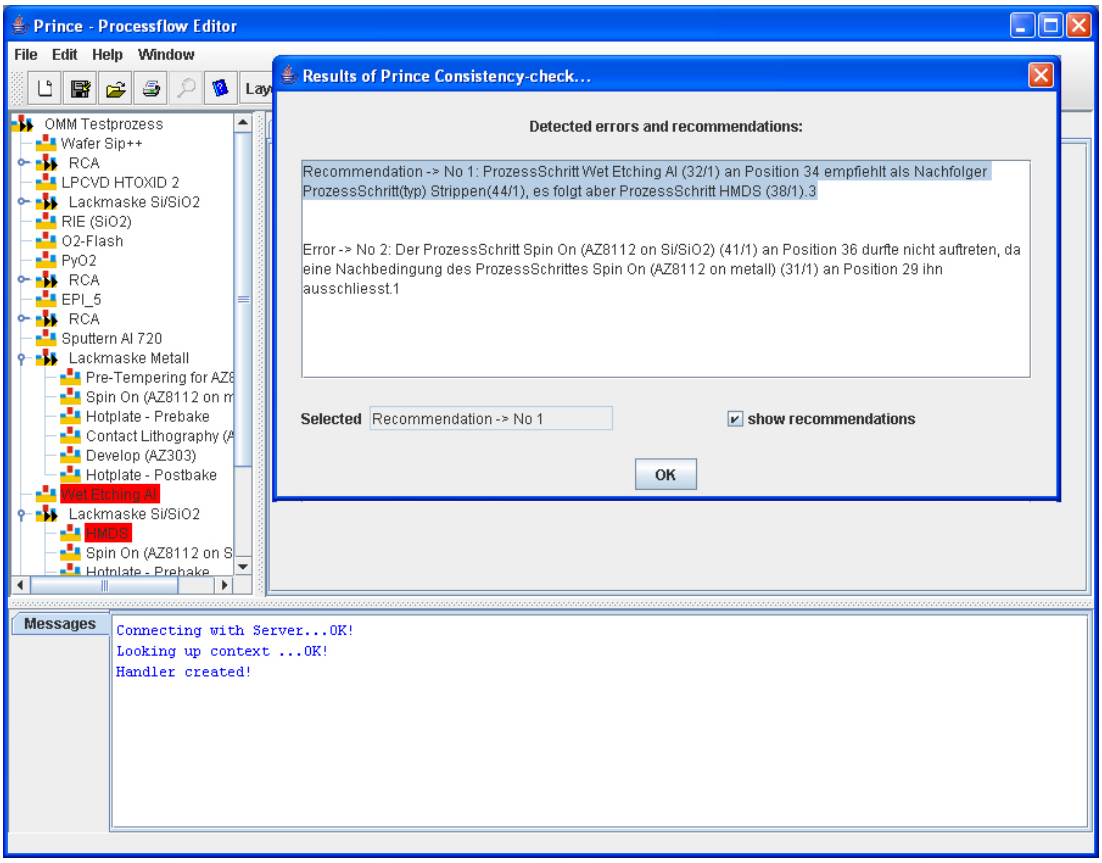


Abbildung 5.8: PRINCE: Konsistenzprüfung

spielsweise, wenn mehrere Ergebnisse vorliegen. Hier besteht die Möglichkeit der Optimierung durch Auswahl des nach den angelegten Kriterien optimalen Prozessflusses.

Eine Optimierung ist allerdings auch während der Synthese möglich. Da in der Regel eine Schicht auf unterschiedliche Art und Weise hergestellt werden kann, muss zwangsläufig eine Auswahl erfolgen, um den Ergebnisraum zu reduzieren. Über die Kriterien der Auswahl lässt sich das Aussehen des Ergebnis steuern.

Optimieren lassen sich nicht nur Prozessfolgen. Auch Prozessschritte lassen Raum für Optimierungen. Jedoch ist hier eine automatische Optimierung in der Regel nicht möglich. Prozessschritte lassen sich einerseits durch Experimente optimieren, die zu neuen Parametern führen, andererseits aber auch durch geschickte direkte Parameterwahl ohne Experiment. Betrachtet man beispielsweise einen Abscheideprozess, so lässt sich oft die Abscheiderate durch eine Reduzierung des Drucks erhöhen. Damit erreicht man, dass der Prozessschritt weniger Zeit in Anspruch nimmt und somit weniger kostet.

5.6 Ergebnisausgabe

Eine Entwurfsunterstützung ist erst dann richtig sinnvoll, wenn die Ergebnisse eines Entwurfs auch nutzbar sind. In der Industrie wird für die Fertigung einer Prozessfolge eine sogenannte *Chargenkarte* oder *Runcard* verwendet. Diese enthält alle Prozessschritte der Prozessfolge inklusive aller fertigungsrelevanten Parameter. Sie kann als Rezept für die Mikrostruktur angesehen werden. Im Zuge dieser Arbeit wird hier das Konzept einer Chargenkarten-Generierung direkt aus dem Entwurf vorgestellt. Bis jetzt war es nicht möglich, einen Prozessfluss direkt in eine Chargenkarte zu übersetzen.

Chargenkarten unterscheiden sich im Layout von Unternehmen zu Unternehmen. Von daher bietet es sich an, die Chargenkarten mit einem generischen Layout aus der Prozessfolge zu erzeugen. Abbildung 5.9 zeigt einen Dialog, der es ermöglicht, die Druckausgabe einer Chargenkarte zu parametrisieren. Es lassen sich die zu druckenden Daten, die Position auf der Seite und die Schriftgröße wählen.

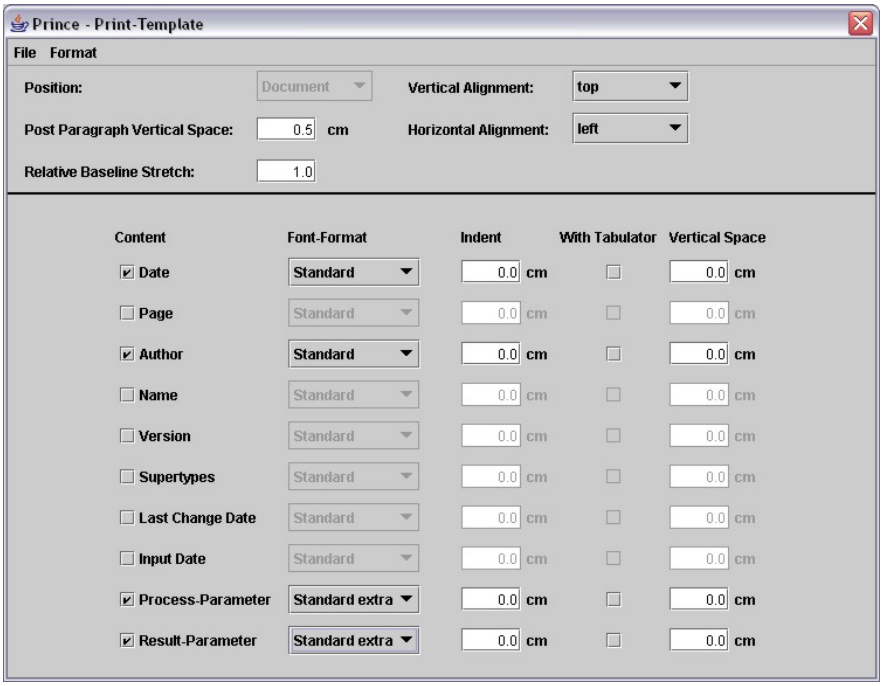


Abbildung 5.9: PRINCE: Konditionierung der Ausgabe

Für Dokumentationszwecke werden auch oft die Ergebnisse der Simulation benötigt. Hier bietet es sich an, die Grafiken in einem gängigen Format zu speichern und den Nutzern zugängig zu machen.

5.7 Zentrale Datenhaltung

Die soeben beschriebenen Module bilden die Werkzeugbasis für eine fertigungsnahe Entwurfsunterstützung in der Mikrosystemtechnik. Um diese Werkzeugmodule jedoch sinnvoll einsetzen zu können und um die Übergänge von einer Entwurfsphase in die nächste zu vereinfachen, fehlen noch die Daten, die die einzelnen Werkzeuge benötigen. Die bisher verfügbaren Werkzeuge (siehe Kapitel 4) sind alle als Solitärsysteme ausgelegt. Das bedeutet, dass ein Zusammenspiel mit anderen Werkzeugen nicht vorgesehen ist. Die gegebenen Modulbeschreibungen zeigen jedoch, dass die unterschiedlichen Module im Grunde alle die gleichen Daten und Informationen brauchen. Deshalb soll hier erstmals das Konzept einer zentralen Datenhaltung eingeführt werden. Die zentrale Datenbasis bildet den Verknüpfungspunkt aller vorgestellter

Werkzeugmodule und ermöglicht dadurch erst einen reibungslosen und unterbrechungsfreien Entwurfsablauf entsprechend dem Kreismodell.

Welche Daten für die Werkzeuge vorgehalten werden müssen, wird ausführlich in [Pop04] diskutiert. Wie sie gesammelt und aufbereitet werden können und auf welche Weise die Datensammlung allein bereits einen Mehrwert für die Mikrosystemtechnik bieten kann, wird im folgenden Kapitel näher beleuchtet.

5.8 Fazit

Die im letzten Kapitel vorgestellte Entwurfsmethodik wurde in diesem Kapitel durch ein geeignetes Unterstützungskonzept ergänzt. Zur Komplexitätsreduzierung wurde das Konzept des charakteristischen Querschnitts eingeführt. Damit ist es erstmals möglich dreidimensionale Strukturen auf *einen* Querschnitt zu reduzieren, ohne die Information über charakteristische und kritische Eigenschaften der Struktur zu verlieren.

Die verschiedenen Möglichkeiten der Technologiesimulation wurden erklärt. Die Technologiesimulation stellt nach dem Rammig'schen Modell einen überprüfenden Schritt dar. Zusätzlich wurde das Konzept der Modellhierarchisierung erstmals vorgeschlagen. Diese Hierarchisierung ermöglicht eine durchgängige Simulation von Prozessfolgen, auch wenn für spezielle Prozessschritte noch keine exakten Modelle vorhanden sind.

Der umgekehrte Weg, also die Synthese einer Schichtfolge als generierende Aktion, wurde am Beispiel der in MISTIC implementierten Methoden dargestellt. Für die Überprüfung von Prozessfolgen auf ihre technologische Machbarkeit wurde eine Konsistenzprüfung vorgeschlagen, die mit Gültigkeiten von Regeln arbeitet und somit das Konzept, das bereits im LIDO-System umgesetzt war, sinnvoll ergänzt.

Den Kern der fertigungsnahen Methodik bildet nicht nur bildlich die Datenhaltung. In [Pop04] wird das Konzept einer zentralen Datenhaltung für die verschiedenen Unterstützungsmodule der einzelnen Entwurfsphasen vorgestellt. Im folgenden Kapitel wird dieses Konzept detailliert erläutert und eine

mögliche Lösung vorgestellt. Anschließend wird mit dem Kapitel 7 die softwaretechnische Grundlage für die Umsetzung der hier vorgestellten Lösungen bereitet. Kapitel 8 schließt schlussendlich den Kreis und zeigt eine Gesamtlösung für die Implementierung des erweiterten Kreismodells.

6 Datenmanagement

Wie schon im letzten Kapitel angedeutet, ist die Datenhaltung im erweiterten Kreismodell ein zentraler Bestandteil. Welche Daten und Informationen vorgehalten werden müssen und wie dies idealerweise bewerkstelligt wird, soll in diesem Kapitel näher betrachtet werden. Dazu werden zuerst einige Konzepte aus der Datenmodellierung vorgestellt, die dann bei der Vorstellung und Modellierung der Daten selbst zur Anwendung kommen. Eine detaillierte Darstellung der Datenmodellierung ist in [Pop04] zu finden.

Anschließend wird analysiert, welche Personengruppen von einer Entwurfsunterstützung profitieren. Es wird erstmals eine Lösung vorgestellt, die die Belange aller Anwendergruppen berücksichtigt.

6.1 Klassifizierung und Vererbung

Bei der Untersuchung der Daten und Informationen, die in der Mikrosystemtechnik vorkommen, steht man schnell vor dem Problem, dass die Menge der Daten unübersichtlich wird. Um dieser Datenflut Herr zu werden und sie sinnvoll nutzen zu können, muss man sie geeignet ordnen. Das Konzept der Objektorientierung aus der Softwaretechnik bedient sich Klassifizierungs-, Abstrahierungs- und Vererbungsstrategien, die sich auch erfolgreich auf die Daten der Mikrosystemtechnik anwenden lassen.

6.1.1 Klassifizierung

Mittels der *Klassifizierung* werden die Datensätze nach Gemeinsamkeiten gruppiert. Beispielsweise kann man Werkstoffe danach klassifizieren, ob sie

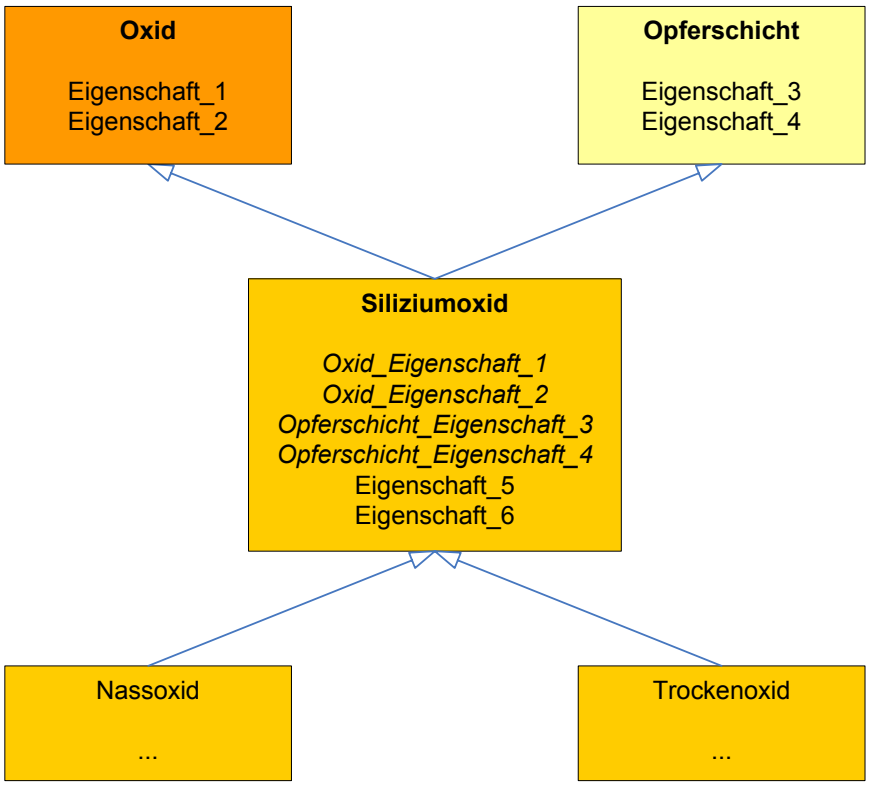


Abbildung 6.1: Vererbung am Beispiel Siliziumoxid

Kunststoffe, Metalle, Oxide, Leiter, Halbleiter, Isolatoren oder als Opferschichten geeignet sind. Man spricht dann von *Klassen* von Objekten, wobei jeder Werkstoff als Objekt gesehen wird. Diese Klassen zeichnen sich dadurch aus, dass ihre Objekte in der Regel gemeinsame Eigenschaften haben. Beispielsweise haben Oxide die Gemeinsamkeit, dass sie das Produkt einer Oxidation sind. Opferschichten haben die Gemeinsamkeit, dass sie geeignet sind, als Opferschicht zu dienen, also besonders gut selektiv entfernt werden können.

Die Klassifizierung lässt sich auch hierarchisieren. Dazu werden die einzelnen Klassen in einer Hierarchie angeordnet. Je tiefer man in dieser Hierarchie steigt, desto spezieller werden die Eigenschaften der Klassen. Als ein Beispiel für die Mikrosystemtechnik kann die Abscheidung herhalten. Die Abscheidung lässt sich spezialisieren in chemische und physikalische Abscheidung, die wiederum in Unterklassen (z. B. APCVD, LPCVD, Sputtern etc.) spezialisiert werden können.

6.1.2 Abstraktion

Zusätzlich zur Klassifizierung werden in [Pop04] abstrakte Klassen von Objekten in der Mikrosystemtechnik erstmalig eingeführt. Abstrakte Klassen sind im Grunde genommen Schablonen für konkrete Klassen. Der Unterschied zwischen abstrakten und konkreten Klassen besteht darin, dass konkrete Klassen auch Objekte haben können, während abstrakte Klassen dies nicht können. Eine konkrete Klasse kann also sowohl Objekte als auch Unterklassen zusammenfassen, während abstrakte Klassen nur Unterklassen enthalten dürfen. Am Beispiel der Mikrosystemtechnik wäre die Abscheidung eine abstrakte Klasse. Möchte man eine Abscheideprozess anlegen, so muss vorher die Entscheidung fallen, ob es sich um physikalische oder chemische Abscheidung handelt. Die chemische und physikalische Abscheidung bilden also Unterklassen der Abscheidung. Objekte, also konkrete Prozessschritte machen auf dieser abstrakten Stufe noch keinen Sinn, da jeder Abscheideprozess eben nach chemischen und physikalischen Verfahren unterschieden werden kann. Somit ließe sich der Prozessschritt in eine der spezielleren Klassen einordnen.

6.1.3 Vererbung

Die Vererbung ermöglicht, wie im richtigen Leben, die Weitergabe von Eigenschaften entlang einer Vererbungshierarchie. An Hand des Beispiels in Abbildung 6.1 soll dies näher erklärt werden. In Abbildung 6.1 sind fünf Klassen in einer Vererbungshierarchie zu sehen. In der Mitte ist die Klasse Siliziumoxid abgebildet. Diese Klasse gehört den (Ober-)Klassen Oxid und Opferschicht an, da Siliziumoxid sowohl ein Oxid ist, sich aber auch sehr gut als Opferschicht eignet. Da sie beiden Klassen angehört, erbt sie deren beider Eigenschaften. Das bedeutet, Siliziumoxid hat die Eigenschaften des Oxids und die der Opferschicht. Diesen Mechanismus nennt man *Mehrfachvererbung*. Mehrfachvererbung bedeutet, dass eine Klasse von mehr als einer anderen Klasse Eigenschaften erben kann. Bei der Mehrfachvererbung ist jedoch Vorsicht geboten. Wenn zwei Oberklassen gleiche Eigenschaften mit unterschiedlichen Werten vererben. Hier müssen geeignete Mechanismen vorgehalten werden, die Regeln zur Kollisionsvermeidung einführen.

Da man das Siliziumoxid nochmals nach Nass- und Trockenoxid unterschei-

den kann, hat die Klasse Siliziumoxid nochmals zwei Unterklassen, die die gesammelten Eigenschaften des Siliziumoxids, damit des Oxids und der Opferschicht, übernehmen. Natürlich kann jede einzelne Klasse noch ihre eigenen Eigenschaften hinzufügen.

Mit Hilfe der Objektorientierung hat man ein mächtiges Hilfsmittel zur Hand, um die Daten übersichtlich zu ordnen. Durch die Strategie der Vererbung erreicht man, dass Informationen nicht mehrmals vorgehalten werden müssen, sondern durch die Klassen propagiert werden. Zu weiteren Überlegungen und Ausführungen zum Paradigma der Objektorientierung sei hier auf [QC94] verwiesen.

6.2 Prozessschritte

Der Prozessschritt stellt die kleinste Einheit der Fertigung dar. Er beschreibt einen eigenständigen Fertigungsschritt. Es gibt eine Vielzahl an Prozessschritten, von denen eine kleine Auswahl bereits in Kapitel 2 vorgestellt wurde. Eben dort werden die Prozessschritte nach ihrem Zweck geordnet vorgestellt. Sie lassen sich aber auch nach anderen Kriterien klassifizieren, wie beispielsweise nach den verwendeten Maschinen, den abgeschiedenen, geätzten oder dotierten Materialien usw. Eine Klassifizierung nach dem Zweck wird bereits in [ZD02, ZD03] vorgestellt. Jedoch wird dort auf eine Abstraktion und einen Vererbungsmechanismus verzichtet. Der Einsatz solcher Mechanismen ist jedoch sehr sinnvoll.

Die Klassifizierung der Prozessschritte geschieht nach gemeinsamen Merkmalen. Folglich sind sich die Prozessschritte entsprechend der Klassifizierungssicht ähnlich. Da sie gemeinsame Merkmale haben, ist es sinnvoll, neben der Klassifizierung auch die Vererbung zu nutzen. Mit [Pop04] wurde das Konzept der Mehrfachvererbung in die Datenhaltung der Mikrosystemtechnik eingeführt. Bezogen auf die Prozessschritte bedeutet das, dass jeder Prozessschritt von beliebig vielen Prozessschritten Eigenschaften erben kann.

Nun kann es vorkommen, dass ein Prozessschritt von zwei weiteren Prozessschritten gleichnamige Eigenschaften mit unterschiedlicher Ausprägung erbt. Beispielsweise könnte das die Prozesstemperatur sein. Der eine Prozessschritt

vererbt als Temperatur das Intervall von 800 bis 1000 Grad und der zweite das Intervall von 700 bis 900 Grad Celsius. In diesem Fall würde sich für den erbenden Prozessschritt die Schnittmenge beider Intervalle, also 800 bis 900 Grad Celsius ergeben. Gibt es eine solche Schnittmenge nicht, oder vererben die Prozessschritte nur einzelne Werte, so wird hier nun das Konzept der nichtrestriktiven Vererbung verfolgt [Pop04]. Das bedeutet, dass der Benutzer die Werte manuell ändern kann, um Kollisionen zu verhindern.

Die Abstraktion bei den Prozessschritten erfolgt über sogenannte Prozessschritttypen. Ein Prozessschritttyp ist eine abstrakte Beschreibung einer Klasse, die nicht als realer Prozessschritt auftreten kann. Der Typ dient dazu, bereits auf hoher Abstraktions- und Hierarchieebene Merkmale der darunterliegenden Prozessschritte festzuschreiben. Das einleitend gegebene Beispiel von der Abscheidung soll hier dazu nochmals aufgegriffen werden. Zwingend notwendig für jeden Abscheideprozess ist eine Abscheiderate. Folglich besitzt der abstrakte Prozessschritttyp Abscheidung die Eigenschaft Abscheiderate. Damit hat über die Vererbung jeder abgeleitete Prozessschritt ebenfalls diese Eigenschaft. Es ist jedoch nicht sinnvoll einen Prozessschritt Abscheidung zu definieren, der als einzige Eigenschaft nur die Abscheiderate hat. Deshalb wird der Prozessschritt abstrakt definiert.

6.2.1 Parameter

Parameter dienen der Konfiguration von Prozessschritten und beschreiben deren Ergebnis. Parameter können sowohl numerischer Natur sein, aber auch aus Textinformationen bestehen. Die Prozesstemperatur, das verwendete Material, die Prozessierungszeit usw. sind z. B. Parameter eines Prozessschritts. Diese lassen sich in Prozessparameter und Ergebnisparameter aufteilen.

Prozessparameter

Prozessparameter bilden die Konfiguration eines Prozessschritts vor und während der Prozessierung ab. Sie geben sozusagen das Rezept für den Prozessschritt an. Grundsätzlich kann nach maschinengebundenen und ungebundenen, zeit- oder ereignisgesteuerten und freien Parametern unterschieden werden.

Maschinengebunden Maschinengebundene Parameter sind solche Parameter, die durch die Eigenart der Maschine fest vorgegeben werden. Das kann beispielsweise der Druck oder auch die Position eines Wafers in der Reaktionskammer sein. Maschinengebundene Parameter sind in der Regel nicht oder nur sehr eingeschränkt änderbar.

Zeitgesteuert Zeitgesteuerte Parameter können sich während der Prozessierung des Prozessschritts abhängig von der Zeit ändern. Beispiele für zeitgesteuerte Parameter sind Gaswechsel zu bestimmten Zeitpunkten oder Temperaturrampen.

Ereignisgesteuert Ereignisgesteuerte Parameter sind ähnlich der zeitgesteuerten Parameter. Der Unterschied liegt darin, dass der Auslöser für einen Zustandswechsel ein bestimmtes Ereignis ist. Beispielsweise kann das Erreichen einer Temperatur zu einer Änderung des Gasflusses führen.

Frei Freie Parameter sind solche, die nicht zu den obigen dreien zu zählen sind.

Eine Kombination der Parametertypen ist grundsätzlich möglich. So kann ein maschinengebundener Parameter auch gleichzeitig ereignisgesteuert sein.

Ergebnisparameter

Ergebnisparameter beschreiben das Ergebnis eines Prozessschritts. Bei den Ergebnisparametern kann man nach Soll- und Istwerten unterscheiden. Sollwerte sind die Werte, die aufgrund der Einstellungen, also der Prozessparameter, zu erwarten sind, während Istwerte die tatsächlich gemessenen Werte darstellen. Diese Unterscheidung ist besonders für die Verifizierung von Modellen notwendig, da mit ihr Differenzen zwischen Modellbildung und Wirklichkeit aufgedeckt werden können.

6.2.2 Bedingungen

Mittels der Bedingungen lässt sich eine Vor- bzw. Nachkonditionierung des Wafers, auf den der Prozessschritt angewendet wird, bewerkstelligen. Bedin-

gungen können Einfluss auf einzelne Parameter aber auch auf komplette Prozessschritte haben. Eine Vorbedingung könnte sein, dass der Wafer vor der Prozessierung gereinigt werden muss. Eine Nachbedingung wäre nach einer Metallisierung, dass die Temperatur bei allen darauf folgenden Prozessschritten den Schmelzpunkt des verwendeten Metalls nicht erreicht.

6.3 Materialien

Da im Vergleich zur Mikroelektronik eine Vielfaches an Materialien in der Mikrosystemtechnik verwendet wird, müssen Informationen zu diesen in geeigneter Weise vorgehalten werden. Als Materialien sollen hier alle vorkommenden Stoffe gelten, also nicht nur Metalle, Halbleiter und Polymere sondern auch Gase, Säuren und andere Stoffe. Damit die Materialinformationen gesucht und gefunden werden können, ist es unerlässlich die Materialien zu klassifizieren. Eine solche Klassifizierung kann nach vielerlei Kriterien geschehen. Denkbar ist eine Klassifizierung nach chemischen oder physischen Gesichtspunkten oder aber auch nach dem Verwendungszweck. Auch das Periodensystem der Elemente stellt eine mögliche Klassifizierung dar. Aus diesem Grund müssen die Materialien mehreren Klassen zugesprochen werden können.

Neben der Klassifizierung interessieren aber auch die Eigenschaften des Materials. Zwar gibt es Eigenschaften, die bei jedem Material angegeben werden können, wie die Dichte oder Gefrier- und Schmelzpunkt, in der Regel besitzt jedes Material aber weitere Eigenschaften. Häufig tritt ein Material auch in verschiedenen Ausprägungen beispielsweise Kristallstruktur auf und weist dann sehr unterschiedliche Eigenschaften auf. Hier ist es sinnvoll, diese Ausprägungen als eigenständiges Material einzuführen und die Zugehörigkeit zu einer Gruppe über die Klassifizierung anzugeben. Am Beispiel des Siliziums könnte so etwas wie folgt aussehen: Es gibt eine Klasse Silizium. Diese Klasse hat Unterklassen z. B. polykristallines Silizium, monokristallines Silizium und amorphes Silizium. Eigenschaften die alle Ausprägungen gemeinsam haben, können über den Mechanismus der Vererbung von der Klasse Silizium an die Unterklassen weitergegeben werden [Sch04].

6.4 Effekte

Der Schlüssel zur Mikrosystemtechnik ist das Wissen über die Effekte. Ohne Kenntnisse über das, was eigentlich bei der Durchführung eines Prozessschritts passiert, bringen die gesammelten Daten über Prozessschritte und Materialien wenig bis gar nichts. Die Effekte beschreiben, was bei der Prozessierung passiert. Das können beispielsweise Haftungseffekte, Ätzungen, Abscheidungen oder Dotierungen sein. Da eine Vielzahl von Effekten denkbar und auch von Nutzen ist, bietet sich hier wieder eine Klassifizierung mit Vererbung an. Zu einem Effekt gehören mindestens ein, in der Regel aber mehr Materialien. Dabei nimmt jedes Material eine bestimmte Rolle ein. Bei einem Ätzeffekt, der das Ätzen von Silizium durch KOH beschreibt, gibt es beispielsweise die Rolle des Ätzmittels, die Rolle des Substrats und des Reaktionsprodukts. KOH wäre also das Ätzmittel und Silizium das Substrat. Auf die Reaktionsprodukte soll hier nicht weiter eingegangen werden.

Die Effekte besitzen natürlich auch Eigenschaften. Eigenschaften können beispielsweise Ätzraten, Haft-Koeffizienten oder Aufwachsraten sein. Nicht selten lassen sich die Effekte und deren Eigenschaften nicht einfach über Konstanten beschreiben. Die Ätzrate hängt in der Regel nicht nur von den beteiligten Materialien sondern auch vom Umgebungsdruck und der Temperatur ab. Diese Zusammenhänge gilt es idealerweise berechenbar, also analytisch, darzustellen. Insbesondere im Hinblick auf eine Unterstützung der Synthese und Verifikation ist es notwendig, die Effekte analytisch oder zumindest in Look-up-Tabellen, die auf empirischen Ergebnissen aufbauen, zur Verfügung zu stellen.

In Abbildung 6.2 ist ein vereinfachtes Datendiagramm zu sehen, das die Beziehungen der einzelnen Komponenten untereinander verdeutlicht. Die Pfeile beschreiben hier eine *hat-* oder *besteht aus*-Beziehung. Im Zentrum steht der Prozessschritt. Er kann beliebig viele Prozessparameter und Ergebnisparameter besitzen, die wiederum Werte besitzen. Jedem Prozessschritt können Bedingungen zugeordnet werden. Effekte beziehen sich auf Materialien und Prozessschritte und können selbst auch Parameter, wie beispielsweise Ätzraten, haben. Die Prozessschritte können selbst genutzt werden, um eine Prozessfolge zu erstellen, die, zusammen mit einem Maskensatz, zu einer Komponente

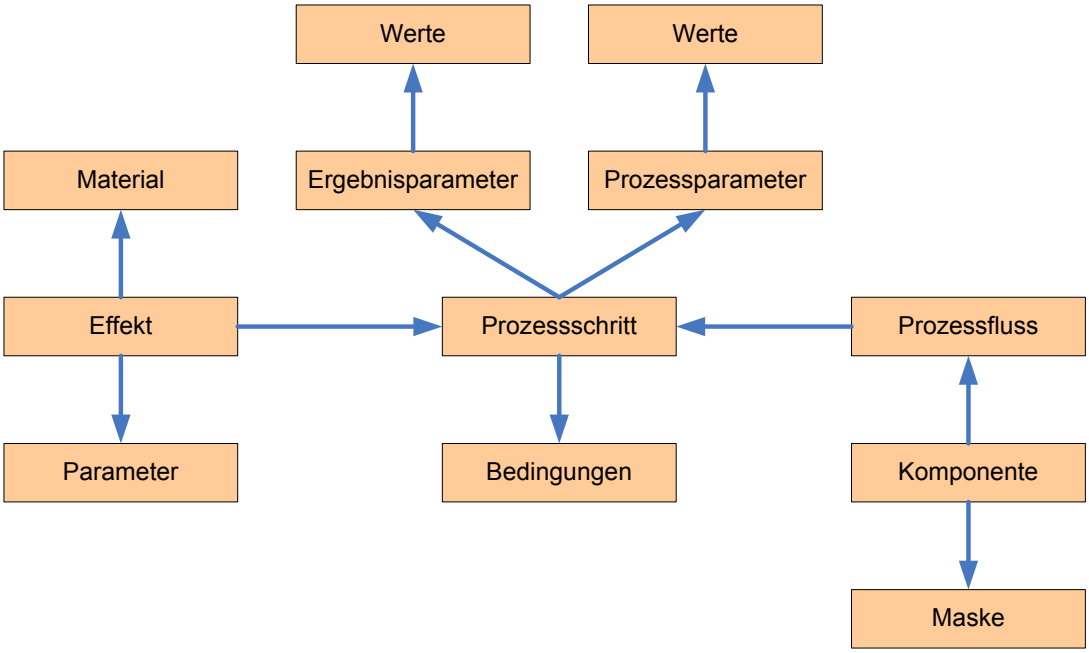


Abbildung 6.2: Vereinfachtes Datendiagramm

gehören. Eine detailliertere Darstellung der Modellierung ist in [Pop04] zu finden.

6.5 Informationsgewinnung

Die soeben aufgeführten Typen von Daten stellen eine Auswahl der wichtigsten Daten dar, die für eine Entwurfsunterstützung in der Mikrosystemtechnik zu berücksichtigen sind. Die Daten allein reichen jedoch nicht aus. Sie müssen interpretiert werden, um von Nutzen zu sein. Mittels Abbildung 6.3 sollen die Zusammenhänge zwischen den Daten und der Informationsgewinnung dargestellt werden. Abbildung 6.3 stellt die Informationsgewinnung als Pyramide dar, deren Basis Daten und Parameter bilden. Die Daten und Parameter können in Beziehung zu einander gestellt werden. Auf diese Weise erhält man Modelle, die beispielsweise die bereits eingeführten Effekte beschreiben. Über die Interpretation dieser Modelle kann man wiederum Wissen erhalten. Beispielsweise könnte über die Interpretation eines Ätzmodells das Wissen er-

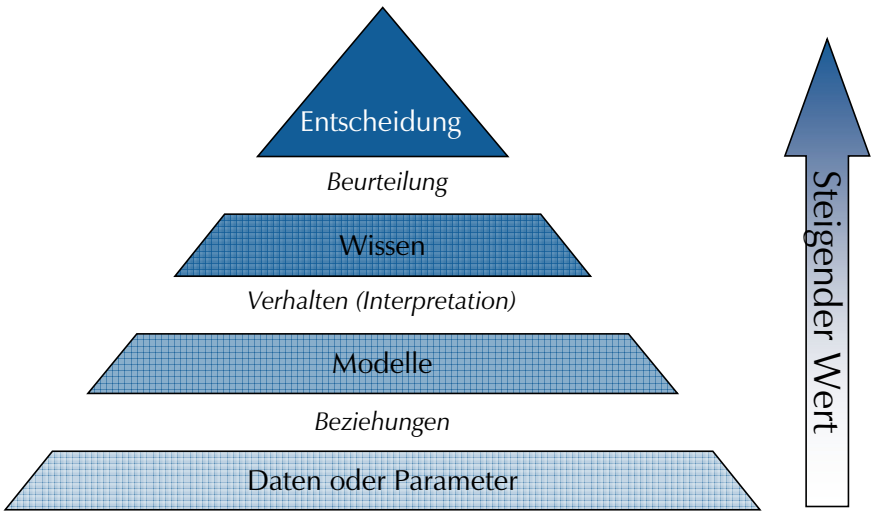


Abbildung 6.3: Wertepyramide der Information nach [Ull02]

langt werden, wie groß die Unterätzrate ausfällt. Dieses Wissen erlaubt dann eine Beurteilung, die schließlich zu einer Entscheidung führt.

Um eine Entscheidung während des Entwurfsprozesses zu fällen, muss genügend Wissen über die Fertigungsprozesse vorhanden sein. Um dieses Wissen zu erlangen, braucht man Modelle, die wiederum auf Daten und Parametern fußen. Unbestritten ist, dass mit der steigenden Anzahl an Daten und Parametern die Modellbildung verbessert wird. Das verdeutlicht, wie wichtig eine große Datenbasis ist, die das Fundament jeder Entscheidung sein sollte.

6.6 Nutzergruppen

Die Mikrosystemtechnik ist ein sehr dynamische Domäne. Fortlaufend wird sie mit neuen Varianten von Prozessschritten bereichert. Zu bereits bestehenden Prozessschritten erlangt man neue Erkenntnisse. Diese Neuerungen sind wichtig für die Informationsgewinnung. Sie können zu Verbesserungen der Modelle und damit zu fundierteren Entscheidungen führen. Würde man jedoch immer erst auf die Veröffentlichung der Ergebnisse auf Konferenzen oder Fachmagazinen warten, so verginge viel wertvolle Zeit für die Entwicklungen. Deshalb soll hier der Ansatz verfolgt werden, die Informationen direkt von der Quelle aufzunehmen. Die Quelle bilden hier die am Entwurf beteiligten

Personengruppen. Das sind die Service-Prozessierer, Einzelprozessentwickler und Prozessfolgenentwickler. Natürlich kann eine Person auch Mitglied von mehreren Gruppen sein. Sie werden im Folgenden vorgestellt. Weiterhin wird aufgezeigt, wie und welche Daten und Informationen die jeweiligen Personen festhalten.

6.6.1 Service-Prozessierer

Der Service-Prozessierer ist meist Spezialist für eine oder mehrere Maschinen, die er betreut. Er kennt sich mit den Eigenheiten der Maschinen aus und weiß, welche Einstellungen zu welchen Ergebnissen führen. Er bietet den Service an, etablierte Prozessschritte durchzuführen und die Ergebnisse zu messen. Bekommt er beispielsweise den Auftrag eine 100nm Schicht zu erstellen, so kennt er die Parametrisierung der Maschine, weiß, welche Vor- und Nachkonditionierung notwendig ist, und kann das Ergebnis anschließend verifizieren. Für die Dokumentation der Prozessschritte und deren Parametrisierung wird ein Laborbuch benutzt.

6.6.2 Einzelprozessentwickler

Die Aufgabe des Einzelprozessentwicklers besteht darin, Prozessschritte neu zu entwickeln oder vorhandene zu optimieren. Dies geschieht in der Regel sehr technologienah. Das bedeutet, dass viel Zeit im Reinraum mit der praktischen Umsetzung der Technologie verbracht wird. Angewiesen ist der Einzelprozessentwickler häufig auf Ergebnisse der Service-Prozessierer, die Vor- oder Nachbereitung für gerade zu entwickelnden Prozessschritte durchführen. Der Einzelprozessentwickler ist Spezialist auf seinem Gebiet der Prozessierung.

Der Einzelprozessentwickler benutzt für die Aufzeichungen seiner Ergebnisse häufig ein Laborbuch. Das Laborbuch ist ein Sammelsurium von informellen Notizen und formalen Prozessdaten. Es dient gleichzeitig als Nachschlagewerk und Dokumentation der Arbeit. In der Regel wird das Laborbuch in Papierform geführt. Teilweise sind auch Eigenlösungen auf Basis von Stan-

dardbüroanwendungen wie Textverarbeitung und Tabellenkalkulation eingeführt worden.

Im Laborbuch des Einzelprozessentwicklers finden sich u. a. die Informationen zu den Prozessschritten wieder. Das sind Einträge zu der verwendeten Maschine und deren Parametrisierung wie Laufzeit, Druck und Temperatur, das Material, gemessene und beobachtete Ergebnisse, besondere Auffälligkeiten und die Vor- und Nachkonditionierung des Wafers. Dies Auflistung soll nur einen Überblick geben und stellt keinen Anspruch auf Vollständigkeit.

Die Aufzeichnungen im Laborbuch dienen nicht nur der Dokumentation und somit der Reproduzierbarkeit der Arbeit. Mit Hilfe der Aufzeichnungen im Laborbuch werden auch statistische Auswertungen durchgeführt. Diese Auswertungen können unterschiedlicher Natur sein. Neben Untersuchungen zu Parameterschwankungen oder Ausbeute können solche Auswertungen auch zur Entscheidungsfindung bezüglich der nächsten Experimente dienen.

Die festgehaltenen Daten zur Parametrisierung des Prozessschritts und die dazugehörigen Ergebnisse dienen wiederum dazu, um Modelle für diesen Prozessschritt zu entwickeln. Je mehr Daten für die Modellbildung zur Verfügung stehen, desto genauer kann diese durchgeführt werden.

6.6.3 Prozessfolgenentwickler

Die Entwicklung von Prozessfolgen zur Fertigung einer Schichtstruktur obliegt dem Prozessfolgenentwickler. Die Struktur ergibt sich aus der Analyse der Anforderungsbeschreibung. In der Regel besteht die Schichtstruktur aus mehr als einer Schicht. Die Aufgabe des Prozessfolgenentwicklers ist die Umsetzung der Schichtfolge in eine Prozessfolge. Dazu greift er auf die Daten zurück, die der Einzelprozessentwickler zu den einzelnen Prozessschritten zur Verfügung stellt. Für jede Schicht muss er einen oder mehrere Prozessschritte finden, mit denen sie gefertigt werden kann. Anschließend müssen die Prozessschritte für die einzelnen Schichten zu einer Gesamtprozessfolge zusammengeführt werden. Dabei kommt es häufig zu Problemen, da Abhängigkeiten und Unverträglichkeiten zwischen den einzelnen Prozessschritten auftreten können. Um diese Abhängigkeiten oder Un-

verträglichkeiten zu erkennen, braucht der Prozessfolgenentwickler ein umfangreiches Technologiewissen und viel Erfahrung.

Seine Dokumentation erledigt der Prozessfolgenentwickler wie der Einzelprozessentwickler mittels des Laborbuchs. Hier trägt er in ähnlicher Weise wie der Einzelprozessentwickler die parametrisierten Prozessschritte seiner Prozessfolge ein und notiert Ergebnisse und Beobachtungen dazu. Der Arbeitsbereich des Prozessfolgenentwicklers ist weniger technologielastig als beim Einzelprozessentwickler. Bevor eine Prozessfolge prozessiert wird, wird sie am Schreibtisch entwickelt und mit Hilfe der verfügbaren Daten parametrisiert. Falls Simulatoren für die jeweiligen Prozessschritte verfügbar sind, wird die Folge zuerst simuliert. Insbesondere bei innovativen Technologien ist die Modellbildung jedoch noch nicht so weit entwickelt, so dass hier für eine Verifikation des Entwurfs derselbe gefertigt werden muss.

6.7 Das elektronische Laborbuch

Die weiter oben beschriebenen Daten bilden die Basis für die Entwurfsunterstützung. Ohne sie ist ein Entwurf nicht möglich. Folglich müssen Mittel und Wege gefunden werden, um diese Daten den Entwicklern und den Werkzeugen in aufbereiteter Form zur Verfügung zu stellen. Dazu müssen die Daten allerdings erst einmal gesammelt werden.

Zu diesem Zweck soll hier das Konzept des *elektronischen Laborbuchs* eingeführt werden. Wie bereits gezeigt, nutzen die am Entwurfsprozess beteiligten Personengruppen für ihre Aufzeichnungen in der Regel physische Laborbücher oder proprietäre elektronische Eigenlösungen. Andererseits sind sie alle auf Daten und Informationen, die von Dritten erhoben wurden, angewiesen. Die Informationsgewinnung aus den bis dato verwendeten Lösungen gestaltet sich besonders schwierig. Es herrscht kein Konsens darüber, was aufgezeichnet wird, noch darüber, wie es aufgezeichnet wird. So kommt es sehr häufig vor, dass Entwicklungsarbeiten in einem Unternehmen, ja sogar in einer Abteilung doppelt und dreifach durchgeführt werden, weil nicht bekannt ist, dass Informationen dazu schon vorhanden oder vorhandene Informationen zu lückenhaft sind [Wag01]. Letzteres liegt häufig daran, dass den Technologieexperten bzw. Experten im Allgemeinen nicht zwangsläufig bewusst

ist, dass bestimmte Sachverhalte, die sie für selbstverständlich und damit für nicht erwähnenswert halten, zu ihrem Expertenwissen und nicht zum Allgemeinwissen gehören. Mehr zur Problematik des Sammelns von Expertenwissen ist in [Lug01, Caw03] zu finden.

Damit ein solches elektronisches Laborbuch Erfolg hat, also von der Zielgruppe eingesetzt wird, darf es die gewohnten Arbeits- und Dokumentationsabläufe nicht einschränken, sondern muss sie vielmehr möglichst übernehmen. Weiterhin hat sich in der Praxis gezeigt, dass mit einer reine Umsetzung des physischen Laborbuchs in ein elektronische Version den Nutzern nicht geholfen ist. Die elektronische Version muss einen direkten Mehrwert gegenüber der bisherigen Lösung bieten. Eine akzeptable Lösung bringt also einen vernachlässigbaren Mehraufwand mit sich und bietet einen Mehrwert gegenüber der physischen Version an. Andernfalls sind viele Entwickler nicht zu einem Umschwenken zu bewegen.

Zusätzlich zum Argument des Mehraufwands kommt noch ein weit gewichtigeres, wenn nicht gar das schwerwiegenste. Viele zukünftige Nutzer scheuen sich vor Systemen, die ihre Daten und ihr Spezialwissen sammeln und anderen Nutzern zur Verfügung stellen. Hier spielt die Angst vor dem Verlust des Expertenstatus und damit die Angst vor Ersetzbarkeit eine bedeutende Rolle. Es gilt also eine Gratwanderung zu bewältigen, in der das elektronische Laborbuch sowohl einen Mehrwert gegenüber den alten Dokumentationssystemen bietet als auch die Souveränität der Experten nicht untergräbt.

6.7.1 Datenhaltung

Prozessschritte

Damit das elektronische Laborbuch sowohl für die Anwender als auch für eine Entwurfsunterstützung von Nutzen ist, muss es die Möglichkeit bieten, die anfallenden Daten geeignet aufzunehmen. Dazu wurde eine Analyse der anfallenden Daten in den Laborbüchern der unterschiedlichen Nutzergruppen durchgeführt. Ein Ergebnis dieser Analyse ist ein Formular zur geordneten Erfassung der Prozessschritte. Die Erfassung der Prozessschritte ist sinnvoll, um vor der Umsetzung einer Datenhaltung die möglichen Datensätze zu sichten. Abbildung 6.4 zeigt eine verkürzte Version des Erfassungsformulars, das

Einzelprozess: **Oxidation**

Datum: **16.10.2001**
Bearbeiter (Autor): **Wagener, Andreas, USI/RS**

Prozesstyp: **Schichterzeugung**
Maschinentyp: **Ofen**

Kurzbeschreibung:
Erzeugen von Oxiden
Dokumentation:
<Wo ist der Einzelprozess beschrieben? (Literatur, Berichte etc.)>

Alternativprozesse:
<Hier bitte Einzelprozesse nennen, die alternativ zu diesem verwendet werden können>

Prozessparameter (PP)

Nr.	Parameter	Größe	Manuell regelbar	Beeinflusst durch
1	Temperatur	900 – 1100°C		
2	Sauerstofffluss	[l, sccm]		
3	Wasserstofffluss	[l, sccm]		
4	Prozesszeit	>20 min – 25 h		
5	Druck	>= 1 bar		
6	Temperzeit (eventuell)	< 1h		

Ergebnisparameter (EP)

Nr.	Parameter	Größe	Priorität	PP-Einfluss	EP-Einfluss
1	Schichtdicke	20 – 3000 nm		4	
2	Brechungsindex	1,46 ± 5%			
3	Stress (Druck)	-250 – 350 MPa		1	
4	Flankenverhalten				

Zusammenhänge/Redundanzen

- Je höher die Temperatur, desto höher die Oxidationskonstante
- Je dichter die Schicht, desto geringer die Oxidationskonstante
- Je niedriger der Si-Anteil, desto geringer die Oxidationskonstante
- Je höher die Temperatur, desto niedriger der Stress

Faustregeln: (informelle Aussagen)

Waferkonditionierung

Vorher	Nachher
Wafermaterial	Temperaturbudget beeinflusst Diffussionsgradient
RCA (+HF)-Reinigung direkt	Stress ändert sich bei nachfolgender Temperung oder Prozessen mit $T > T_{Abscheidung}$
Kein Metall oder Organik	

Abbildung 6.4: Formular zur Erfassung von Prozessschrittdaten

bereits mit einigen Beispieldaten gefüllt ist. Am Beispiel eines einfachen Oxidationsprozesses soll es kurz vorgestellt werden.

Im oberen Bereich sind die allgemeinen Informationen zu dem Prozessschritt zu finden, wie der Name, der Autor und das Erstellungsdatum. Weiter kann hier eine Eingruppierung nach Prozesstypen erfolgen und die notwendigen Maschinen angegeben werden. Die Kurzbeschreibung dient der Übersicht, während unter dem Punkt Dokumentation Literaturangaben oder anderweitige Dokumentation zu finden ist. Über die Angabe von Alternativprozessen besteht die Möglichkeit eine Alternativauswahl anzubieten. Prozessparameter werden in einer Tabelle aufgelistet. Wichtige Angaben zu den Pozessparametern sind die möglichen Werte nebst Einheiten. Weiterhin kann bestimmt werden, ob der Parameter manuell regelbar oder durch die Maschinen vorgegeben und von welchen anderen Parametern er beeinflusst wird.

Ähnliches gilt für die Ergebnisparameter. Betrachtet man beispielsweise den Parameter Schichtdicke, so stellt man fest, dass er von dem Prozessparameter Prozesszeit abhängt, was auch einleuchtend ist. Über die Angabe der Priorität lässt sich festlegen, wie wichtig diese Informationen sind. Diese Angabe dient in erster Linie den Messungen, also welche Messung als erstes durchgeführt werden sollte. Zusammenhänge, die noch nicht formal beschrieben werden können, werden unter der Rubrik Zusammenhänge beschrieben. Hier trifft man in der Regel auf *je-desto*-Regeln. Noch unpräzisere Regeln, sogenannte Faustregeln, beschreiben Phänomene, die noch unerklärt sind, wie beispielsweise absichtliche Verunreinigungen von Flüssigkeiten zur Erlangung eines Ergebnisses etc. Die Waferkonditionierung letztlich gibt die Vor- und Nachkonditionierung des Prozessschritts an. Die ausführlichen Ergebnisse dieser Analyse sind in [Pop04] zu finden. Sie sind direkt in die Entwicklung des elektronischen Laborbuchs eingeflossen.

Die in Abbildung 6.5 dargestellte Maske des entwickelten elektronischen Laborbuchs ermöglicht den Benutzern die Eingabe und Verwaltung von Prozessschritten. Die Prozessschritte werden der Übersichtlichkeit halber in einer Baumstruktur entsprechend der bereits vorgestellten Hierarchisierung dargestellt. Durch die Navigation durch die Baumstruktur gelangt man relativ schnell zu jedem Prozessschritt. Jeder Prozessschritt kann auf Grund der Mehrfachvererbung in unterschiedlichen Ästen des Baumes auftreten und wird dort auch angezeigt. Es handelt sich jedoch jeweils immer um denselben

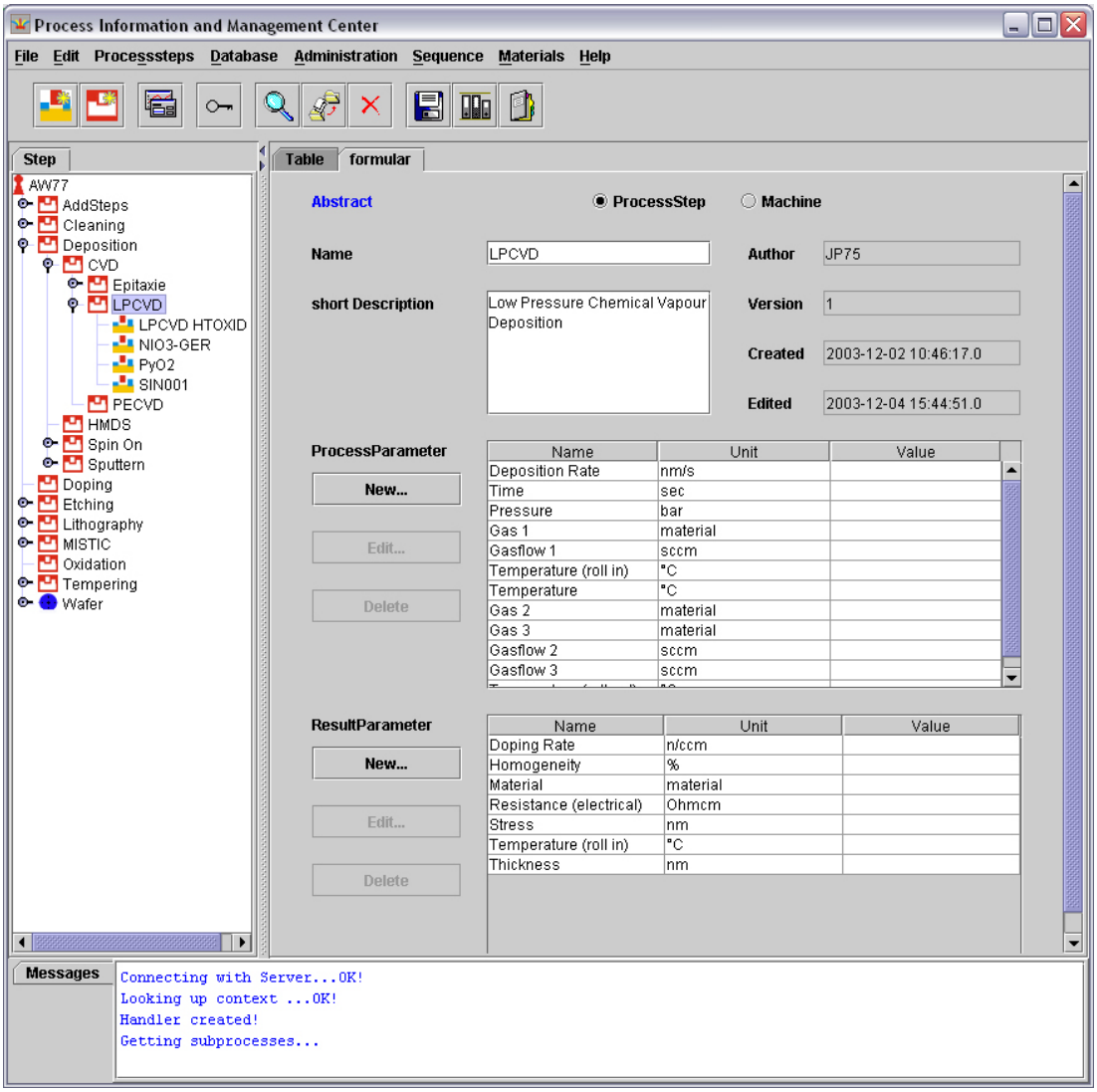


Abbildung 6.5: Elektronisches Laborbuch: PRINCE: Prozessschritteditor

Prozessschritt. Wird ein neuer Prozessschritt angelegt, so geschieht das direkt in der Baumstruktur. Der neue Prozessschritt erbt automatisch die Eigenschaften, also die Parameter, Bedingungen etc. von seinem Vaterprozessschritt. Da die Vererbung der Parameter nicht restriktiv ist, können die Werte jederzeit geändert werden. Durch die automatische Übernahme wird dem Benutzer einerseits viel Arbeit abgenommen und andererseits können die geerbten Parameterwerte auch als Vorschlag für neue Prozesschritte genutzt werden.

Zu jedem Prozessschritt können neben den Prozess- und Ergebnisparametern, die in den beiden unteren Tabellen dargestellt werden, auch Verwaltungsdaten festgehalten werden. Verwaltungsdaten umfassen alle Informationen, die für die Prozessierung (also die Technologie) nicht relevant, jedoch für die Archivierung und Zuordnung nötig sind. Unter Verwaltungsdaten sind folgende Eintragungen relevant:

Eindeutiger Name Der eindeutige Name dient der Benennung und einfachen Referenzierung.

Autor Die Angabe des Autors dient dazu, dass der Urheber nachvollzogen werden und für eventuelle Nachfragen zur Verfügung stehen kann.

Erstellungsdatum Das Erstellungsdatum gibt an, wann der Prozessschritt erstmals eingetragen wurde.

Änderungsdatum Das Änderungsdatum gibt den Zeitpunkt der letzten Änderung an.

Beschreibung Die Beschreibung dient der textuellen Beschreibung des Prozessschritts.

Zusätzliche beschreibende Dokumente, wie Fotos, Grafiken, Publikationen, Ausgaben von Maschinen oder alte Aufzeichnungen lassen sich über ein Dokumentenmanagement verwalten. Zu jedem Prozessschritt können beliebig viele Dokumente angegeben werden, wie in Abbildung 6.6 zu sehen ist.

Parameter

Die Eingabe der Parameter zu einem Prozessschritt erfolgt getrennt nach Prozess- und Ergebnisparametern entsprechend der weiter oben vorgestell-

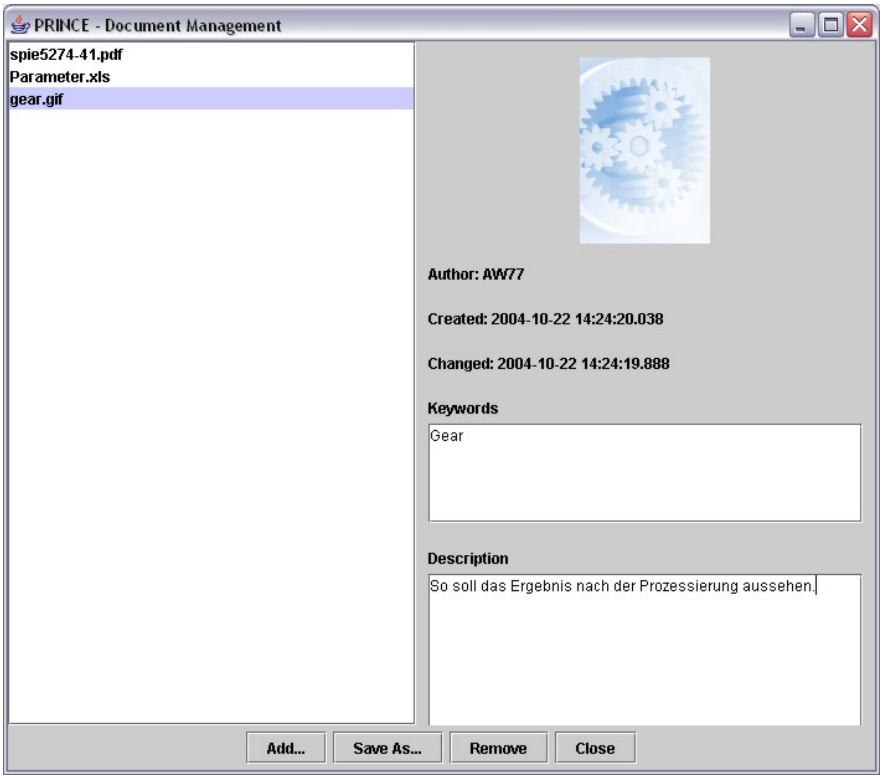


Abbildung 6.6: PRINCE: Dokumentenmanagement

ten Kriterien. Nun soll das elektronische Laborbuch nicht nur physischen Laborbücher ersetzen, sondern die gesammelten Daten für eine Entwurfsunterstützung zur Verfügung stellen. Die Parameter, die für Berechnungen notwendig sind, werden anhand ihres Namens erkannt. Beispielsweise benötigt ein Temperschritt die Temperatur, bei der getempert werden soll. Würde man nun den Autoren die Freiheit geben, die Namen der Parameter frei einzugeben, so könnte es schnell vorkommen, dass viele verschiedene Namen für ein und denselben Parameter vergeben werden. Bei dem gegebenen Beispiel der Temperatur könnte man *temperatur*, *temp.*, *temperature* schreiben. Der menschliche Nutzer wüsste wahrscheinlich sofort, dass jeweils immer der gleiche Parameter gemeint ist. Ein Softwarealgorithmus tut sich damit aber schwer. Um solche Namensvarianten für einen Parameter zu vermeiden, enthält das elektronische Laborbuch eine Parameterdatenbank. Diese Datenbank enthält alle Parameter mit den passenden Einheiten, Umrechnungsformeln und Zusatzinformationen. Der Autor eines Prozessschritts wählt aus dieser Datenbank den Namen des einzufügenden Parameters aus. Automatisch werden ihm dann auch die verfügbaren Einheiten des Parameters angezeigt.

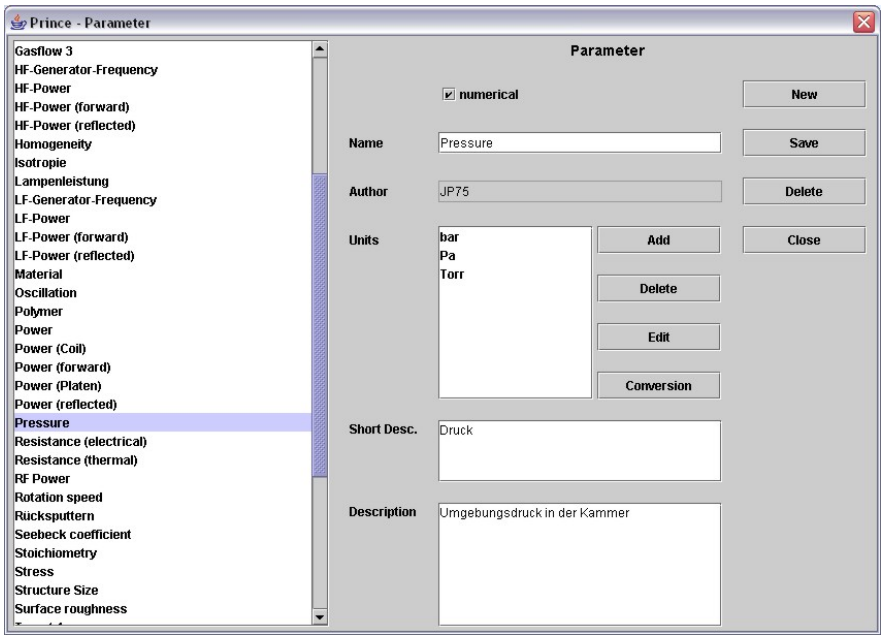


Abbildung 6.7: PRINCE Parametermanagement

Abbildung 6.7 gibt einen Überblick über diese Datenbank. Zur linken Hand befindet sich die Liste der Parameter. Rechts davon werden die Informationen

zu dem ausgewählten Parameter angezeigt. Diese sind der eindeutige Name, die verwendbaren Einheiten und Beschreibungen zum Parameter. Für die unterschiedlichen Einheiten gibt es die Möglichkeit, Umrechnungsformeln anzugeben, um eine Normalisierung und Vergleichbarkeit zu erreichen. Die Umrechnungsformeln werden in der standardisierten mathematischen Beschreibungssprache MathML [Wor03] beschrieben. MathML basiert auf der Extensible Markup Language (XML) [Wor04]. Abbildung 6.8 zeigt als Beispiel die Umrechnung von Stunden in Minuten.

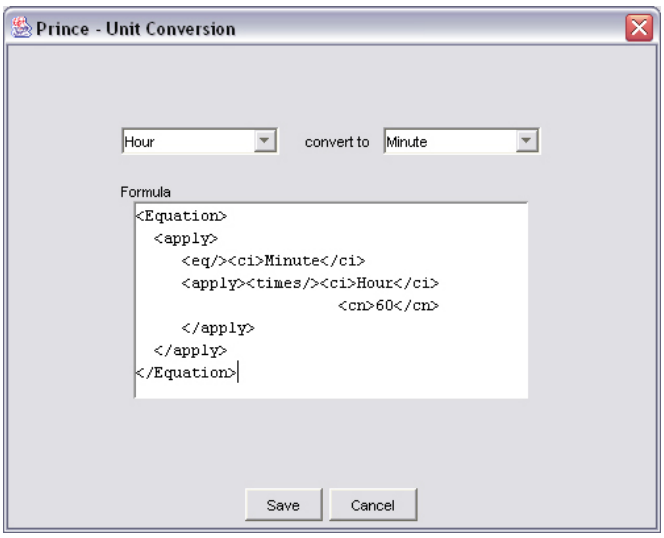


Abbildung 6.8: PRINCE: Einheitenumrechnung mit Hilfe von MathML

Bedingungen

Die Bedingungen werden in einem gesonderten Bedingungsdialog eingegeben. Aufgesplittet nach Vor- und Nachbedingungen präsentiert sich dem Nutzer eine intuitive Maske zur Eingabe der Bedingungen. Entsprechend der schon vorgestellten Regeltypen können die Regeln über einfaches Auswählen und Konfigurieren erstellt werden. Zusätzlich lassen sich zu jeder Regel Einschränkungen und Aufhebungen definieren. Auf diese Weise wird es ermöglicht, auch komplexere Zusammenhänge ohne die Nutzung von komplizierten Beschreibungssprachen darzustellen. Abbildung 5.7 zeigt eine solche Maske, die bereits mit einigen Regeln popularisisiert wurde.

Materialien

Für die Verwaltung der unterschiedlichen Materialien und ihrer Eigenschaften bietet sich ebenfalls eine Baumstruktur an. Ähnlich der Prozessschritte sind auch die Materialien in einer Hierarchie angeordnet, die auf dem Prinzip der Klassifizierung und Vererbung basiert. Abbildung 6.9 zeigt ausgewählte Materialien in einer solchen Baumstruktur. Zu jedem Material können beliebig viele Eigenschaften angegeben werden. Für informelle Dokumentationen besteht die Möglichkeit, Dokumente jeglicher Herkunft zu einem Material anzugeben.

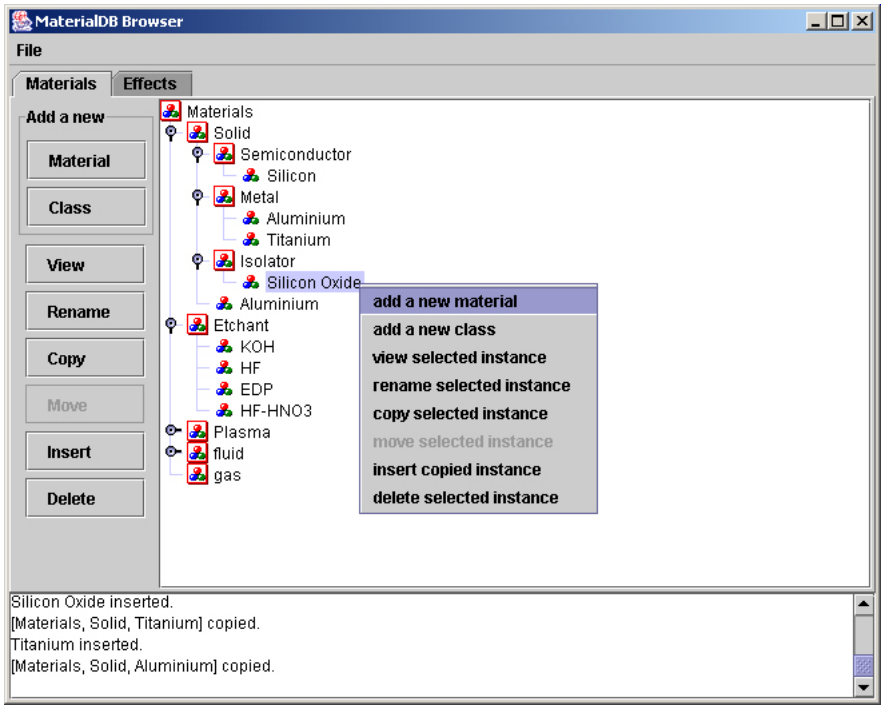


Abbildung 6.9: PRINCE: Materialverwaltung

Effekte

Effekte werden ähnlich der Materialien verwaltet. Auch sie werden in einer Baumstruktur verwaltet. In Abbildung 6.10 sind die Eigenschaften des Effekts *KOH etches* SiO_2 zu sehen. Neben den Parametern des Effekts, hier Temperatur, Ätzrate und Temperaturkoeffizient, besteht die Möglichkeit Modelle anzugeben. Jedem Effekt können beliebig viele Modelle zugeordnet werden. So

erreicht man, dass je nach Bedürfnis des Nutzers das passende Modell gewählt werden kann. Zu jedem Modell kann eine Schnittstelle angegeben werden. Die Schnittstelle beschreibt, welches Programm für die Berechnung des Modells benötigt wird. Zum einen lassen sich so unterschiedlich genaue Modelle vorhalten. Zum anderen ist so aber auch gewährleistet, dass zu unterschiedlichen Programmen auch Modelle vorgehalten werden können.

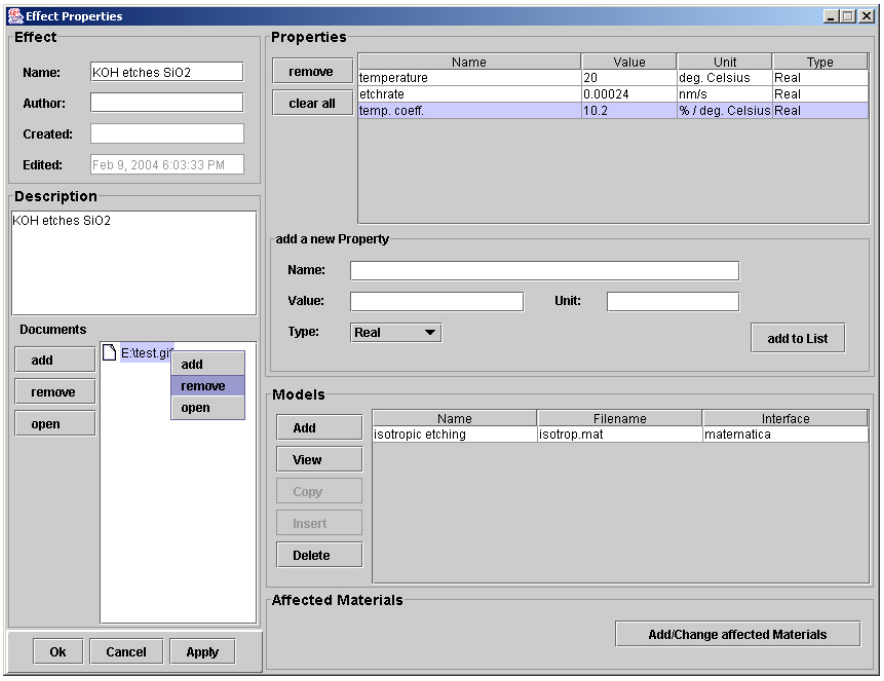


Abbildung 6.10: PRINCE: Eigenschaften von Effekten

6.7.2 Suchen und Auswerten

Ein echter Mehrwert gegenüber den traditionellen Laborbüchern ist schon erreicht, wenn über den Gesamtdatenbestand eine Suche nach Prozesschritten mit unterschiedlichen Suchkriterien ausgeführt werden kann. In den Papierversionen der Laborbücher ist eine Suche nahezu unmöglich, sobald mehrere Bücher involviert sind. In Abbildung 6.11 sieht man die Suchmaske für Prozessschritte. Prozessschritte lassen sich nach beinahe beliebigen Kriterien suchen. Neben den obligatorischen Suchanfragen nach Namen oder Autoren, können Prozessschritte auch anhand ihrer Parameter bzw. Parameterwerte

(siehe Abbildung 6.11) gesucht werden. Auf diese Weise lassen sich einfach und schnell die gewünschten Informationen im Laborbuch finden.

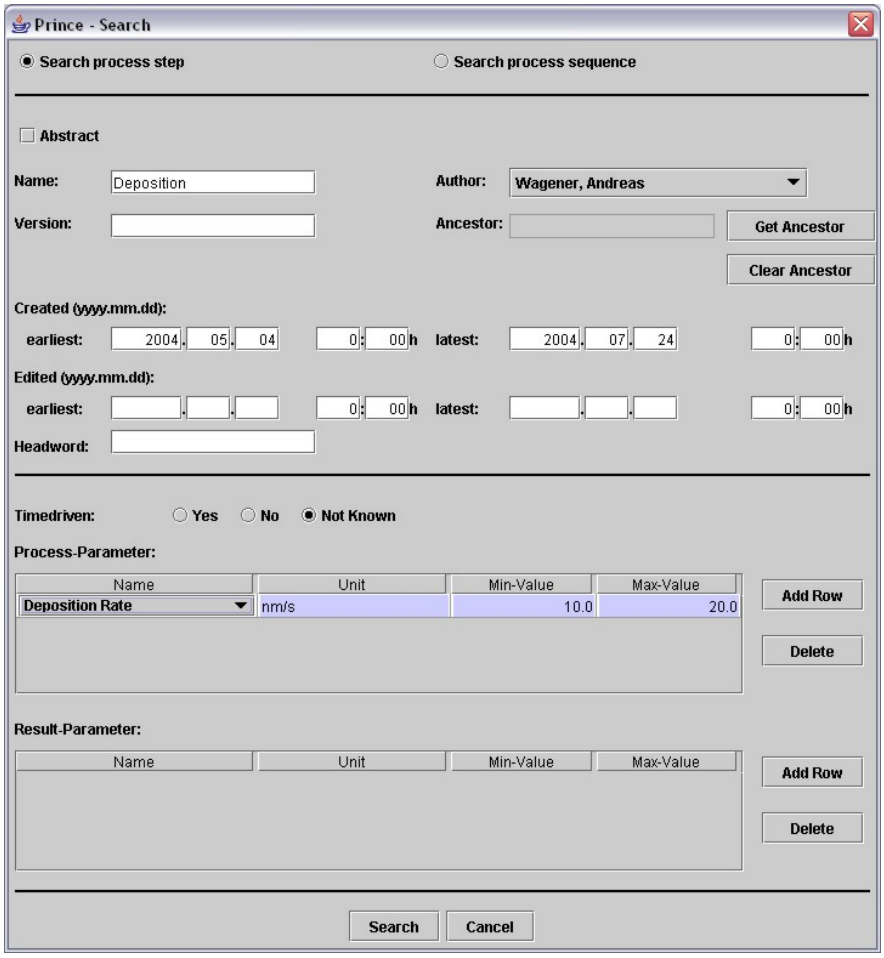


Abbildung 6.11: PRINCE Suche von Prozessschritten

Ein weiterer Mehrwert ist gegeben durch die Möglichkeit der Auswertung. Für die Entwickler ist es häufig interessant zu wissen, ob und wie ein Parameter von anderen abhängt. Solche Abhängigkeiten lassen sich besonders gut mittels Diagrammen darstellen und erkennen. Über solche Auswertungen können auch die Parameterschwankungen von Prozessschritten untersucht werden. Hat man beispielsweise mehrere Läufe von einem Ätzprozessschritt mit immer den gleichen Parametern durchgeführt, kann die Schwankung der Ergebnisse einfach anhand einer Kurve abgelesen werden.

6.7.3 Zugriff

Ein besondere Eigenschaft des traditionellen Laborbuchs ist die Möglichkeit, überall und jederzeit auf die Einträge zuzugreifen, wenn man es bei sich hat. Nachteilig wiederum ist, dass nur die eigenen Eintragungen in dieser Freiheit zugreifbar sind. Eintragungen in fremden Laborbüchern sind in der Regel nicht oder nur sehr schwer erreichbar. An dieser Stelle hat die elektronische Fassung einen bedeutenden Vorteil. Durch eine geschickte Wahl der Architektur, beispielsweise als verteiltes Client-Server-System, kann der Zugriff auf die Daten von jedem netzwerkfähigen Computer ermöglicht werden. Damit

Abbildung 6.12: PRINCE Login Dialog

nun nicht jeder beliebige Computerbenutzer Zugriff auf die doch teils sehr sensiblen Daten hat, ist eine Zugriffskontrolle sinnvoll. Bevor der Nutzer also Zugang zum elektronischen Laborbuch bekommt, muss er sich authentifizieren. Dies geschieht über eine Kombination aus Benutzernamen und Passwort. Weiterhin muss er angeben, auf welches elektronische Laborbuch er Zugriff haben möchte. Dies geschieht über die Angabe des Servers, auf dem das elektronische Laborbuch platziert ist (siehe Abbildung 6.12). Denkbar ist zum Beispiel, dass in einer Firma im Rahmen von Joint-Ventures mit anderen Partnern mehrere Laborbücher Verwendung finden.

Damit nun nicht jeder, der Zugang zum elektronischen Laborbuch hat, alle Daten einsehen kann, ist ein Rechtemanagement für diese notwendig. Über das Rechtemanagement kann für jeden Prozessschritt angegeben werden, wer darauf Zugriff hat und was er mit dem Prozessschritt machen darf. So ist es

beispielsweise sinnvoll, Prozessschritte, die noch in der Entwicklungsphase sind, nicht für alle sichtbar zu machen, sondern nur für das eigene Entwicklerteam. Auf diese Weise können auch externe Partner, um das Beispiel der Joint-Ventures aufzugreifen, Zugang zum Laborbuch erhalten, sehen aber nur die für sie bestimmten Datensätze. Abbildung 6.13 verdeutlicht diesen Aspekt.

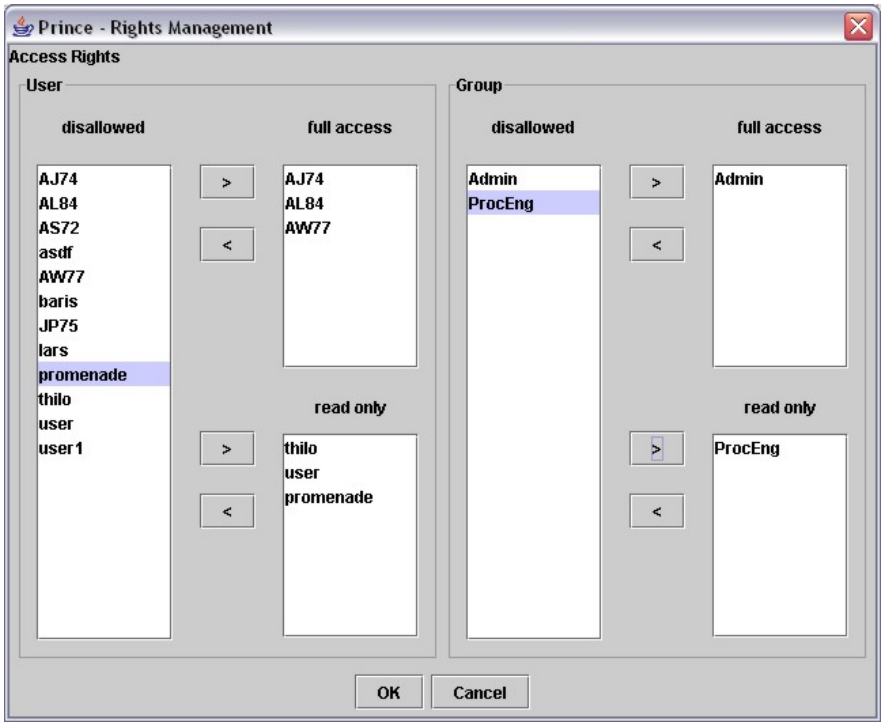


Abbildung 6.13: PRINCE Rechtemanagement

Für jeden Prozessschritt lässt sich detailliert angeben, welcher Nutzer oder welche Personengruppe die Eintragungen lesen bzw. verändern darf. Dazu können die Benutzer des elektronischen Laborbuchs verschiedenen Gruppen zugeordnet werden. Der Gruppenzuteilung sind keine Grenzen gesetzt. Jeder registrierte Nutzer des elektronischen Laborbuchs kann beliebig vielen Gruppen zugeteilt werden.

6.8 Fazit

Das Konzept der Objektorientierung wurde bereits sehr früh auch für die Mikrosystemtechnik entdeckt. Jedoch wird es erst in [Pop04] vollständig um-

gesetzt durch die Einführung der Mehrfachvererbung. Die Abstraktion und Klassifizierung in Kombination mit der Mehrfachvererbung ermöglichen eine intelligente Datenhaltung.

Ebenfalls neu in der Mikrosystemtechnik ist das Konzept der Effekte. Durch die Abstrahierung der Modelle und der Möglichkeit einer allgemeinen Beschreibung der Effekte eröffnen sich neue Bereiche der Unterstützung. In Kombination mit der hierarchisierten Darstellung der Materialien stellen sie ein mächtiges Werkzeug für die Synthese und Simulation zur Verfügung.

Die Untersuchung der möglichen Nutzergruppen führte zu einem an die Bedürfnisse der Nutzer angepasste Lösung der Datenhaltung. Das elektronische Laborbuch ermöglicht erstmals die zentrale Sammlung von Daten und Informationen und den orts- und zeitunabhängigen Zugriff auf diese. Dadurch ist es zum ersten Mal möglich, den Informationsfluss in Hinsicht auf Zeit und Ort aufzutrennen.

Das vorgestellte elektronische Laborbuch lässt sich nur mittels eines verteilten Systems realisieren. Welche Architektur sich von Seiten der Softwaretechnik dafür anbietet, wird im folgenden Kapitel vorgestellt.

7 System-Architektur

Die beschriebenen Eigenschaften des elektronischen Laborbuchs lassen sich nur mit Hilfe eines verteilten Systems realisieren. In diesem Kapitel wird eine mögliche Softwarearchitektur vorgestellt, die auf Basis innovativer Technologien erstellt wurde. Den Anfang macht ein Überblick über mögliche verteilte Systeme. Anschließend wird die Architektur für den Server-Teil und für den Client-Teil separat betrachtet.

7.1 Client-Server-System

Verteilte Computersysteme findet man heutzutage überall. Charakteristisch für ein verteiltes System ist die Zusammensetzung aus einzelnen Komponenten, die beispielsweise auf verschiedenen, über ein Netzwerk verbundenen Computern platziert sind. Die Zusammenarbeit der einzelnen Komponenten wird über den Austausch von Nachrichten realisiert [CDK02]. Prominentestes Beispiel für ein verteiltes System ist das Internet. Auf vielen über die ganze Welt verteilten Computern werden unterschiedliche Dienste angeboten, wie z. B. Email-Dienste, Webseiten und Newsgroups. Programme, die einen oder mehrere Dienste zur Verfügung stellen, nennt man auch *Server*. Im Gegenzug heißen Programme, die Dienste in Anspruch nehmen *Clients*. Alternativ spricht man deshalb bei verteilten Systemen auch von Client-Server-Systemen. Ein Client kann ein Programm sowohl ohne als auch mit Benutzerinteraktion sein. Der Einfachheit halber soll hier unter dem Begriff Client ein Programm mit Benutzerschnittstelle verstanden werden. Somit kommuniziert der Benutzer mit dem Client und der Client mit einem oder mehreren Servern. Damit ist bereits das Hauptmerkmal eines verteilten Systems eingeführt, nämlich der nicht monolithische Aufbau.

Client-Server-Systeme lassen sich nach der Anzahl der Komponenten oder auch Schichten kategorisieren [Som01]. Dabei stellt eine Schicht eine geschlossene Einheit (Client oder Server) dar, die über eine wohldefinierte Schnittstelle mit anderen Schichten kommuniziert. Das einfachste Client-Server-System ist ein Zweischichtsystem. Eine Ausprägung der Zweischichtsysteme besteht beispielsweise aus einer Datenhaltungs- und einer Datenverarbeitungsschicht. Die Datenhaltungsschicht übernimmt die Rolle des Servers, beispielsweise eines Datenbankservers. Die Datenverarbeitungsschicht stellt den Client dar, der auf den Datenbankserver zugreift und mit dem Benutzer interagiert.

Der Unterschied zur monolithischen Struktur besteht darin, dass die persistente Datenhaltung ausgelagert ist. Das birgt mehrere Vorteile. Die Datenhaltung wird häufig über Datenbanken realisiert. Datenbanken gibt es als eigenständige Programme. Sie müssen also nicht selbst implementiert werden. Über meist standardisierte Schnittstellen kann man mit einer Datenbank kommunizieren. Die Datenbank wird also als eigenständige Komponente bzw. Schicht betrieben. Auf diese Weise können auch mehrere Nutzer gleichzeitig dieselbe Datenhaltung nutzen. So lässt sich realisieren, dass viele Nutzer denselben Datenbestand vorfinden, der wiederum zentral administriert werden kann.

Ein Nachteil der zweischichtigten Systeme ist jedoch, dass die gesamte Programmlogik beim Client liegt. Die Wartung der Software wird erschwert und je nach Art und Umfang des Programms können spezielle Anforderungen an die Hardware gestellt werden. Dreischicht-Architekturen trennen deshalb einen Großteil der Programmlogik vom Client und stellen sie als eigenständige Schicht in Form eines Servers zur Verfügung. Da nun ein Großteil der Applikation in einem Server läuft, spricht man auch von Applikationsserver oder -schicht. Durch die Separation der eigentlichen Applikation erreicht man, dass die Hauptaufgabe des Clients nur noch auf die Benutzerkommunikation und Darstellung der Ergebnisse beschränkt wird. Der Client kommuniziert direkt mit dem Applikationsserver, der wiederum mit dem Datenbankserver Nachrichten austauscht. Eine direkte Kommunikation des Clients mit dem Datenbankserver ist in der Regel nicht vorgesehen. Einige Vorteile eines Applikationsservers sind die einfachere, da zentrale Wartung und die einfache Hardwareanpassung. So erreicht man, dass auf Clientseite keine besonderen An-

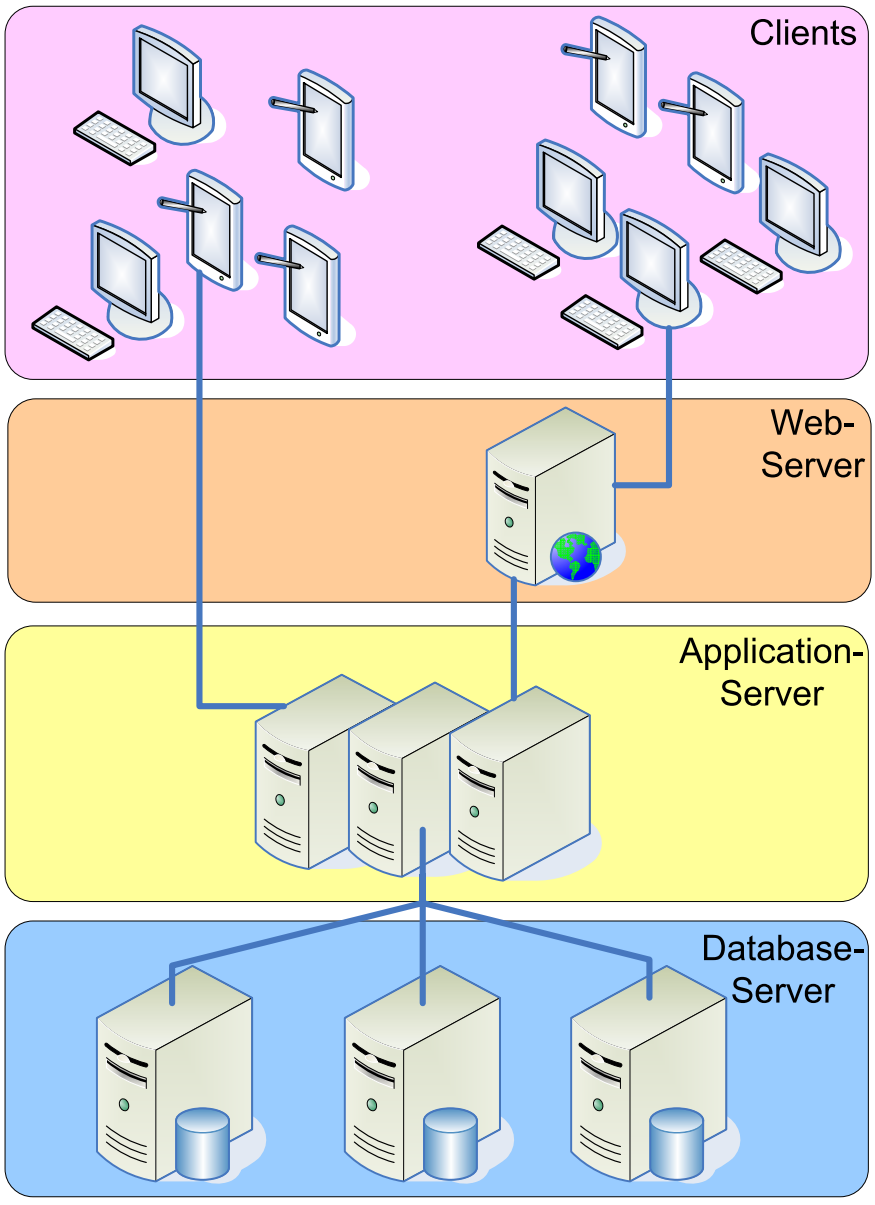


Abbildung 7.1: Mehrschicht-Architektur als verteiltes System

sprüche an die Hardwareumgebung erhoben werden. Solche Clients werden auch als Thin-Clients bezeichnet.

Für den Benutzer erfolgt die Auftrennung der Software in mehrere Schichten, die eventuell noch örtlich getrennt ausgeführt werden, völlig transparent. Die Schichten kommunizieren in der Regel über genau definierte Schnittstellen. Durch die Nutzung von definierten Schnittstellen ist es sogar möglich, beispielsweise die Datenbank transparent auszutauschen, solange die gleichen Schnittstellen verwendet werden.

Der Nutzer arbeitet direkt nur mit der Darstellungsschicht, die oft nur noch das Nötigste an Programmlogik beinhaltet. Möchte man auch noch darauf verzichten, kann man die Darstellungsschicht weiter aufsplitten. Man spricht bei mehr als drei Schichten allgemein von Mehrschicht-Architekturen. Über den Einsatz eines Webservers, der über einen Browser angesprochen werden kann, lässt sich auch die Darstellung auf die Serverseite holen. Der Webserver stellt die Benutzeroberfläche als Webseite dar, auf die die Nutzer zugreifen können. In diesem Fall spricht man auch von Web-Clients, da der Nutzer lediglich noch einen Webbrowser benötigt, die Software also von nahezu jedem netzwerkfähigen Rechner ausgeführt werden kann.

Abbildung 7.1 stellt eine solche Mehrschicht-Architektur dar. Auf unterster Ebene ist die Datenhaltungsschicht abgebildet. Sie besteht in der Regel aus einer oder mehreren Datenbanken. Darüber liegt die Applikationsschicht. Hier werden alle zentralen Algorithmen ausgeführt. Die Darstellung und Kommunikation mit dem Nutzersystem erfolgt über einen Webserver. Alternativ kann der Client auch direkt mit der Applikationsschicht kommunizieren. Im Folgenden sollen die einzelnen Komponenten und die Kommunikationsschnittstellen näher betrachtet werden.

Für die Implementierung des elektronischen Laborbuchs und der bereits in Kapitel 5 vorgestellten Unterstützungsmodule bietet sich die Drei- oder Mehrschichtarchitektur an. Bereits im letzten Kapitel wurde verdeutlicht, dass eine zentrale Datenhaltung mit Zugriff von nahezu allen netzwerkfähigen Computern notwendig ist. Aufgrund der Komplexität der Algorithmen, die für die Entwurfsunterstützung notwendig sind (man denke nur an die Simulation oder die Konsistenzüberprüfung von Prozessfolgen), ist es unumgänglich, diese in einer gesonderten Applikationsschicht unter zu bringen.

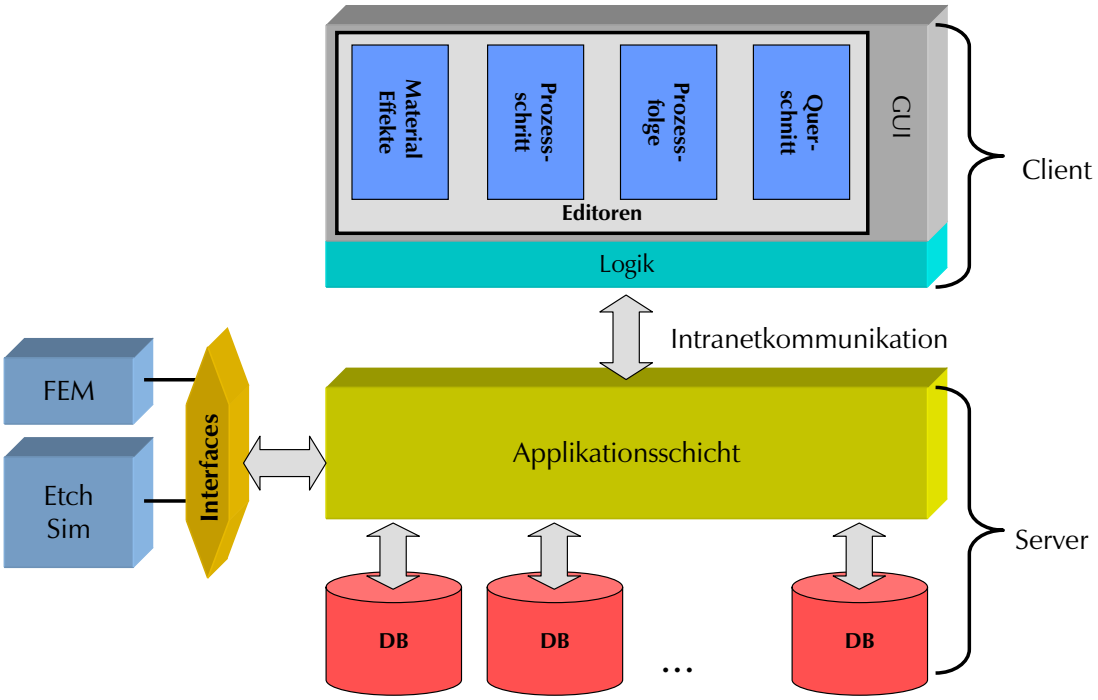


Abbildung 7.2: System-Architektur

Abbildung 7.2 gibt einen Überblick über die gewählte Architektur. Für die Datenhaltung werden relationale Datenbanken verwendet. Obwohl die vorkommenden Daten, wie im letzten Kapitel gezeigt, objektorientiert modelliert werden, wird keine objektorientierte Datenbank verwendet. Der Grund liegt in der unzureichenden Einhaltung der Standardisierung. Zwar gibt es einen Standard für objektorientierte Datenbanken [CB00], jedoch implementiert jeder Hersteller eigene Erweiterungen, sodass die Datenbanken nicht mehr austauschbar sind. Im Gegensatz dazu sind relationale Datenbanken in der Regel leicht austauschbar und entsprechen einheitlichen Standards, sodass relationale Datenbanken in der Regel ohne großen Aufwand ausgetauscht werden können. Weiterhin spricht für den Einsatz relationaler Datenbanken der hohe Verbreitungsgrad. Sowohl freie als auch kommerzielle relationale Datenbanken sind fast überall im Einsatz.

Für die Applikationsschicht wird ein Java Enterprise Server verwendet. Die Java 2 Enterprise Technologie (J2EE) wurde von der Firma Sun Microsystems [Sun04b] spezifiziert. Die Spezifikation ist unter [Sun03] zu finden. Dabei handelt es sich um einen Applikationsserver, der plattformunabhängig imple-

mentiert und mit sogenannten Enterprise Java Beans (EJB) bestückt wird. Die EJB enthalten den Programmcode der Applikation. EJB werden in der Programmiersprache Java beschrieben, die ebenfalls von Sun entwickelt wurde. Der Applikationsserver selbst stellt einen Großteil der für einen Serverbetrieb notwendigen Mechanismen bereits zur Verfügung. So werden die Schnittstellen zur Kommunikation mit Clients und anderen Servern bereit gestellt. Auch für die Anbindung relationaler Datenbanken gibt es standardisierte Schnittstellen. Die Herstellung der Objektpersistenz erfolgt automatisch durch den Server. Auf weitere Details zur J2EE Technologie soll hier verzichtet werden. Einen guten Überblick über die Java Enterprise Technologie und die einzelnen bereitgestellten Dienste und Mechanismen gibt [FCF02]. Speziell für die Entwicklung von EJB empfiehlt sich die Lektüre von [MH04].

Die im letzten Kapitel vorgestellten Effekte können unterschiedliche Modelle besitzen, für deren Berechnung eventuell Spezialsoftware notwendig ist. Bisherige Lösungen setzen voraus, dass diese Spezialsoftware jedem Benutzer zur Verfügung steht. Da solche Spezialsoftware in der Regel sehr teuer ist und deren Wartung sich oft auch sehr aufwendig gestaltet, soll hier ein anderes Konzept vorgestellt werden. Über spezielle Schnittstellen wird die Spezialsoftware direkt vom Applikationsserver angesprochen und genau wie dieser zentral vorgehalten. Dies geschieht vollständig transparent für den Benutzer.

7.2 Benutzerschnittstelle

Die Benutzerschnittstelle wird dem Benutzer als Thin-Client zur Verfügung gestellt. Technologie der Wahl ist hier wiederum Java. Java ist die einzige plattformunabhängige Programmiersprache, so dass gewährleistet ist, dass die Software auf allen gängigen Computern ausführbar ist, wie es für das elektronische Laborbuch bereits gefordert wurde. Für die Entwicklung von Java-Programmen gibt es zwei Strategien. Einerseits kann ein Java-Programm als *stand-alone* Lösung implementiert werden, für deren Ausführung lediglich die Java Runtime Environment (JRE) erforderlich ist. Andererseits gibt es die Möglichkeit, das Programm als Applet zu implementieren. Applets sind Java-Programme, für deren Ausführung ein Browser von Nöten ist. Sie müssen vor

der Ausführung von einer Webseite geladen werden. Beide Ansätze haben ihre Vor- und Nachteile.

Der Vorteil einer Java-Applikation gegenüber einem Applet liegt darin, dass das Programm nur einmal installiert werden muss. Applets hingegen werden vor jeder Ausführung erneut von der Webseite geladen. Je nach Größe des Programms kann dieses Herunterladen sehr aufwendig sein. Vorteil des Applets gegenüber der Applikation ist das sogenannte Sandbox-Prinzip. Das Sandbox-Prinzip beschreibt den Sachverhalt, dass ein Applet, das in einem Browser ausgeführt wird, keinen automatischen Zugriff auf lokale Ressourcen wie Laufwerke, Drucker oder Netzwerk erhält. Es läuft in einer abgeschotteten Umgebung. Der Benutzer muss jeder Ressourcennutzung zustimmen. Weiterhin lassen sich Programmupdates über Applets sehr viel einfacher verteilen als über Applikationen. Da jedes neue Starten des Applets automatisch ein neues Herunterladen mit sich bringt, kann sich der Nutzer sicher sein, immer die aktuellste Version der Software vorliegen zu haben. Bei der Applikation geschieht das Update nicht automatisch, sondern muss händisch erledigt werden.

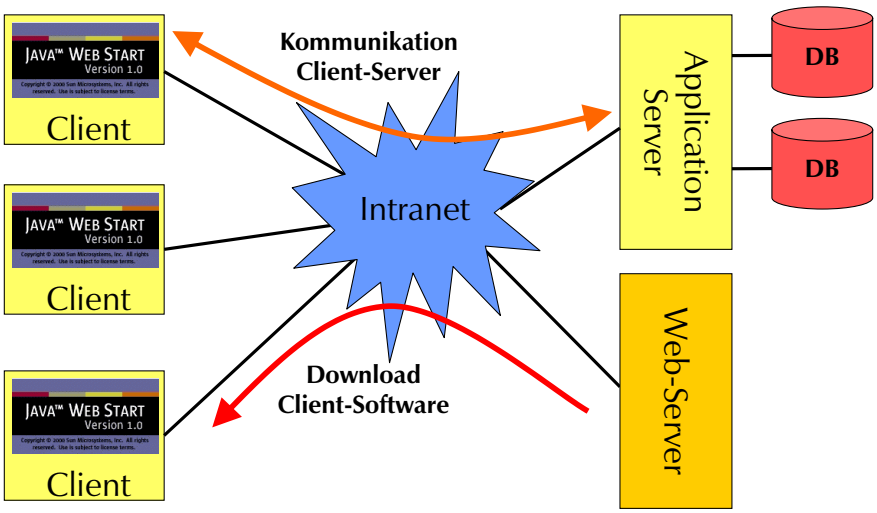


Abbildung 7.3: Distribution via Java WebStart

Zusammenfassend kann man sagen, dass weder die Applikation noch das Applet besonders hervorragen. Geschickter ist eine Kombination beider Ansätze. Die Java WebStart Technologie von Sun verfolgt diesen kombinierten Ansatz [Sun04a]. Direkt am Beispiel soll diese Technologie vorgestellt werden. Abbil-

dung 7.3 stellt die einzelnen beteiligten Schichten der Architektur noch einmal im Kontext der Distribution dar. Bekannt aus der Dreischicht-Architektur sind die Datenbankschicht und die Applikationsschicht rechts in der Grafik. Neu hinzugekommen ist ein Webserver. Der Webserver stellt auf einer Webseite die Clientsoftware bereit. Diese ist als Java Applikation geschrieben. Der Benutzer kann sich die Applikation über seinen Browser wie ein Applet auf seinen Computer laden. Im Gegensatz zum Applet wird sie allerdings nicht im Browser sondern in der WebStart-Umgebung ausgeführt. Beendet der Benutzer das Programm, so wird es wie eine Applikation lokal auf dem Rechner gespeichert.

Beim Neustarten des Programms kommt nun die Zwitterstellung der WebStart Technologie zum Tragen. Bevor das Programm neu gestartet wird, gleicht die WebStart-Umgebung automatisch die vorhandene Version mit der auf dem Webserver verfügbaren Version ab. Ist dort eine neuere Version vorhanden, wird sie automatisch heruntergeladen und anschließend gestartet. Sind die Versionen gleich, wird die lokale Version direkt gestartet, der Download entfällt. Genau wie ein Browser stellt die WebStart-Umgebung eine Sandbox-Umgebung für die Applikation bereit. Zugriffe auf lokale Ressourcen sind nur nach Rücksprache mit dem Benutzer möglich. Auf diese Weise vereint die WebStart-Umgebung die Vorteile beider Java-Programmausprägungen und erleichtert Softwareaktualisierungen enorm. Sobald das Programm erst einmal geladen ist, verhält es sich wie ein ganz normales Java-Programm. Der Client kann also uneingeschränkt mit dem Applikationsserver kommunizieren.

7.3 Fazit

Mit der Wahl einer Dreischicht-Architektur wurde eine Architektur für die Entwurfsunterstützung ausgesucht, die etabliert und quasi als Industriestandard angesehen werden kann. Auf der Serverseite werden Technologien eingesetzt, die sich durch Plattformunabhängigkeit, hohe Performanz und leichte Erweiterbarkeit auszeichnen.

Für den Client wird die Strategie des Thin-Clients verfolgt. Diese erleichtert den Einsatz und damit die Etablierung des Systems enorm. Der Thin-Client

stellt kaum Anforderungen an die Hardwareausstattung des Arbeitsplatzrechners und zeichnet sich ebenfalls durch Plattformunabhängigkeit aus.

Durch den erstmaligen Einsatz eines verteilten Systems zur Unterstützung des Mikrosystementwurfs ist es möglich, von jedem netzwerkfähigen Computer auf die Daten und Unterstützungsmodule zurückzugreifen. Durch den Einsatz der WebStart-Technologie wird weiterhin gewährleistet, dass die Nutzer sich nicht um Updates oder Installation kümmern müssen.

Im folgenden Kapitel wird die gesamte Entwurfsunterstützung vorgestellt, die bis jetzt nur in Teilen und kontextbezogen eingeführt wurde.

8 PRINCE Entwicklungsumgebung

In Zusammenarbeit mit der Forschungsabteilung für dünne Schichten der Robert Bosch GmbH, einem der führenden Hersteller von Mikrosystemen, wurde im Rahmen dieser Arbeit das Softwarepaket *PRINCE* entwickelt. PRINCE ist ein Akronym für ***Pr****ocess* ***In****formation and Management* ***Ce****nter*. PRINCE bietet zum einen die Möglichkeit, Prozessschritte und prozesschrittrelevante Daten einfach zu verwalten. Zum anderen können Prozessfolgen erzeugt, geprüft und simuliert werden. Durch die Nutzung von innovativen Softwarekonzepten, wie das im letzten Kapitel vorgestellte Client-Server-Konzept in Verbindung mit der Java- und WebStart-Technologie wurde ein intuitives System entwickelt, das sich nicht nur durch einfache Bedienung, Wartbarkeit und Erweiterbarkeit sondern auch durch Implementierung des erweiterten Kreismodells auszeichnet.

Grundlage der Arbeit ist die im Jahr 2001 ebenfalls in Kooperation mit Bosch erstellte System- und Anforderungsanalyse [Wag01] für eben ein solches Softwarewerkzeug. PRINCE zeichnet sich dadurch aus, dass erstmals die in Kapitel 4 vorgestellte Methodik des Kreismodells in ein Werkzeug umgesetzt wurde. Bisherige Lösungen decken jeweils nur Teilbereiche des fertigungsnahen Entwurfs ab. Insbesondere die Strategie der Kombination eines Entwurfswerkzeugs mit dem elektronischen Laborbuch wird mit PRINCE erstmals verfolgt.

Teile der Entwicklungsumgebung PRINCE wurden bereits in den vorherigen Kapiteln kontextbezogen vorgestellt. Hier sollen nun diese Teile zu einem Gesamtbild zusammengeführt und um die noch nicht diskutierten Bereiche ergänzt werden. Zu Anfang wird ein Überblick über die Architektur gegeben. Anschließend wird das Datenmanagement und die Umsetzung der Methodik diskutiert.

8.1 Architektur

Bereits im letzten Kapitel wurde die Gesamtarchitektur des PRINCE-Systems vorgestellt. Basierend auf der J2EE-Technologie ist PRINCE aus drei Schichten aufgebaut. Somit ist es erstmals in der Mikrosystemtechnik möglich, standort- und plattformunabhängig von nahezu überall auf Daten zuzugreifen und gemeinsam an Entwürfen zu arbeiten. Nachdem die Gesamtarchitektur bereits diskutiert wurde, soll hier ein Blick auf die Aufteilung der einzelnen Komponenten im PRINCE-System geworfen werden. In Abbildung 8.1 sind die einzelnen Komponenten und die Kommunikationswege aufgezeigt.

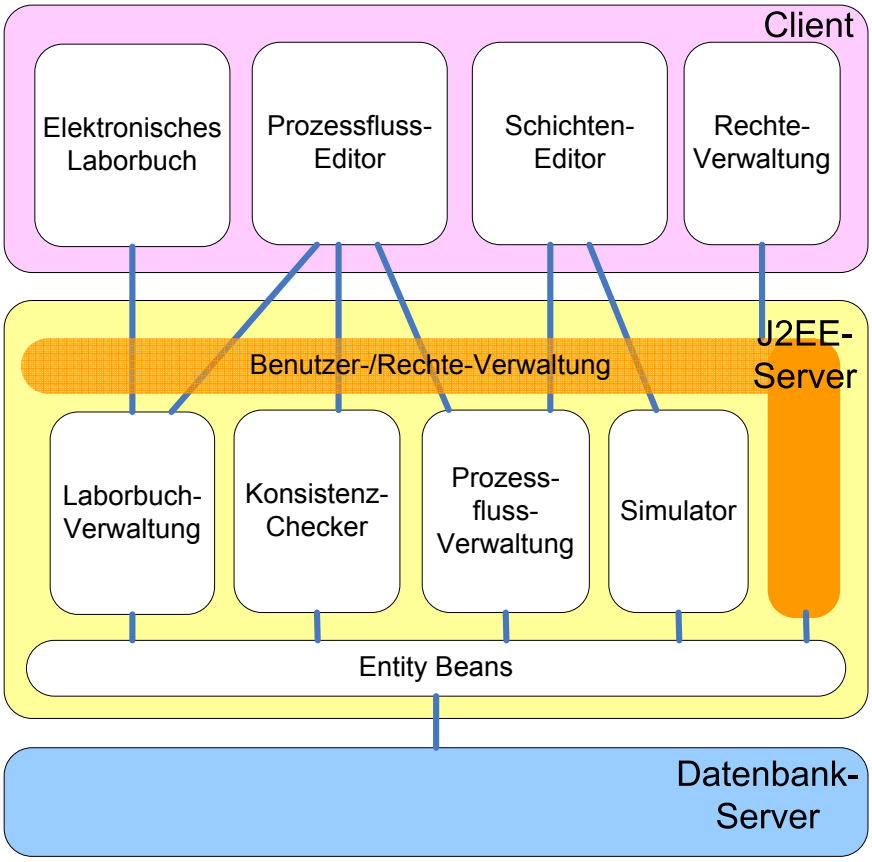


Abbildung 8.1: PRINCE: Software Architektur

Der Client setzt sich aus vier Modulen zusammen. Das sind das elektronische Laborbuch, das bereits in Kapitel 6 detailliert vorgestellt wurde, der Prozessflusseditor und der Schichteneditor, die nachfolgend besprochen werden, und die Rechteverwaltung, die ebenfalls schon vorgestellt wurde. Der Server

teilt sich in die Bereiche der Laborbuch-Verwaltung, des Konsistenz-Checkers, der Prozessfluss-Verwaltung und des Simulators ein. Umrahmt werden alle Bereiche von der Benutzer- und Rechteverwaltung. Die Kommunikation mit der Datenbank wird über eine Schicht realisiert, die die Entity-Beans enthält. Entity-Beans sind in der J2EE-Technologie Repräsentanten für Datenobjekte. Wird ein Datenobjekt angefordert, so wird automatisch eine Entity-Bean für dieses Objekt erstellt. Grundlage dafür sind die Daten in der Datenbank. Die Datenbank bleibt aber für alle Komponenten, mit Ausnahme der Entity-Beans, unsichtbar.

Die Kommunikationswege werden ebenfalls dargestellt. Zu bemerken ist hier, dass jegliche Kommunikation des Clients mit dem Server durch die Benutzer- und Rechte-Verwaltung geht. Auf diese Weise wird sicher gestellt, dass kein Benutzer auf Daten zugreifen kann, die ihm nicht zugänglich sein sollen.

8.2 Umsetzung Entwurfsmethodik

Die Entwicklungsumgebung PRINCE implementiert als erstes Werkzeug die Methodik, die durch das Kreismodell beschrieben ist. In Abbildung 8.2 ist noch einmal das erweiterte Kreismodell für den fertigungsnahen Entwurf abgebildet. Zusätzlich sind den einzelnen Phasen und dem Kernbereich bereits die Module zugeordnet worden, die in PRINCE implementiert sind.

Das Vorhalten der notwendigen Daten, die die Anwendung des Kreismodells überhaupt erst möglich machen, geschieht in PRINCE über das elektronische Laborbuch. Die Zusammenstellung der Prozessschrittfolgen geschieht, genauso wie deren Modifikation, mittels des Prozessfolgen Editors. Die anschließende Verifikation wird über den Schichteneditor in Verbindung mit dem Simulator ermöglicht. Der Schichteneditor ist ebenfalls für die Behandlung der Masken zuständig. Ein durch PRINCE noch nicht abgedeckter Bereich ist der der Synthese, also vom Layout bzw. Querschnitt zur Prozessfolge. An der Synthese wird in einem Folgeprojekt bereits gearbeitet, das zum Ende dieses Kapitels im Ausblick vorgestellt wird.

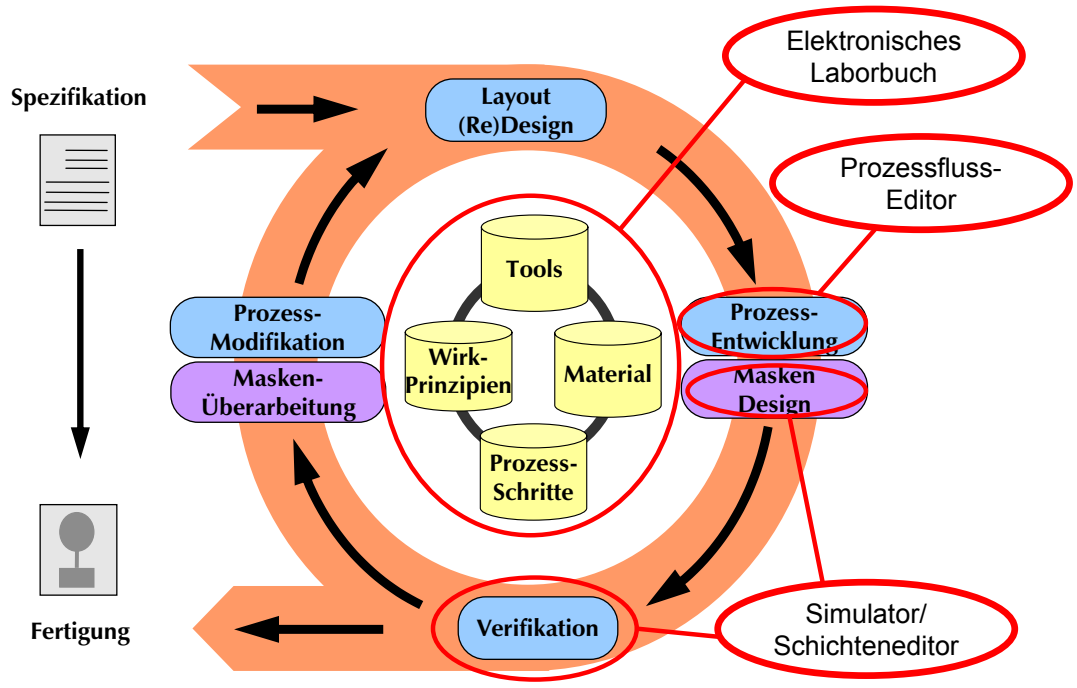


Abbildung 8.2: Umsetzung des erweiterten Kreismodells

8.2.1 Datenmanagement

Das erweiterte Kreismodell stellt den methodischen Hintergrund des PRINCE-Systems dar. Für die Umsetzung des Kreismodells ist es notwendig, bestimmte Daten und Informationen vorzuhalten. In PRINCE wird die Sammlung und Verwaltung der Daten und der beschränkbare Zugriff auf dieselben mittels des elektronischen Laborbuchs realisiert, dessen Konzept bereits in Kapitel 6 vorgestellt wurde. Das elektronische Laborbuch ermöglicht es erstmals, für die Mikrosystemtechnik relevante Daten und Informationen in geordneter Weise elektronisch abzulegen, zu verwalten und wieder abzurufen. Entsprechend Abbildung 8.2 sind Daten und Informationen zu Tools, Wirkprinzipien, Materialien und Prozessschritten vorzuhalten.

Materialien Materialdaten werden im elektronischen Laborbuch mittels des Material- und Effektmanagers verwaltet. Sie können in beliebiger Komplexität eingegeben und über die Mechanismen der Vererbung und Hierarchisierung leicht zugänglich gemacht werden.

Wirkprinzipien Die Wirkprinzipien und Effekte werden ebenfalls über den Material- und Effektmanager verwaltet. Hier gilt das gleiche, wie für die Materialien.

Tools Zum einen bildet natürlich PRINCE die zentrale Softwareumgebung für den Entwurf. Zusätzlich können aber über Schnittstellen externe Werkzeuge angesprochen werden. Dies geschieht in erster Linie über die Effekte. Den Effekten können Modelle zugeordnet werden, die zu externen Tools gehören.

Prozessschritte Prozessschritte bilden die zentralen Elemente des elektronischen Laborbuchs. Prozessschritte können mit allen relevanten Daten und informellen Informationen verwaltet werden. Zusätzlich können den Prozessschritten Effekte zugeordnet werden.

Diese Auflistung zeigt deutlich, dass PRINCE die daten- und informationstechnischen Voraussetzungen für den fertigungsnahen Entwurf vollständig erfüllt. Im folgenden Abschnitt soll gezeigt werden, wie die Entwurfsmethodik selbst in PRINCE umgesetzt wurde.

8.2.2 Prozessfluss Editor

Der Prozessfluss Editor ermöglicht die Erstellung von Prozessflüssen. Ein Prozessfluss besteht aus einer Sequenz von Prozessschritten. Für ein einfaches Handling können Prozessflüsse auch aus bereits bestehenden Prozessflüssen erstellt werden. Das ist besonders hilfreich für immer wiederkehrende Sequenzen, wie sie beispielsweise bei der Reinigung auftreten. In Abbildung 8.3 enthält der Prozessfluss mehrmals die Prozesssequenz RCA-Reinigung. Anstatt nun die Sequenz der Reinigungsschritte jedesmal manuell einzugeben, kann die Reinigungssequenz wie ein einzelner Prozessschritt eingefügt werden.

Die Prozessflüsse werden wie die Prozessschritte in der zentralen Datenbank abgelegt. Die Suche nach Prozessflüssen erfolgt auf gleiche Art und Weise wie die Suche nach Prozessschritten. Dargestellt werden die Prozessflüsse als Baum. In Abbildung 8.3 ist exemplarisch ein Prozessfluss zu sehen. Eingefügte

Prozessflüsse können auf diese Weise aus Gründen der Übersicht minimiert dargestellt werden.

Ähnlich wie bei den Prozessschritten können auch hier abstrakte Prozessflüsse erstellt werden. Abstrakt bedeutet hier, dass mindestens einer der einzelnen Prozessschritte der Sequenz selbst abstrakt ist. Ein abstrakter Prozessschritt ist, wie bereits in Kapitel 6 vorgestellt, eine Art Schablone für eine Reihe anderer Prozessschritte. Die Nutzung von abstrakten Prozessschritten in einer Prozesssequenz ermöglicht nun, den Entwurfsablauf möglichst durchgängig zu unterstützen. Eher selten kommt es vor, dass ein Entwickler bereits zu Anfang des Entwurfsprozesses jedes Detail, speziell hier jeden einzelnen Prozessschritt, kennt. Vielmehr dürfte er in Kategorien und Klassen denken, wie Abscheideprozessschritte, Reinigungsschritte usw. Der Prozessfluss Editor unterstützt diese Denkweise, indem eine Sequenz abstrakte Prozessschritte enthalten darf. So kann der Entwickler mit einer relativ abstrakten Vorstellung beginnen und den Entwurf immer weiter verfeinern und detaillieren.

Für die Fertigung müssen die abstrakten Prozessfolgen natürlich konkretisiert werden. Auch dieser Vorgang wird von PRINCE unterstützt. Soll eine abstrakte Folge konkretisiert werden, so macht PRINCE Vorschläge für die Konkretisierung der einzelnen abstrakten Prozessschritte. Dabei bedient sich PRINCE der Prozessschritthierarchie. Es werden also nur Prozessschritte vorgeschlagen, die von dem abstrakten Prozessschritt geerbt haben, also für die der abstrakte Prozessschritt als Schablone stand. Hat der Entwickler beispielsweise einen abstrakten Prozesschritt chemische Abscheidung (z. B. CVD) eingebaut, so werden ihm alle Prozessschritte für die Konkretisierung vorgeschlagen, die eine chemische Abscheidung realisieren. Details zur Nutzung des Prozessfluss Editors sind im Benutzerhandbuch [PWFB03] zu finden.

8.2.3 Konsistenz-Checker

Der Konsistenz-Checker dient der Verifikation einer Prozesssequenz. Dabei handelt es sich jedoch nicht um eine Verifikation im Sinne des kybernetischen Modells von Rammig. Für jeden Prozessschritt können, wie bereits in Kapitel 6 vorgestellt, beliebig viele Regeln bezüglich der Vor- und Nachprozessierung angegeben werden. Diese Regeln können sich sowohl auf Parameter als auch

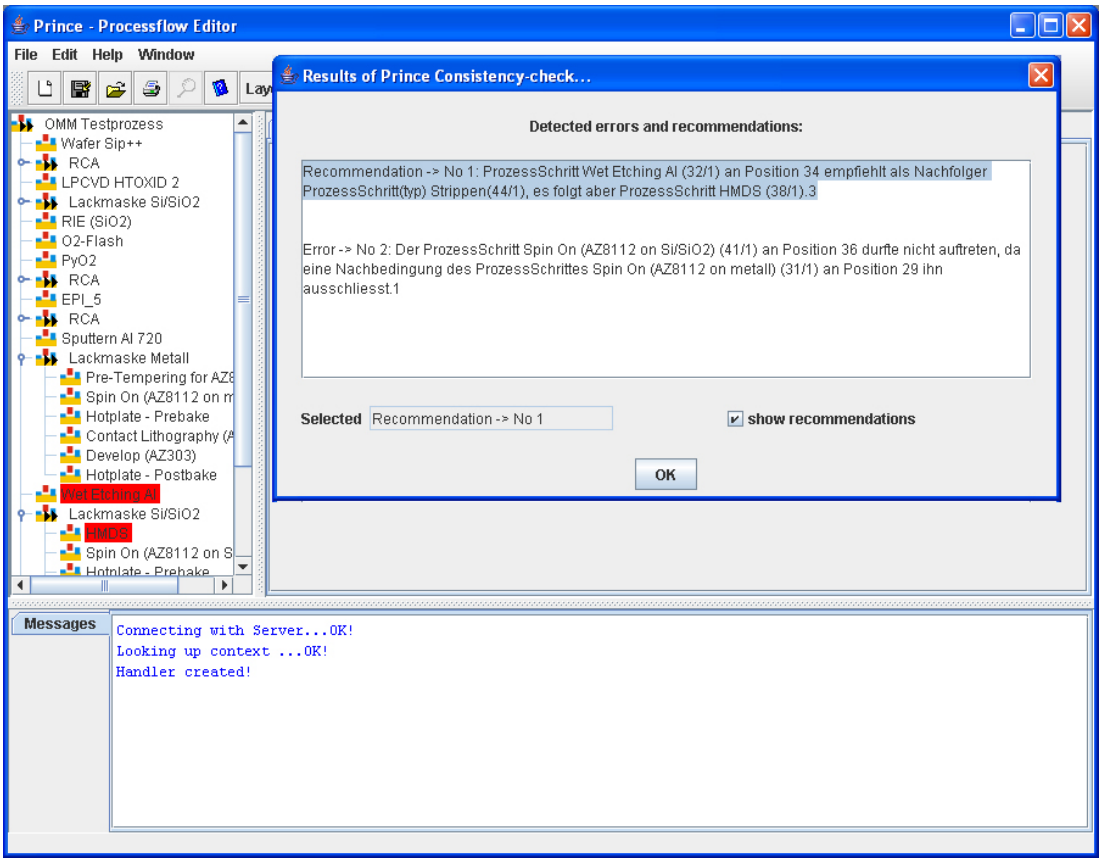


Abbildung 8.3: PRINCE: Prozessfluss Editor mit Konsistenzüberprüfung

auf ganze Prozessschritte beziehen. Sie schränken die möglichen Kombinationen an Prozessschritten in einer Folge ein bzw. fordern das Vorhandensein von bestimmten Prozessschritten.

Nun können im Prozessfluss Editor theoretisch beliebige Folgen von Prozessschritten zu Sequenzen zusammengesetzt werden. Ob diese Folgen nun technologisch Sinn machen, also überhaupt fertigbar sind, kann mittels des Konsistenz-Checkers überprüft werden. Der Konsistenz-Checker untersucht dazu für jeden einzelnen verwendeten Prozessschritt dessen Regeln auf Einhaltung. Treten Verletzungen der Regeln auf, so wird der Entwickler darüber informiert. Dazu wird die Verletzung selbst und die Stelle der Verletzung in der Prozessschrittsequenz angegeben. In Abbildung 8.3 ist das Ergebnis eines Konsistenz-Checks zu sehen. In diesem Beispiel wurden eine Empfehlung und ein Fehler ausgegeben und die Verursacher im Prozessfolgenbaum markiert. Erst nachdem die Konsistenz der Prozessfolge geprüft und sichergestellt ist, kann die Simulation der Prozessfolge erfolgen. Auf den verwendeten Algorithmus soll hier nicht eingegangen werden. Eine ausführliche Beschreibung der Arbeitsweise des Konsistenz-Checkers ist in [Stu04] zu finden.

8.2.4 Simulation

Die Simulation dient der Verifikation einer Prozessfolge. Wurde eine Prozessfolge erstellt, die die Konsistenzprüfung bestanden hat, kann sie in PRINCE simuliert werden. Für die Simulation stellt PRINCE bereits Simulationsmodelle auf hohem Abstraktionsniveau bereit. Idealerweise gibt es für jeden einzelnen Prozessschritt angepasste Modelle, die in PRINCE über die Effekte verwaltet werden. Das Vorhandensein eines spezialisierten Modells ist bei der Fülle der Prozessschritte nicht garantiert. Für diesen Fall wurde das Hierarchiekonzept ebenfalls bei den Simulationsmodellen eingeführt. Ist kein spezielles Modell für einen Prozessschritt vorhanden, so wird das Modell verwendet, das dem Prozessschritt zugeordnet ist, der in der Vererbungshierarchie eine Ebene über dem aktuellen Prozessschritt steht. Damit die Suche nach Modellen in der Hierarchie nicht ins Leere läuft, sind für die abstraktesten Klassen, wie Abscheidung, Lithografie, Ätzen etc. bereits qualitative Modelle implementiert. So wird sicher gestellt, dass auf jeden Fall simuliert werden kann. Die Ergebnisse der Simulation werden im Layer Editor angezeigt.

Zellulärer Prozesssimulator

Die in PRINCE vorhandenen Simulationsmodelle sind geometrischer Natur. Bei der Diskussion der Effekte wurde bereits erwähnt, dass auch externe Werkzeuge von PRINCE über definierte Schnittstellen angesprochen werden können. Im Rahmen dieser Arbeit wurde ein Simulationswerkzeug auf der Basis eines zellulären Automaten erstellt. Für die Realisierung des zusätzlichen Simulationswerkzeugs sprechen zwei Gründe. Erstens lassen sich einige Effekte sehr gut mit Hilfe zellulärer Automaten modellieren und zweitens dient das externe Simulationswerkzeug als Proof-of-Concept für die Anbindung externer Werkzeuge an PRINCE.

Die Besonderheit dieses zellulären Automaten besteht darin, dass der Zustand einer jeden Zelle durch zwei Werte beschrieben wird, nämlich das Material und einem reellen Wert zwischen 0 und 1. Dieser reelle Wert hängt in erster Linie von dem zu simulierenden Prozess ab. Bei einem Ätzprozess beschreibt er beispielsweise den Materialabtrag der Zelle. Ist der Wert 1, so wurde die Zelle noch nicht geätzt. Ist der Wert hingegen 0, so wurde die Zelle vollständig geätzt.

Für den Simulator wurde ein zweidimensionales und rechtwinkliges Gitter mit Moore-Umgebung gewählt. Die Moore-Umgebung stellt alle angrenzenden Zellen als Nachbarn dar. Im Gegensatz dazu gibt es noch die von-Neumann Umgebung, die nur die Zellen als Nachbarn ansieht, die direkt oben, unten, links oder rechts angeordnet sind. Eine Besonderheit stellt neben den Zellzustandsbeschreibungen auch die Übergangsfunktion dar. Die Grundform des zellulären Automaten geht von einer Übergangsfunktion für alle Zellen aus. Hier wurde aber eine Methode gewählt, die für verschiedene Zellen auch verschiedene Übergangsfunktionen zulässt. Auf diese Weise können beispielsweise abhängig vom Material der Zelle unterschiedliche Verhalten simuliert werden. So verhält sich eine Maskenmaterialzelle anders als eine Siliziumzelle. Das Material dient aber nicht nur der Unterscheidung sondern liefert auch Informationen zu den Parametern der Übergangsfunktion. Die Parameter werden vor Beginn der Simulation festgelegt. Die dafür notwendigen Informationen werden aus der zentralen Datenhaltung von PRINCE geholt.

Da die Laufzeit der Simulation ein kritischer Punkt in Bezug auf die Akzep-

tanz darstellt, wurde der Simulator optimiert. Die Prozesse der Mikrosystemtechnik spielen sich oftmals hauptsächlich in der Nähe der Oberfläche einer Schicht ab. Bezogen auf den zellulären Automaten heißt das, dass oberflächenferne Zellen oft untätig sind. Diesen Sachverhalt kann man sich zunutze machen, indem man Zellen in einen aktiven oder passiven Zustand versetzt. Ist eine Zelle im passiven Zustand, wird ihre Übergangsfunktion nicht neu berechnet. So wird erreicht, dass nur die unbedingt notwendigen Berechnungen durchgeführt werden.

Für den hier vorgestellten Simulator wurden generische Algorithmen entwickelt, die es ermöglichen, schnell und einfach für neue Materialien oder Prozessschritte Übergangsfunktionen zu erstellen. Die herkömmlichen Simulatoren beschränken sich in der Regel auf eine kleine Menge fest vorgegebener Materialien und können meist nur einen Prozesstyp (z.B. Ätzen) simulieren. Dieser Simulator ist jedoch in der Lage, durch die Möglichkeit des Initialisierens neuer Modelle zu Beginn der Simulation beliebige Prozessschritte zu simulieren. Als Beispiel wurden Modelle für Nasschemisches und Ionenstrahl-Ätzen, die Schichtabscheidung [Sch04] und die Oxidation [KS04] erstellt. Eine detaillierte Beschreibung der Algorithmen für die Übergangsfunktionen ist in [Sch04, KS04] zu finden. In Abbildung 8.4 ist das Ergebnis einer Simulati-

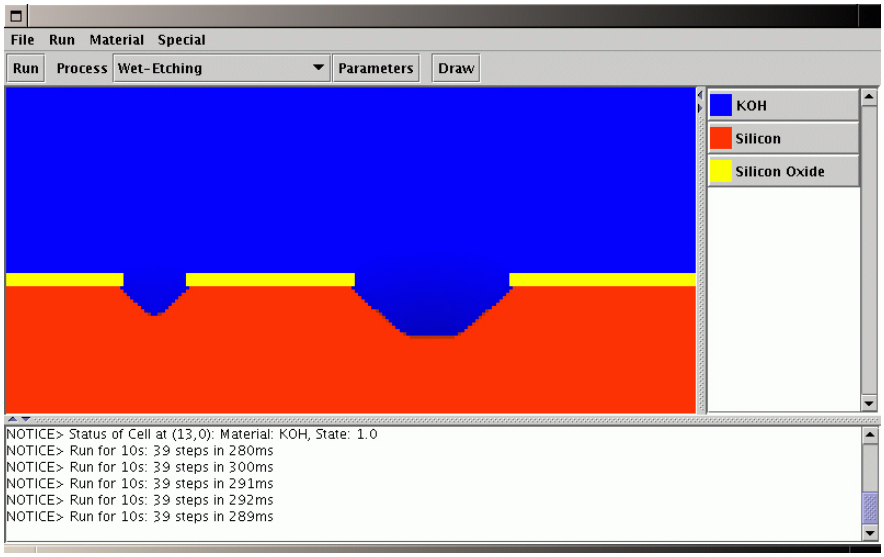


Abbildung 8.4: Zellulärsimulator

on mit Hilfe des zellulären Automaten zu sehen. In diesem Beispiel wurde

ein Nassätzschritt von Silizium mittels KOH simuliert, wobei Siliziumoxid als Maske dient.

8.2.5 Layer Editor

Liegt eine konsistente Prozessfolge vor, so kann die Simulation gestartet werden. Dies geschieht im Layer Editor. Der Layer Editor dient einerseits der Steuerung der Simulation und der Anzeige der Simulationsergebnisse. Andererseits können im Layer Editor auch Masken den einzelnen Prozessschritten zugeordnet werden. Es besteht die Möglichkeit, den Prozessfluss im gesamten zu simulieren oder Schritt für Schritt vorzugehen. Insbesondere letztere Strategie bietet sich an, wenn man die Auswirkungen der einzelnen Prozessschritte untersuchen möchte.

Für Lithografieschritte können Masken als Strichmasken angegeben werden. Strichmasken sind im Grunde Linien, die die Bereiche der Abschattung markieren. Diese werden direkt im Layer Editor eingezeichnet, was gewisse Vorteile hat. So können die Masken direkt passend zum Untergrund gezeichnet werden, beispielsweise wenn eine schrittweise Simulation durchgeführt wird. Das führt zu einer besseren Anpassung der Masken und verringert dadurch den Aufwand, da weniger Iterationen zur Maskenanpassung durchlaufen werden müssen.

Die Masken können für die Prozessfolge abgespeichert werden. Durch das Abspeichern liegen die Maskendaten für erneute Simulationen bereit. In Abbildung 8.5 ist das Ergebnis der Simulation der Prozessfolge zu sehen, deren Baumansicht im Prozessfluss Editor bereits in Abbildung 8.3 gezeigt wurde. Deutlich erkennt man, dass die Modelle stark auf Polygonstrukturen reduzieren. Jedoch ist sowohl der Simulator als auch der Layer Editor in der Lage komplexere Strukturen zu behandeln.

Über den Layer Editor können nicht nur die Ergebnisse der Simulation angezeigt werden. Auch lassen sich im Layer Editor weitere Schichten einzeichnen, die in der Simulation als Untergrund berücksichtigt werden. Insbesondere bei der schrittweisen Simulation lässt sich so sehr einfach testen, welche Auswirkungen eingefügte Schichten auf das Gesamtergebnis haben können. Angedacht ist, dass PRINCE für die manuell eingezeichneten Schichten Vorschläge

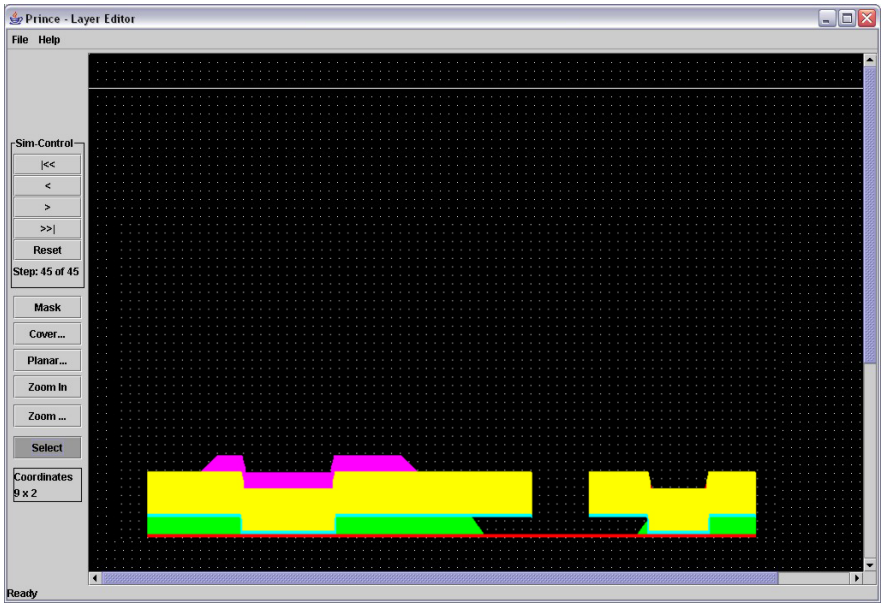


Abbildung 8.5: Simulationsergebnis

für erzeugende Prozessschritte bereitstellt. Dieser Aspekt wird im folgenden Ausblick aber näher beleuchtet.

8.3 Vorgehensbeschreibung

Zur Verdeutlichung der Methodenumsetzung in PRINCE soll hier das generelle Vorgehen beim Einsatz der Software beschrieben werden. Eine detaillierte Beschreibung ist im Handbuch zur Entwicklungsumgebung PRINCE [Ins04a] zu finden.

Bevor der eigentliche Entwurf angegangen werden kann, muss das elektronische Laborbuch mit Daten gefüllt werden. Dies geschieht durch Eingabe von Rezepten, Materialien und Effekten über die entsprechenden Eingabemasken. Ist dies geschehen, kann mit dem Entwurf begonnen werden. Dazu öffnet der Entwerfer den Prozessfolgen Editor und erstellt eine neue Prozessfolge. Die Prozessfolge setzt er aus den im elektronischen Laborbuch vorhanden Prozessschritten zusammen. Initial wird sicherlich nicht jeder Prozessschritt fixiert sein, so dass die abstrakten Prozessschritte in den ersten Entwürfen zum Zuge kommen. Um sich zu vergewissern, dass die Prozessfolge rein technolo-

gisch überhaupt realisierbar ist, kann er einen Konsistenz-Check durchführen. Der Konsistenz-Checker wird ihm dann eventuelle Probleme anzeigen.

Steht das Grundgerüst der Prozessfolge, gibt es die Möglichkeit mit Hilfe von PRINCE für die abstrakten Prozessschritte geeignete reale zu finden. Je nachdem wie groß das Portfolio der im elektronischen Laborbuch gespeicherten Prozessschritte ist, stellt sich dem Entwickler eine mehr oder weniger große Auswahl dar. Die fertige Prozessfolge kann dann über den Simulator im Layer Editor als Schichtfolge angezeigt werden. Zuvor müssen allerdings noch zu eventuellen Lithografieschritten die passenden Masken angegeben werden. Dies geschieht ebenfalls im Layer Editor.

8.4 Fazit

Die Entwicklungsumgebung PRINCE ist das erste System für die Mikrosystemtechnik, das eine durchgängige Unterstützung für den fertigungsnahen Entwurf bietet. Durch die Entwicklung in Kooperation mit einem der größten Hersteller von Mikrosystemen ist gewährleistet, dass die Anforderungen und Belange der betroffenen Entwickler beim Entwurf des Systems berücksichtigt wurden.

Durch seine innovative Architektur ermöglicht PRINCE den nebenläufigen Entwurf, bei dem viele Personen an einem Gesamtprojekt arbeiten (concurrent engineering). Dies geschieht durch eine Client-Server-Architektur, die es möglich macht, dass die Entwickler standort- und zeitunabhängig auf die Software und damit auf die Informationen zugreifen können.

PRINCE setzt die Entwurfsmethodik um, die durch das erweiterte Kreismodell beschrieben wird. Die Voraussetzung für die Methodik ist die Realisierung einer geeigneten Datenhaltung. Diese wird mit dem elektronischen Laborbuch den Entwicklern an die Hand gegeben. Darauf aufbauend ermöglicht der Prozessfluss Editor die Entwicklung von Prozessfolgen auf beliebigen Abstraktionsebenen. Der Konsistenz-Checker sorgt dafür, dass, bei einer geeigneten Regelbasis im elektronischen Laborbuch, die Prozessfolge technologisch fertigbar bleibt.

Mittels des vorgestellten Simulators in Kombination mit dem Layer Editor ist es möglich, Prozessfolgen auf einfache Art zu simulieren. Dabei wurde sichergestellt, dass auch Prozessfolgen simuliert werden können, für deren Prozessschritte nur lückenhaft Modelle vorhanden sind. Durch die Hierarchisierung der Prozessschritte und Modelle wird dann dem Entwickler zumindest ein Ergebnis geliefert, das qualitativ richtig ist, also sich nur in Details von der Realität unterscheidet.

Über Schnittstellen ist die Anbindung von externen Werkzeugen möglich, was am Beispiel eines eigens entwickelten Simulators auf Basis eines zellulären Automaten vorgeführt wurde. Dieser zelluläre Automat wurde genau auf die Anforderungen der Mikrosystemtechnik angepasst. Durch die entwickelten generischen Algorithmen für die einzelnen Prozessschritttypen ist es sehr einfach, weitere Spezialmodelle für die Simulation zu implementieren.

Die Relevanz der Entwicklungsumgebung PRINCE wird außerdem dadurch unterstrichen, dass die Europäische Union ein Folgeprojekt aufbauend auf PRINCE mit namhaften Partnern aus Forschung und Industrie fördert [Eur04]. Näheres dazu jedoch im folgenden Ausblick.

9 Ausblick

In dieser Arbeit wurde ein Gesamtkonzept für die fertigungsnahe Entwurfsunterstützung in der Mikrosystemtechnik vorgestellt. Das Konzept besteht einerseits aus einer geeigneten Methodik und andererseits aus der Umsetzung derselben in einer Entwicklungsumgebung. Es wurde an Hand eigener Untersuchungen und Veröffentlichungen in der Fachwelt gezeigt, dass die vorhandene Toollandschaft eben diesen fertigungsnahen Entwurf nur unzureichend unterstützt.

Untersucht wurden verschiedene Ansätze für die Entwurfsmethodik. Als Lösung wurde das erweiterte Kreismodell vorgestellt, das als methodische Grundlage für den fertigungsnahen Mikrosystementwurf dienen kann. Basierend auf dieser Entwurfsmethodik wurde ein Unterstützungskonzept entwickelt und vorgestellt, um die Lücke in der Toollandschaft zu schließen. Das Konzept gliedert sich auf in die zwei Bereiche der Datenhaltung und der direkten Entwurfsunterstützung entsprechend dem erweiterten Kreismodell.

Für die Daten- und Informationsgewinnung wurde die zentrale Datenhaltung eingeführt. Diese wird durch das hier erstmals vorgestellte elektronische Laborbuch realisiert, das es nun ermöglicht, die Daten und Informationen zwischen den Entwicklern einfach, standort- und zeitunabhängig auszutauschen und persistent zu machen.

Weiterhin wurden die Anforderungen an die einzelnen Unterstützungsmodule definiert, die für die Umsetzung des Kreismodells von Nöten sind. Für die Vereinfachung der Datenhaltung, Simulation und Synthese wurden Abstraktions- und Hierarchisierungsstrategien in weiten Bereichen der Daten- und Informationshaltung eingeführt. Diese Strategien ermöglichen es erstmals, einen Entwurf trotz eventuell fehlender Informationen zumindest eingeschränkt weiter zu führen. Sowohl die Simulation als auch die Synthese profitieren von einem solchen Vorgehen.

Als Proof-of-Concept wurde das PRINCE-System in enger Kooperation mit der Industrie entwickelt. PRINCE vereint das elektronische Laborbuch mit den Unterstützungsmodulen und realisiert damit die Entwurfsmethodik des Kreismodells. Die durchweg positive Resonanz der Industrie auf PRINCE wird noch dadurch bestärkt, dass das hier vorgestellte Konzept in einem Folgeprojekt namens PROMENADE von der Europäischen Union gefördert wird. Zu diesem Zweck haben sich namhafte Partner aus den Bereichen TCAD-Softwareentwicklung, Simulation und Technologie zusammen geschlossen, um die prototypenhafte Implementierung von PRINCE in ein marktfähiges System zu überführen.

In diesem Kontext wird auch die in PRINCE noch fehlende Synthese von Schichtfolgen realisiert, so dass mit dem Abschluss von PROMENADE ein System auf dem Markt sein wird, das den fertigungsnahen Entwurf von Mikrosystemen vollständig unterstützt. Eine detaillierte Beschreibung des PROMENADE-Systems ist in [Eur04] zu finden.

Als ein weiteres aus dieser Arbeit resultierendes Forschungsgebiet kann die Anbindung des verhaltensnahen Entwurfs gesehen werden. Nicht nur der fertigungsnahe sondern auch der verhaltensnahe Entwurf ist, wie in dieser Arbeit schon erwähnt, von Fertigungsdaten abhängig. Denkbar ist hier eine Ankopplung über das elektronische Laborbuch, wie es in Abbildung 9.1 dargestellt ist. Die Abbildung zeigt rechts den fertigungsnahen und links den verhaltensnahen Bereich des Mikrosystementwurfs. Im Zentrum steht das elektronische Laborbuch. Über eben dieses können dann Technologiedaten auch für den verhaltensnahen Entwurf bereit gestellt werden und so die Modellbildung beschleunigen.

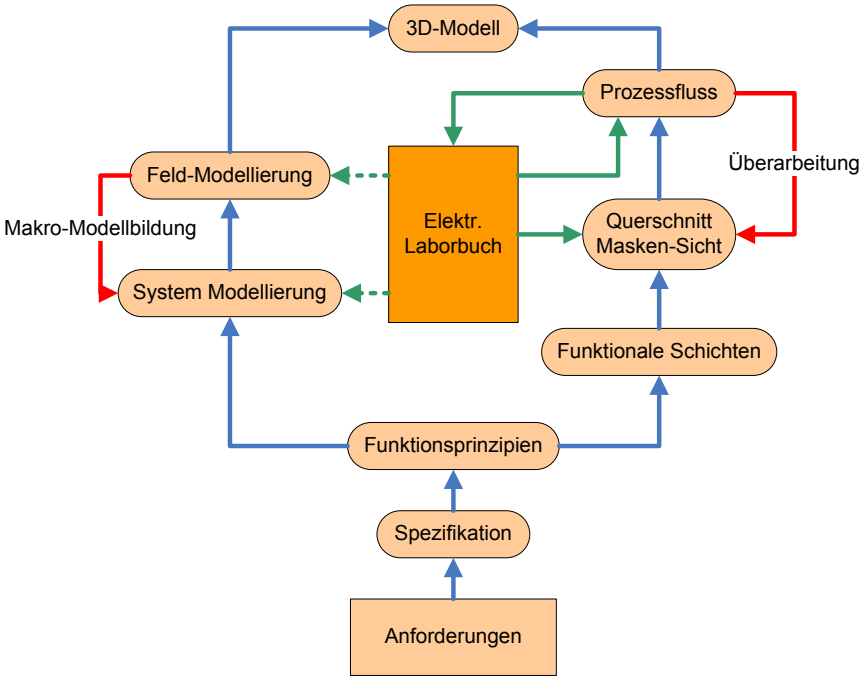


Abbildung 9.1: Unterstützung des verhaltensnahen Entwurfs

Abbildungsverzeichnis

Literaturverzeichnis

[Ant96a] ANTONSSON, Erik K.: Structured Design Methods for MEMS / California Institute of Technology, Pasadena, CA. 1996. – Forschungsbericht

[Ant96b] ANTONSSON, Erik K.: Structured Design Methods for MEMS. In: ANTONSSON, Erik K. (Hrsg.): *Structured Design Methods for MEMS*. California Institute of Technology, 1996

[Ant01] ANTONSSON, Erik K.: Microsystem Design Synthesis. In: ANTONSSON, Erik K. (Hrsg.) ; CAGAN, Jonathan (Hrsg.): *Formal Engineering Design Synthesis*. Cambridge University Press, 2001, S. 126–169

[AS96] ANANTHASURESH, G. K. ; SENTURIA, Stephen D.: Structured Design Methodology for MEMS. In: ANTONSSON, Erik K. (Hrsg.): *Structured Design for MEMS*. California Institute of Technology, 1996

[Bau99] BAUSELLS, Joan: *SensoNor Foundry Processes - Design Handbook Version 2.1*. SensoNor asa., 1999

[BRS01] BRÜCK, Rainer ; RIZVI, Nadeem ; SCHMIDT, Andreas: *Angewandte Mikrotechnik: LIGA - Laser - Feinwerktechnik*. Hanser Verlag, 2001

[Brü96] BRÜCK, Rainer. *Der fertigungsnahe Entwurf von Mikrosystemen*. Habilitation, Universität Dortmund. 1996

[Cae] CAESAR, Gaius J.: *De Bello Gallico*. Reclam

[Caw03] CAWSEY, Alison: *Künstliche Intelligenz im Klartext*. Pearson Studium, 2003

[CB00] CATTELL, R. G. G. ; BARRY, Douglas K.: *The Object Data Standard ODMG 3.0*. Morgan Kaufmann Publishers, 2000

[CDK02] COULOURIS, George ; DOLLIMORE, Jean ; KINDBERG, Tim: *Verteilte Systeme, Konzepte und Design*. 3., überarbeitete Auflage. Pearson Studium, 2002

[Cov04] COVENTOR INC. *Coventor Home Page*. http://www.coventor.com. November 2004

[EHM96] ECKER, Wolfgang ; HOFMEISTER, Michael ; MÄRZ-RÖSSEL, Sabine: The Design Cube: A Model for VHDL Designflow Representation and its Application. In: WAXMAN, Ronald (Hrsg.) ; BERGÉ, Jean-Michel (Hrsg.) ; LEVIA, Oz (Hrsg.) ; ROUILLARD, Jaques (Hrsg.): *High-Level System Modeling: Specification and Design Methodologies*. Kluwer Academic Publishers, 1996

[Eur04] EUROPEAN COMMISSION. *PROMENADE - Process management and design system for microsystem technologies*. Specific targeted research or innovation project, Contract 507965, IST-2002-2.3.1.2. 2004

[FCF02] FARLEY, Jim ; CRAWFORD, William ; FLANAGAN, David: *Java Enterprise in a Nutshell*. 2nd Edition. O'Reilly, 2002

[Fed96] FEDDER, Gary K.: Structured Design Methodology for MEMS. In: ANTONSSON, Erik K. (Hrsg.): *Structured Design Methods for MEMS*. California Institute of Technology, 1996

[Fed99] FEDDER, Gary K.: Structured Design of integrated MEMS. In: *Technical Digest of the IEEE International Conference on Micro Electro Mechanical Systems, Orlando*, 1999. – MEMS 99

[Fed00] FEDDER, Gary K.: Top-Down Design of MEMS. In: *Proceedings of the International Conference on Modeling and Simulation of Microsystems, San Diego*, 2000. – MSM 2000

[Fed03] FEDDER, Gary K.: Issues in MEMS Macromodeling. In: *Proceedings of the IEEE/ACM Int. Workshop on Behavioral Modeling and Simulation, San Jose*, 2003. – BMAS 2003

[FR97] FATIKOW, S. ; REMBOLD, U.: *Microsystem Technology and Microrobotics*. Springer-Verlag Berlin Heidelberg, 1997

[Frü04] FRÜHAUF, Joachim. *Simode*. www.infotech.tu-chemnitz.de/~wetel/simode.htm. Oktober 2004

[GD97] GERLACH, G. ; DÖTZEL, W.: *Grundlagen der Mikrosystemtechnik*. Carl Hanser Verlag München, 1997

[GDWL92] GAJSKI, Daniel D. ; DUTT, Nikil ; WU, Allen ; LIN, Steve: *High-Level Synthesis: Introduction to Chip and System Design*. Kluwer Academic Publishers, 1992

[GK83] GAJSKI, D. D. ; KUHN, R. H.: New VLSI Tools - Guest Editor's Introduction. In: *IEEE Computer* Bd. 16. 1983

[GR01] GIELEN, G. ; RUTENBAR, R. A.: Synthesis of Analog and Mixed-Signal Integrated Electronic Circuits. In: ANTONSSON, Erik K. (Hrsg.) ; CAGAN, Jonathan (Hrsg.): *Formal Engineering Design Synthesis*. Cambridge University Press, 2001

[GT88] GAJSKI, Daniel D. ; THOMAS, Donald E.: Introduction to Silicon Compilation. In: GAJSKI, Daniel D. (Hrsg.): *Silicon Compilation*. Addison-Wesley, 1988, S. 1–48

[GVNG94] GAJSKI, Daniel D. ; VAHID, Frank ; NARAYAN, Sanjiv ; GONG, Jie: *Specification and Design of Embedded Systems*. Prentice Hall, 1994

[Hah98a] HAHN, Kai: *Methoden und Werkzeuge zur fertigungsnahen Entwurfsverifikation in der Mikrotechnik*, Universität Siegen, Diss., 1998

[Hah98b] HAHN, Kai: Die Prozessbeschreibungssprache LIDO-PDL / Universität Dortmund. 1998 (695). – Forschungsbericht

[Hor02] HORENSTEIN, Mark N.: *Design Concepts for Engineers*. Second Edition. Prentice Hall, 2002

[Hub96] HUBBARD, Ted J.: Structured Design Methodology for MEMS. In: ANTONSSON, Erik K. (Hrsg.): *VLSI and MEMS, VLSI vs. MEMS*. California Institute of Technology, 1996

[HW03] HAHN, K. ; WAGENER, A.: *Considerations for MEMS physical design stages*. Invited Tutorial, Sophia Antipolis Microelectronics, Sophia Antipolis. 2003. – SAME 2003

[HWPB03] HAHN, K. ; WAGENER, A. ; POPP, J. ; BRÜCK, R.: Process Management and Design for MEMS and Microelectronics Technologies. In: *Proceedings of SPIE: Microelectronics: Design, Technology, and Packaging, Perth* Bd. 5274, 2003. – SPIE Perth 2003

[Ins04a] INSTITUT FÜR MIKROSYSTEMTECHNIK: *PRINCE Benutzerhandbuch*. Universität Siegen, Dezember 2004

[Ins04b] INSTITUT FÜR MIKROTECHNIK MAINZ GMBH. *Die LIGA-Technik*. http://imm-mainz.de. 2004

[Jac62] JACCODINE, R.: Use of modified free energy theorems to predict equilibrium growing and etching shapes. In: *Journal of Applied Physics* Bd. 33, 1962

[JKH99] JANTSCH, Axel ; KUMAR, Shashi ; HEMANI, Ahmed: The Rugby Model: A Conceptual Frame for the Study of Modelling, Analysis and Synthesis Concepts of Electronic Systems. In: *Proceedings of Design, Automation and Test in Europe*, 1999. – DATE 1999

[Kas00] KASPER, Manfred: *Mikrosystementwurf: Entwurf und Simulation von Mikrosystemen*. Springer Verlag, Berlin Heidelberg, 2000

[Kie98] KIELKOWSKI, Ron M.: *Inside SPICE*. Second Edition. McGraw-Hill, 1998

[KS04] KÄMPF, Andreas ; STOCK, Andre. *Simulation eines Oxidationsprozesses mit Hilfe Zellulärer Automaten*. Seminar Entwurfsunterstützung und Simulation in der Mikrosystemtechnik, Universität Siegen. 2004

[Kun00] KUNDERT, Ken: Top-down tops for mixed-signal. In: *EE Times* (2000). – http://www.eedesign.com/

[LN04] LENTZER, A. ; NEUL, R. *Ergebnisse der Umfrage zum Werkzeugeinsatz und Werkzeugbedarf beim Entwurf von Mikrosystemen*. 2004

[Lor99] LORENZ, Gunar: *Netzwerksimulation mikromechanischer Systeme*. Shaker Verlag, Aachen, 1999. – Dissertation, Universität Bremen

[Lug01] LUGER, George F.: *Künstliche Intelligenz, Strategien zur Lösung komplexer Probleme*. Pearson Studium, 2001

[MA00] MA, Li ; ANTONSSON, Erik K.: Automated Mask-Layout and Process Synthesis for MEMS. In: *Proceedings of the International Conference on Modeling and Simulation of Microsystems, San Diego*, 2000. – MSM 2000

[Mad02] MADOU, Marc: *Fundamentals of Microfabrication*. Second Edition. CRC Press, 2002

[Mas04] MASTRANGELO, Carlos H. *The MISTIC TCAD Home Page*. http://www.eecs.umich.edu/mistic/. Februar 2004

[MB93] MENZ, Wolfgang ; BLEY, Peter: *Mikrosystemtechnik für Ingenieure*. VCH Verlagsgesellschaft mbH, Weinheim, 1993

[MC80] MEAD, Carver ; CONWAY, Lynn: *Introduction to VLSI systems*. Addison-Wesley, Reading, Massachusetts, 1980

[MF97] MUKHERJEE, Tamal ; FEDDER, Gary K.: Structured Design of Microelectromechanical Systems. In: *Proceedings of the Design Automation Conference, Anaheim*, 1997. – DAC 97

[MF98] MUKHERJEE, Tamal ; FEDDER, Gary K.: Design Methodology for Mixed-Domain Systems-on-a-Chip. In: *Proceedings of the IEEE Computer Society Workshop on VLSI*, 1998

[MFB99] MUKHERJEE, Tamal ; FEDDER, Gary K. ; BLANTON, R. D. (.: Hierarchical Design and Test of Integrated Microsystems. In: *IEEE Design and Test Magazine*, 1999

[MH04] MONSON-HAEFEL, Richard: *Enterprise JavaBeans*. 4th Edition. O'Reilly, 2004

[MHTA98] MARCHETTI, J. ; HE, Y. ; THAN, O. ; AKKARAJU, S.: Efficient process development for bulk silicon etching using cellular automata simulation techniques. In: *Proceedings of SPIE Symposium on Micromachining and Microfabrication, Micromachined Devices and Components, Santa Clara*, 1998

[MM02] MENZ, Wolfgang ; MOHR, Jürgen: The LIGA Microfabrication Technique. In: *Proceedings of the International Summer School on advanced microelectronics, Grenoble*, 2002. – MIGAS02

[MOM96] MIYAHARA, Y. ; OUMI, Y. ; MORIYAMA, S.: Design Methodology for Analog High Frequency ICs. In: *Proceedings of Design Automation Conference, Las Vegas*, 1996. – DAC 1996

[Moo65] MOORE, Gorden E.: Cramming more components onto integrated circuits. In: *Electronics* Bd. 38, 1965

[Na67] NATHANSON, H. C. ; ET AL.: The Resonant Gate Transistor. In: *IEEE Trans. Electron Devices* Bd. 14, 1967

[NEX02] NEXUS! TASK FORCE: *NEXUS Market Analysis*. NEXUS, 2002

[Pop04] POPP, Jens: *Ein Konzept für die Datenhaltung in der Mikrosystemtechnik*, Universität Siegen, Diss., 2004

[PSWH05] POPP, J. ; SCHMIDT, T. ; WAGENER, A. ; HAHN, K.: MEMS fabrication process management environment. In: *Proceedings of SPIE: Micromachining and Microfabrication Process Technology X, San Jose* Bd. 5715, 2005. – Photonics West 2005

[PWFB03] PLANINIC, Julijana ; WÜNNENBERG, Martin ; FREISCHLAD, Stefan ; BECKMANN, Achim. *Handbuch für den Prince-Sequenzeditor*. Projektgruppe Prince, Universität Siegen. 2003

[PWOB04] POPP, Jens ; WAGENER, Andreas ; ORTLOFF, Dirk ; BEUNDER, Mike: A Novel Approach Towards Standardization of MEMS Process Development. In: *Proceedings of the 9th International Conference on the Commercialization of Micro and Nano Systems, Edmonton*, 2004. – COMS 2004

[QC94] QUIBELDEY-CIRKEL, Klaus: *Das Objekt-Paradigma in der Informatik*. Teubner Verlag, 1994

[Ram89] RAMMIG, Franz J.: *Systematischer Entwurf digitaler Systeme*. B. G. Teubner, Stuttgart, 1989

[RBH+02] REITZ, Sven ; BATIAN, Jens ; HAASE, Joachim ; SCHNEIDER, Peter ; SCHWARZ, Peter: System Level Modeling of Microsystems using Order Reduction Methods. In: *Proceedings of Design, Test, Integration and Packaging of MEMS and MOEMS, Cannes*, 2002. – DTIP2002

[Sch04] SCHMIDT, Thilo: *Konzeption und Implementierung einer Material- und Effektdatenbank für Simulation und Analyse in der Mikrosystemtechnik*, Universität Siegen, Diplomarbeit, 2004

[Sen98] SENTURIA, Stephen D.: CAD Challenges for Microsensors, Microactuators ,and Microsystems. In: *Proceedings of the IEEE* Bd. 86, 1998

[Sen01] SENTURIA, Stephen D.: *Microsystem Design*. Kluwer Academic Publishers, 2001

[SGCD02] DA SILVA, M. G. ; GIASOLLI, R. ; CUNNINGHAM, S. ; DEROO, D.: MEMS Design for Manufacturability (DFM). In: *Proceedings of IEEE Sensors Expo and Conference, Orlando*, 2002. – Sensors 2002

[Sil04] SILVACO. *Silvaco Website*. www.silvaco.com. Februar 2004

[Som01] SOMMERVILLE, Ian: *Software Engineering*. Pearson Studium, 2001

[Stu04] STUFF, Alexander: *Entwicklung und Implementierung eines Algorithmus zur Überprüfung und Sicherstellung der Konsistenz von Prozessfolgen in der Mikrosystemtechnik*, Universität Siegen, Diplomarbeit, 2004

[Sun03] SUN MICROSYSTEMS INC. *Java 2 Platform Enterprise Edition Specification, v1.4*. http://java.sun.com/j2ee/. 2003

[Sun04a] SUN MICROSYSTEMS INC. *Java Web Start Homepage*. http://java.sun.com/products/webstart/. Februar 2004

[Sun04b] SUN MICROSYSTEMS INC. *Sun Microsystems*. http://www.sun.com/. 2004

[SWPH04] SCHMIDT, T. ; WAGENER, A. ; POPP, J. ; HAHN, K.: Technology Interfaces to Microsystem and Nanoelectronic Processes. In: *Proceedings of SPIE: Smart Structures, Devices, and Systems II, Sydney* Bd. 5649, 2004. – SPIE Sydney 2004

[TLMP02] TÉTELIN, A. ; LÉVI, H. ; MONGELLAZ, B. ; PELLET, C.: Behavioral Modeling of a Humidity Sensor using an Analog Hardware Description Language. In: *Proceedings of the International Conference on Modeling and Simulation of Microsystems, San Juan*, 2002. – MSM 2002

[Ull02] ULLMAN, David G.: *The Mechanical Design Process*. 3. Edition. McGraw-Hill, 2002

[Wag01] WAGENER, Andreas: *System- und Anforderungsanalyse für ein Prozessdesign-Werkzeug in der Mikrosystemtechnik*, Universität Siegen, Diplomarbeit, 2001

[WH03] WAGENER, A. ; HAHN, K.: Eine Entwurfsmethodik für die Mikrosystemtechnik und Post-CMOS. In: *Proceedings Austrochip 2003, Linz*, 2003. – Austrochip 2003

[Wor03] WORLD WIDE WEB CONSORTIUM (W3C). *Mathematical Markup Language (MathML) Version 2.0, W3C Recommendation 21 October 2003*. http://www.w3.org/TR/MathML/. 2003

[Wor04] WORLD WIDE WEB CONSORTIUM (W3C). *Extensible Markup Language (XML) 1.0, W3C Recommendation 04 February 2004*. http://www.w3.org/TR/REC-xml. 2004

[WPH03] WAGENER, A. ; POPP, J. ; HAHN, K.: Prince - Process Information and Management Center. In: *Proceedings Micro System Technologies 2003, München*, 2003. – MST 2003

[WPH04] WAGENER, A. ; POPP, J. ; HAHN, K.: Process design environment for microfabrication technologies. In: *Proceedings of SPIE: Micro-*

machining and Microfabrication Process Technology IX, San Jose Bd. 5342, 2004. – Photonics West 2004

[WPHB02a] WAGENER, A. ; POPP, J. ; HAHN, K. ; BRÜCK, R.: PDML - A XML-Based Process Description Language. In: *Proceedings of the 9th European Concurrent Engineering Conference, Modena*, 2002. – ECEC 2002

[WPHB02b] WAGENER, A. ; POPP, J. ; HAHN, K. ; BRÜCK, R.: Requirements to a physical design support tool for microsystem technology. In: *Proceedings of the 14th European Simulation Symposium, Dresden*, 2002. – ESS 2002

[WPHB04] WAGENER, A. ; POPP, J. ; HAHN, K. ; BRÜCK, R.: PRINCE - Design Tools for Micro System Fabrication Management. In: *10. GMM Workshop: Methoden und Werkzeuge für den Entwurf von Mikrosystemen, Cottbus*, 2004. – GMM 2004

[WT85] WALKER, R. A. ; THOMAS, D. E.: A Model for Design Representation and Synthesis. In: *Proceedings of the Design Automation Conference*, 1985

[Zam97] ZAMAN, Mohammed H.: *Process compilation methods for thin film devices*, Universität Michigan, Diss., 1997

[ZCM99a] ZAMAN, Mohammed H. ; CARLEN, Edwin t. ; MASTRANGELO, Carlos H.: Automatic Generation of Thin Film Process Flows - Part I: Basic Algorithms. In: *Proceedings of IEEE Transactions on Semiconductor Manufacturing* Bd. 12, 1999

[ZCM99b] ZAMAN, Mohammed H. ; CARLEN, Edwin t. ; MASTRANGELO, Carlos H.: Automatic Generation of Thin Film Process Flows - Part II: Recipe Generation, Flow Evaluation, and System Framework. In: *Proceedings of IEEE Transactions on Semiconductor Manufacturing* Bd. 12, 1999

[ZD02] ZHA, Xuan F. ; DU, H.: Web-based knowledge-intensive support framework for collaborative design of MEMS. In: *Journal of Micromechanics and Microengineering* 12 (2002)

[ZD03] ZHA, Xuan F. ; DU, H.: Manufacturing process and material selection in concurrent collaborative design of MEMS devices. In: *Journal of Micromechanics and Microengineering* 13 (2003)

[ZT00] ZIENKIEWICZ, O. C. ; TAYLOR, R. L.: *The Finite Element Method*. Bd. 1: The Basis. fifth edition. Butterworth-Heinemann, Oxford, 2000