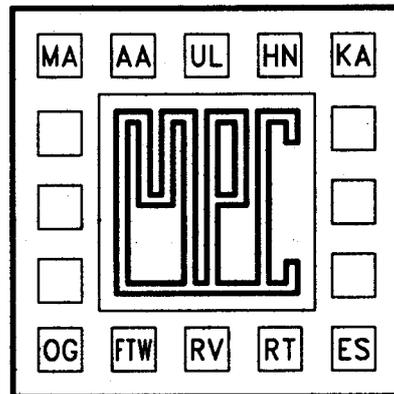


MULTIPROJEKT CHIP-GRUPPE

BADEN - WÜRTTEMBERG

WORKSHOP JULI 1993

RAVENSBURG-WEINGARTEN



HERAUSGEBER: FACHHOCHSCHULE ULM

© 1993 Fachhochschule Ulm

Das Werk und seine Teile sind urheberrechtlich geschützt.
Jede Verwertung in anderen als den gesetzlich zugelassenen
Fällen bedarf deshalb der vorherigen schriftlichen Einwilligung
des Herausgebers.

Inhaltsverzeichnis

1. Zur Synthese analoger ICs Seite 3
G. Forster, FH Ulm

2. Fortschritte in der Entwicklung von Seite 11
Anwendungsspezifischen
Integrierten Schaltungen (ASIC)
D. Jansen, FH Offenburg

3. Technologie-transparentes ASIC Design Seite 25
auf einem Low-Cost Entwicklungssystem
J. Gandowitz, IMS Stuttgart

4. Realisierung großer Netze Seite 43
M. Faas, P. Leber, W. Rülling, FH Furtwangen

5. Ein Demonstrationssystem für die Bilddatenübertragung Seite 53
N. Höptner, FH Pforzheim

6. Aufbau und Spice-Simulation von nichtlinearen Seite 73
chaotischen Schaltungen am Beispiel von Chua's Circuit
B. Maurer, R. Hägele, H. Khakzar, FH Esslingen

7. Erstellen von Hardmakros und Aufbau einer Zellbibliothek Seite 93
unter Verwendung des ES2-Library-Kits
D. Pouw, D. Jansen, FH Offenburg

8. Ein Praktikum zur Schaltungsintegration Seite 101
M. Faulhaber, A. Führer, M. Rathenow, FH Ulm

Zur Synthese analoger ICs

Prof. G. Forster

Fachhochschule Ulm
Eberhard-Finck-Str. 11
89075 Ulm

Einleitung

Wirtschaftlich erfolgreiche ASIC-Entwicklung ist heute gekennzeichnet durch immer kürzere Entwicklungszeiten bei weiterhin steigender Schaltungskomplexität. Diese Anforderungen sind nur zu erfüllen, wenn die erforderlichen Arbeitsgebiete:

- System- und Schaltungsentwicklung
- Design, Layout
- Wafer processing
- Meßtechnische Untersuchung

weiter formalisiert und automatisiert werden.

Im Bereich der Digitalschaltungen wurde auf diesem Gebiet ein beachtlicher Stand erreicht. Mit FPGAs, Gate Arrays und Standardzellverfahren kann, bei Einschränkung der Freiheitsgrade, der Aufwand für die ersten zwei bzw. drei Arbeitsgebiete entscheidend reduziert werden. Die Vorteile liegen somit in der besonders schnellen und kostengünstigen Entwicklung [1].

Auf dem Gebiet der Massenprodukte steht jedoch neben der kurzen Entwicklungszeit die Forderung nach hoher Packungsdichte und optimaler Ausnutzung der technologischen Möglichkeiten im Vordergrund. Dies sind aber gerade die Merkmale des full custom designs.

So entstand ein großer Bedarf zur Entwicklung von Synthesewerkzeugen, welche ausgehend von der Systembeschreibung in einer Hochsprache die Schaltungs- und Layoutentwicklung mit Hilfe parametrisierbarer Blöcke automatisiert bzw. interaktiv ermöglichen [2]. Bei extrem kurzer Entwurfszeit werden damit annähernd auch die technischen Daten eines Hand-Designs erzielt.

Stand der Automatisierung in der Analog-IC-Entwicklung

Mit den sog. Linear Arrays ist es auch im Analog-Bereich möglich, die Entwicklungsphase schneller und kostengünstiger zu durchlaufen. Bild 1 zeigt das Array B1000S von TEMIC, dessen Einsatzbereich bei Schaltungen bis etwa 3 GHz liegt [3]. Es eignet sich hervorragend für Anwendungen bei ge-

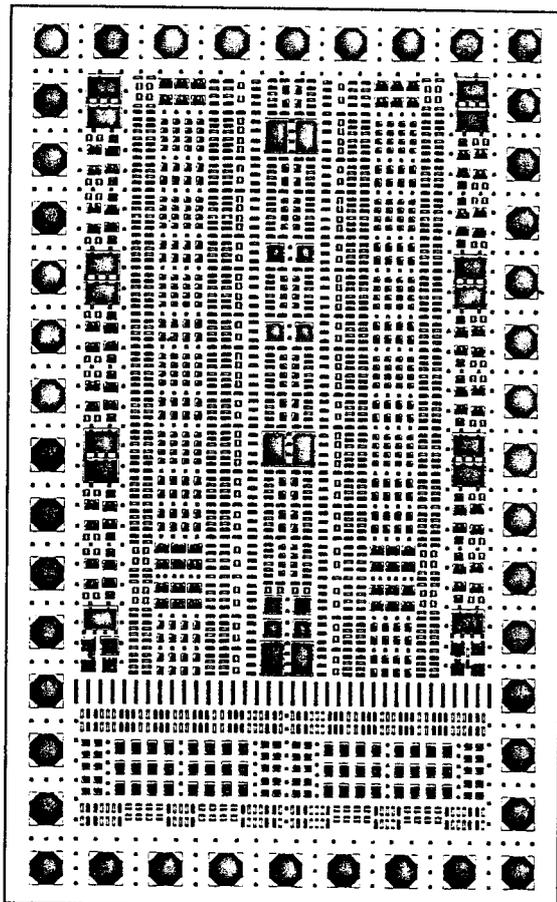


Bild 1: Linear Array B1000S

ringen Stückzahlen oder für System- und Schaltungsuntersuchungen wegen der geringen Maskenkosten und Durchlaufzeit. Automatisierte Ver-

drahtung wird auf diesem Gebiet allerdings kaum eingesetzt.

Auch das Standardzellverfahren ist bei Analog-ICs durchaus üblich. Bild 2 zeigt hierzu als Bei-

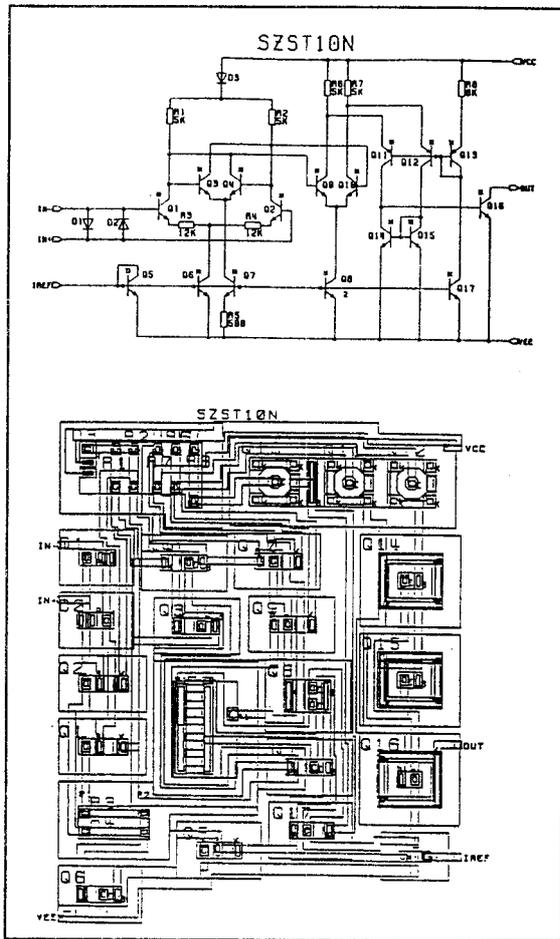


Bild 2: Bipolar-Standardzelle aus ADASIC

spiel einen Analog-Komparator aus der Standardzellbibliothek "ADASIC" von AEG [4].

Die Layouts aller Zellen zeigen in einer Dimension gleiche Abmessungen. Beim Einsatz einer zweiten Verdrahtungsebene eignen sie sich deshalb zur automatisierten Platzierung und Verdrahtung.

Ein wesentlicher Nachteil des Standardzellverfahrens ist allerdings wie im Digitalbereich die mangelnde Flexibilität mit vorgegebenen

Zellen. Gerade Analog-Design ist jedoch in besonderem Maße auf Flexibilität angewiesen.

Dieser Problematik versucht ein Ansatz von CSEM unter dem Namen "IDAC" gerecht zu werden. Hier stehen in einer Bibliothek parametrisierbare Makrozellen zur Verfügung [5]. Für jede dieser Zellen (OPV, Komparator, Oszillator usw.) steht ein Datenblatt mit Standardeinstellungen zur Verfügung. Entsprechend der Änderung dieses Datenblatts versucht das System eine automatisierte Dimensionierung der Zelle. Die Dimensionierung betrifft Transistortyp und W/L-Verhältnisse der MOS-Transistoren, ohne die Struktur der Gesamtschaltung einer Zelle zu verändern. So wird ermöglicht, daß auch das Layout als parametrisierbare Zelle vorliegt.

Ein weiteres Synthesewerkzeug wird unter dem Namen "Explorer AutoFilter" von Mentor Graphics angeboten. Es befaßt sich allerdings nur mit dem eng umgrenzten Bereich der SC-Filter [6]. Hier liegt eine Standardisierung auf der Hand, weil nur eine geringe Anzahl von parametrisierbaren Standardzellen erforderlich ist (CMOS-OPV, Kondensatoren, Schalter, Taktgenerator).

Anforderungen an ein Analog-Synthesewerkzeug

Beim Gedanken an ein globales Analog-Synthesesystem liegt, in Anlehnung an den digitalen Silicon Compiler, die Vorstellung eines TOP-DOWN-Entwurfs natürlich sehr nahe. Ausgangspunkt sollte also die Blockschaltung für ein komplettes IC oder zumindest größerer Teile davon sein. Ein Beispiel hierzu wäre ein kompletter AGC-Verstärker (Bild 3) für ein Telekommunikations-IC, bestehend aus Operationsverstärker,

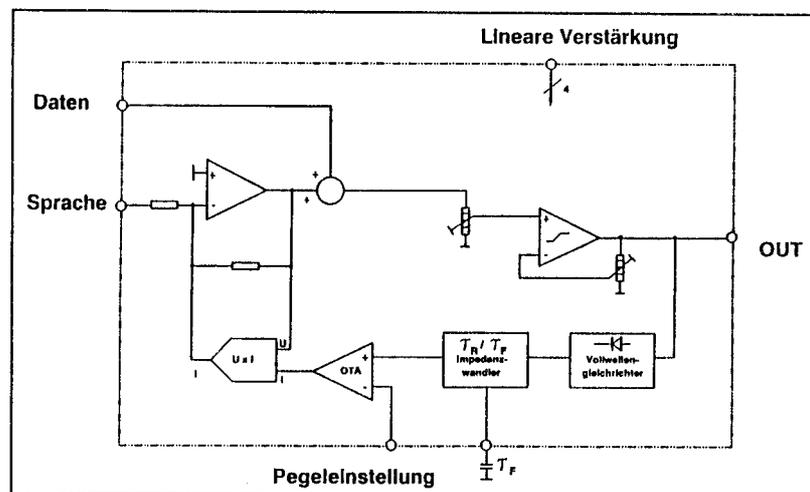


Bild 3: Blockschaltbild aus Makrozellen

Multiplizierer, OTA, Gleichrichter, etc. [7]. Dieser Entwurf sollte noch nicht technologiegebunden sein, damit auch im Architekturstadium von einer Technologie auf die andere "umgeschaltet" werden kann. Es versteht sich von selbst, daß das System lernfähig sein muß, damit sich die Wissensbasis mit jeder erfolgreichen Synthese erweitert.

Im Vergleich zur Digital-Synthese sind jedoch bei der Analog-Synthese eine ganze Reihe zusätzlicher Herausforderungen zu bewältigen:

- (1) Das Analogwissen ist viel weniger formalisiert.
- (2) Die Basiszellen (Transistoren, Widerstände) sind in ihrer analogen Natur zu erfassen.
- (3) Die Abhängigkeiten sind in der Regel nichtlinear, ausgehend von der Exponentialfunktion für den Bipolar-Transistor oder dem Übergang von der schwachen zur starken Inversion beim MOSFET.
- (4) Analogschaltungen sind im allgemeinen nicht rückwirkungsfrei.
- (5) Die Spezifikation von Makrozellen ist aufwendig. So sind z. B. für einen Analogkomparator neben den "digitalen" Daten Parameter wie Hysterese, Gleichtaktbereich, Kleinsignalverstärkung, Eingangsstrom, Eingangsoffset, Rauschen usw. zu spezifizieren. Diese Daten lassen sich nicht immer unmittelbar den Systemanforderungen entnehmen.
- (6) Design und Schaltungsentwicklung sind eng verzahnt, weshalb auch der Schaltungsentwickler in hohem Maße Designkenntnisse zu berücksichtigen hat.

Analogschaltungen müssen also neben der logischen Richtigkeit zusätzliche Kriterien erfüllen, wenn sie mit wirtschaftlicher Ausbeute gefertigt werden sollen. Hierzu lassen sich gewisse Regeln formulieren als Voraussetzung für "gutes" Analogdesign:

- (1) Die Schaltung betreffend:
 - Schaltung nur auf hohe Genauigkeit relativer Größen ausrichten.
 - Schaltungskonzepte mit hoher Versorgungsspannungsunterdrückung

suchen.

- Substratströme schaltungstechnisch vermeiden.
- Gehäuseparasiten bei der Schaltungsentwicklung berücksichtigen.

(2) Das Design betreffend:

- Versorgungen von Endstufen und Verstärkerstufen entkoppeln.
- Erhöhte Leitbahnwiderstände nur an Knoten mit hoher Quellimpedanz zulassen.
- Transistor- und Widerstandspaare auf Isothermen legen.
- GND-Sternpunkt vom Chip auf das Gehäuse verlagern.
- Kapazitive Kopplungen zwischen Leitbahnen vermeiden.

Nicht alle Regeln sind leicht einzuhalten. Besonders schwierig ist die Entscheidung, welche Regel aus Kostengründen zu vernachlässigen ist oder welcher Regel im konkreten Fall Vorrang vor einer anderen zu geben ist.

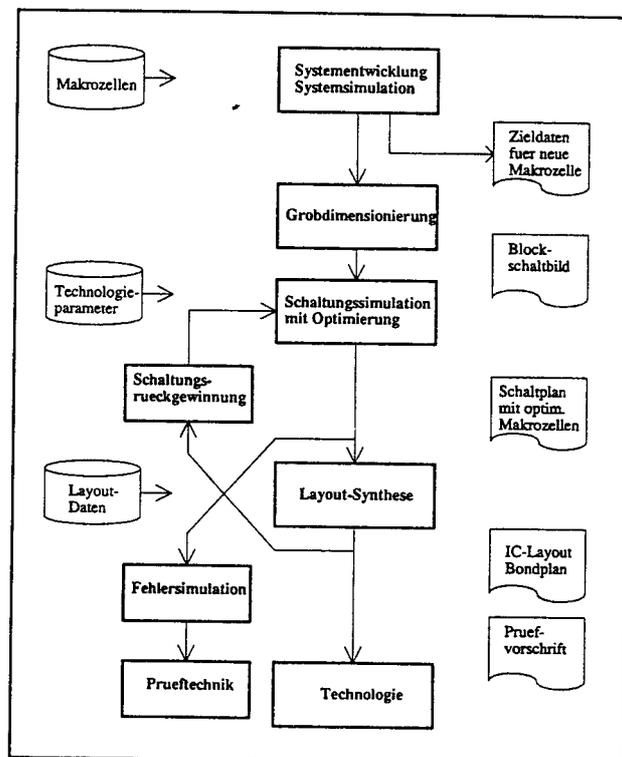


Bild 4: Konzept für die Analog-IC-Synthese

Konzept für die Analog-IC-Synthese

Der mögliche Ablauf einer Synthese läßt sich, auch unter Berücksichtigung obiger Anforderungen, mit dem in Bild 4 dargestellten Schema skizzieren. Die Systementwicklung erfolgt hiernach mit Hilfe der Systemsimulation, welche die Makrozellen (z. B. OPV) mittels Ersatzelementen (z. B. gesteuerten Quellen) beschreibt. Mit Hilfe eines Suchalgorithmus wird dann eine passende Makrozelle aus der Bibliothek ausgewählt. Der Makrozellenbibliothek sind zu diesem Zweck Daten zugeordnet, die den Eigenschaftsraum einer Makrozelle beschreiben. Ergebnis der Systemsimulation ist somit ein Blockschaltbild, bestehend aus parametrisierbaren Makrozellen mit Zieldaten, die mit den Makrozellen bei geeigneter Dimensionierung erreichbar sind. Die anschließende Grobdimensionierung führt dann zu einem Blockschaltbild mit Makrozellen, welche je einzeln auf ihre Zieldaten hin dimensioniert sind, allerdings ohne Berücksichtigung der Umgebung.

Die Gesamtschaltung kann nun mit aufgelösten Makrozellen simuliert werden. Da erwartet werden kann, daß in diesem Stadium die Gesamtspezifikation bereits annähernd erreicht ist, sollte sich mit Hilfe der linearen Optimierung das sog. "design centering" durchführen lassen. Die in der Layoutsynthese hinzukommenden parasitären Effekte werden über die Schaltungsrückgewinnung in die Optimierung zurückgeführt. Die zugehörigen Parameter (z. B. Schichtwiderstände) liegen in der Technologie-Bibliothek vor. Nach erfolgreicher Synthese gehen IC-Layout, Bondplan und Prüfvorschrift in die Technologie, wobei die Prüftechnik wieder Daten für die Technologieparameter-Bibliothek liefert.

Von entscheidender Bedeutung sind nun die verfügbaren Makrozellen. Realistischerweise muß man davon ausgehen, daß in der Regel die gesuchten Makros nicht zur Verfügung stehen, weil die Mannigfaltigkeit einfach zu groß ist. Die Systemsimulation ist dann lediglich in der Lage, Zieldaten für eine neu zu erstellenden Makrozelle abzugeben. Der entscheidende Schritt wird damit also die Synthese neuer Makrozellen sein.

Synthese analoger Makrozellen

Ausgehend von der Erkenntnis, daß es unmöglich sein wird, eine analoge Makrozellenbibliothek vorzuhalten, die den Bedarf bei der Analog-IC-Entwicklung auch nur annähernd deckt, ergibt sich ein Ansatz mit kleineren, universelleren Einheiten.

Als kleinste Einheit wären die Bauelemente (Transistoren, Widerstände, ...) anzusehen. Sie sind jedoch aus der schaltungstechnischen Betrachtung zu unspezifisch. Ein schaltungstechnischer Ansatz ist allenfalls denkbar mit den Transistor-Grundschaltungen (Emitter-, Basis-, Kollektorschaltung) oder größeren Einheiten. Als guter Kompromiß zwischen Universalität und schaltungstechnischer Funktionalität sind sicher Schaltungseinheiten wie Stromquellen, Differenzstufen, Lastelemente, Impedanzwandler etc. anzusehen.

Nach Bild 5 werden mit solchen Einheiten, hier als Skelettzellen bezeichnet, Makrozellen aufge-

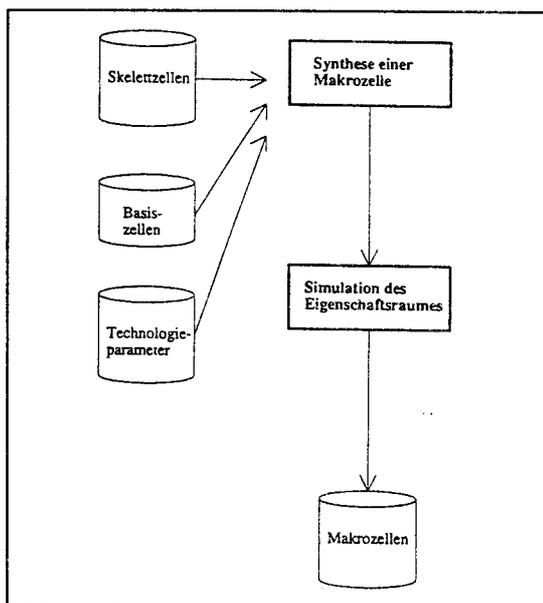


Bild 5: Synthese einer Makrozelle

baut. Zur Auswahl geeigneter Skelettzellen wird hierbei zusätzlich auf Transistor- und Technologieparameter zugegriffen. Nach erfolgter Synthese kann über ein automatisiert gesteuertes Simulationsprogramm, welches den Parameterbereich durchläuft, der Eigenschaftsraum der Makrozelle bestimmt werden [8]. Dieser Eigenschaftsraum kann dann als Beschreibung mit in die Makrozellenbibliothek aufgenommen werden.

Ein Beispiel für die mögliche Darstellung des Eigenschaftsraumes ist in Bild 6 dargestellt. An einem Operationsverstärker wurden einige Parameter in den zulässigen Grenzen schrittweise variiert. Für die sich ergebenden Parameterkombinationen sind dann jeweils die Werte für die OPV-Daten (im Beispiel Transitfrequenz und Leerlaufverstärkung) verfügbar. Wenn auch die Simulation vollautomatisch erfolgen kann, so wird doch im Einzelfall zu prüfen sein, ob sich der Aufwand im Vergleich zur Neusynthese lohnt.

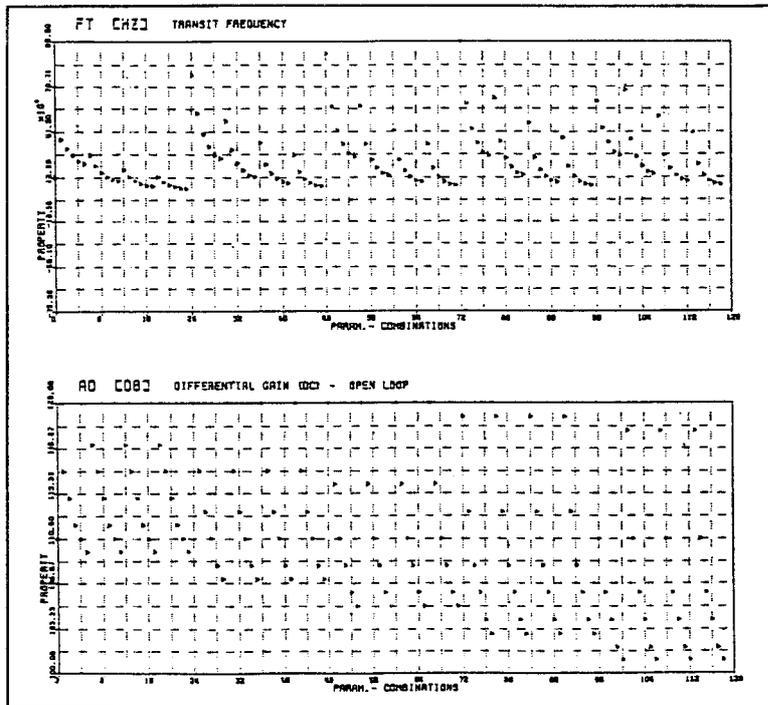


Bild 6: Eigenschaftsraum eines OPV

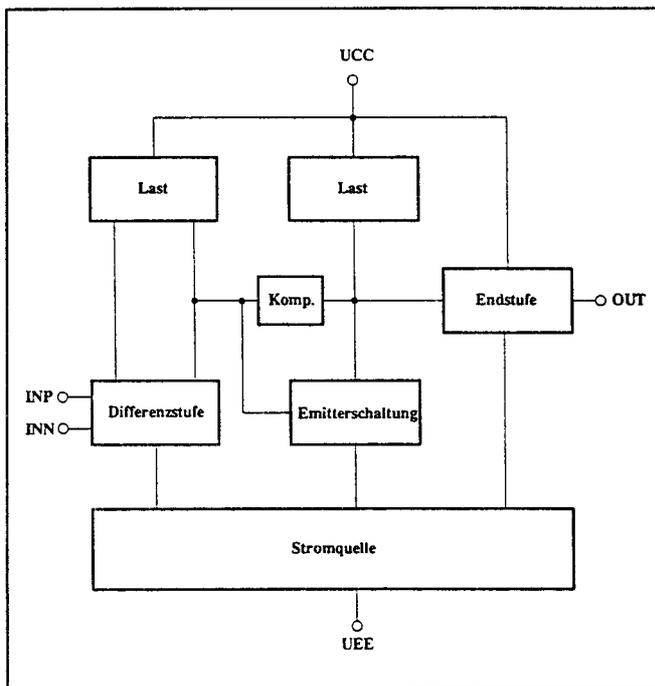


Bild 7: OPV, aufgebaut aus Skelettzellen

Synthese eines Operationsverstärkers

Anhand des Operationsverstärkers soll die Möglichkeit der Synthese, ausgehend von der Ebene der Skelettzellen, gezeigt werden. Nach Bild 7 setzt sich z. B. ein zweistufiger Operationsverstärker zusammen aus einer Differenzstufe mit Last-

element, einer zweiten Verstärkerstufe, bestehend aus Emitterschaltung (oder Differenzstufe) mit Last und einer Endstufe als Impedanzwandler. Zusätzlich wird eine Kompensationseinheit zur Frequenzgangkorrektur und eine Stromquelle für die Arbeitspunkteinstellung benötigt. Das Schema kann ergänzt werden durch Schutzschaltungen an den Signalpins. Ein einstufiger OPV läßt sich entsprechend aus einer Teilmenge dieser Skelettzellen aufbauen.

Der Ablauf der Synthese erfolgt dann nach Bild 8. Anhand der Spezifikation muß zunächst mit Hilfe von Auswahlregeln eine Entscheidung zugunsten einer der bekannten Topologien (im Beispiel einstufiger oder zweistufiger OPV) getroffen werden. Sollte die Entscheidung wie im Beispiel auf den einstufigen Verstärker gefallen sein, so ist bekannt, daß sich der OPV aus einer Differenzstufe, einer Last, einer Stromquelle, einer Endstufe und einer Kompensationseinheit zusammensetzen läßt. Der erste Schritt der Dimensionierung besteht nun darin, das Datenblatt für den OPV abzubilden auf die Datenblätter der einzelnen Skelettzellen. Aufgrund dieser Datenblätter ist es dann wiederum möglich, für die jeweilige Skelettzelle die geeignete Topologie zu wählen.

Dieser Vorgang wird im Bild 9 anhand der Stromquelle gezeigt. So wird das Expertensystem N-Typ- und P-Typ-Stromquellen kennen, darunter jeweils Quellen mit unterschiedlicher Zahl an Ausgängen, wobei wiederum unterschiedliche Topologien zur Auswahl stehen. Während die Stromquelle mit Emitterfolger den Einfluß der Stromverstärkung vermindert, erreicht die kaskodierte Stromquelle höhere Innenwiderstände. Nach Festlegung der Topologie ist bekannt, daß sich die Stromquelle im Beispiel aus drei Transistoren und zwei Widerständen (Wechselstromgegenkopplung) zusammensetzen läßt.

Die Aufgabe der Dimensionierung besteht nun wiederum darin, das Datenblatt für die Stromquelle abzubilden auf die Datenblätter der drei Transistoren und zwei Widerstände. Als Transi-

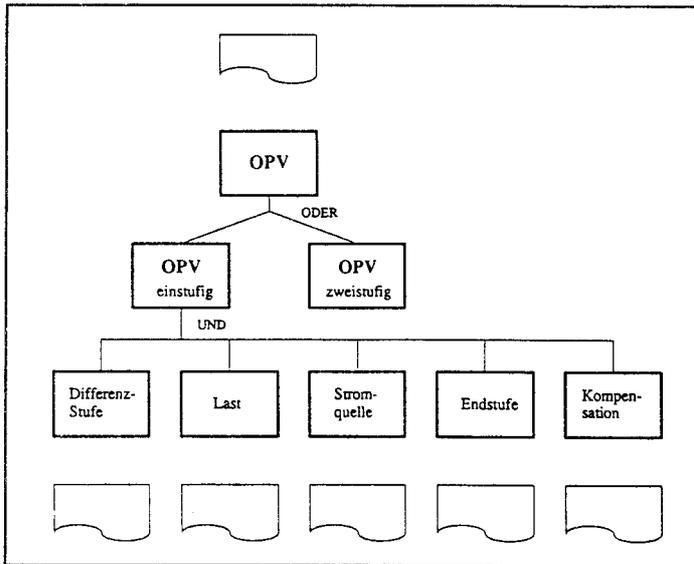


Bild 8: Syntheseplan Operationsverstärker

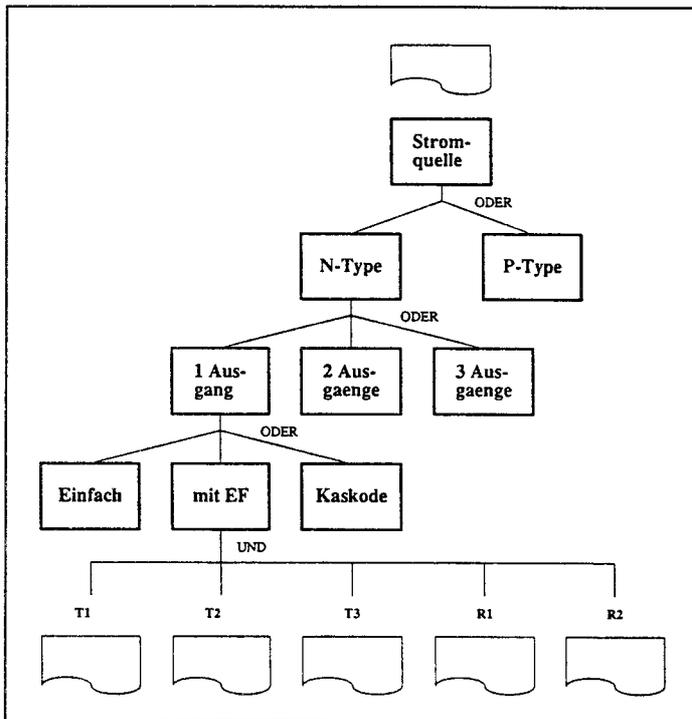


Bild 9: Syntheseplan Stromquelle

stordaten ergeben sich dann z. B. der Strombereich, Rauschdaten und Paarungsgenauigkeiten (Spannungsgrenzwerte und Frequenzgrenzwerte gehen bereits über die Technologiedaten in die Auswahlkriterien ein). Die Widerstandsdaten setzen sich aus den Widerstandswerten, der maximalen Stromstärke und ebenfalls Paarungsgenauigkeiten zusammen.

Die drei in Bild 9 dargestellten Entscheidungsebenen lassen sich natürlich formal auch in einer einzigen Ebene durchlaufen. Auf diese Weise läßt sich der TOP-DOWN-Syntheseweg dahingehend verallgemeinern, daß immer abwechselnd ODER-Knoten bzw. UND-Knoten durchlaufen werden. Während am ODER-Knoten jeweils die Topologie festgelegt wird, erfolgt am UND-Knoten die Dimensionierung. Es ist allerdings offensichtlich, daß beim Scheitern eines Dimensionierungsvorgangs die Möglichkeit bestehen muß, die vorher getroffene Auswahl aufgrund neuer Erkenntnisse zu revidieren.

Zusammenfassung

Die Synthese analoger Schaltungen mit Hilfe parametrisierbarer Makrozellen wird aufgrund der erforderlichen Vielfalt nur ein begrenztes Anwendungsspektrum abdecken können. Wesentlich aussichtsreicher erscheint die Synthese auf der Basis kleinerer, universellerer Einheiten, den Skeletzzellen. Darüberhinaus eignen sich Skeletzzellen wegen ihrer geringen Bauelementezahl hervorragend für die Layoutsynthese mit Hilfe von Modulgeneratoren. Ebenfalls kann in der Regel die topologische Anordnung der Skeletzzellen im Schaltplan direkt auf das Layout übertragen werden.

Ein großer Teil der spezifischen Analoganforderungen bezüglich Schaltung und Design läßt sich anhand der relativ einfachen Schaltungseinheiten gut beherrschen. Somit bestehen gute Voraussetzungen dafür, daß die bei der Layoutsynthese anfallenden parasitären Elemente so klein gehalten werden, daß der Bereich der linearen Optimierung nicht verlassen wird.

Auf diese Weise stellt das Konzept der Skeletzzellen einen vielversprechenden Ansatz dar, auch auf dem Gebiet der Analog-ICs den Entwicklungsschwerpunkt auf die Systemebene zu verlagern, das Fehlerrisiko zu verringern ("correct by design") und das Entwickler-Potential auf die besonders anspruchsvollen Schaltungsteile zu konzentrieren.

Literaturverzeichnis

- [1] R. Hagelauer: "ASIC-Entwurf: Varianten, Verfahren, Systeme", Mikroelektronik Bd. 5 (1991) Heft 1
- [2] M. Glesner, P. Pöchmüller: "Silicon Compilation - Überblick und zukünftige Entwicklungstrends", Mikroelektronik Bd. 4 (1990) Heft 3
- [3] H. Knotz, Günther Tränkle: "Der Einsatz von HF-Transistor-Arrays in der Kommunikationstechnik", ITG-Fachbericht 119, Stuttgart 1992
- [4] Standardzellenbibliothek ADASIC, AEG 1990
- [5] M. Degrauwe et. al.: "Towards an analog System Design Environment", IEEE Journal of Solid-State Circuits, Vol. SC-24, pp. 659-671, June 1989
- [6] Syntheseprogramm "Explorer AutoFilter" Mentor Graphics Corporation 1991
- [7] G. Forster, F. Beck, G. Koppi, I. Walz: "Monolithische Integration der Signalverarbeitung für Funkgeräte", ITG-Fachbericht 119, Stuttgart 1992
- [8] Esprit Project No. 802 "CAD for VLSI Systems" (CVS Project) 1989

Fortschritte in der Entwicklung von Anwendungsspezifischen Integrierten Schaltungen (ASIC)

**Bericht von der CICC 93 - St. Diego/Kalifornien, USA
vom 9. - 12. Mai 1993**

**Prof. Dr.-Ing. Dirk Jansen
Fachhochschule Offenburg**

**Vortrag anlässlich des Workshops der
Multiprojekt-Chip Gruppe der Fachhochschulen BW
- Ravensburg/Weingarten -**

Übersicht:

- 1. Die CICC 93**
- 2. Technologische Trends**
- 3. Gate Arrays**
- 4. Programmierbare ASICs**
- 5. Herausragende Anwendungen**
- 6. Zusammenfassung**

1. CICC 93 : Custom Integrated Circuit Conference, St. Diego / USA

- 9. - 12. Mai 1993 in St. Diego / USA
- ca. 600 Teilnehmer, überwiegend USA/Fernost, geringe europ. Beteiligung
- 195 Vorträge in 4 Sälen parallel
- 4 Evening Sessions mit Podiums-Diskussionen
- 12 Tutorials bzw. Educational Sessions am Tag zuvor (Sonntag !)
- Spezialmesse von ASIC-Herstellern und CAE-Firmen

Thema:

- breites Forum sowohl für Anbieter als auch für Anwender von ASIC's
 - ↳ Technologien (CMOS, BiCMOS, Bipolar, Low Voltage CMOS ...)
 - ↳ Tools (CAE-Methoden, Algorithmen, Modellierung)
 - ↳ fertigungsgerechter Entwurf (Test, Zuverlässigkeit)
 - ↳ neue Bausteine (Gate Arrays, FPGA's)
 - ↳ Applikationen (analog, digital)

Teilnehmer der MPC-Gruppe:

Prof. Dr. Albert, Prof. Ritzert, Prof. Dr. Khakzar, Prof. Dr. Jansen, Prof. Dr. Toepfer, Prof. Nielinger

kombiniert CMOS mit Bipolar-Eigenschaften

0.8 μm - Standard BiCMOS - Prozeß

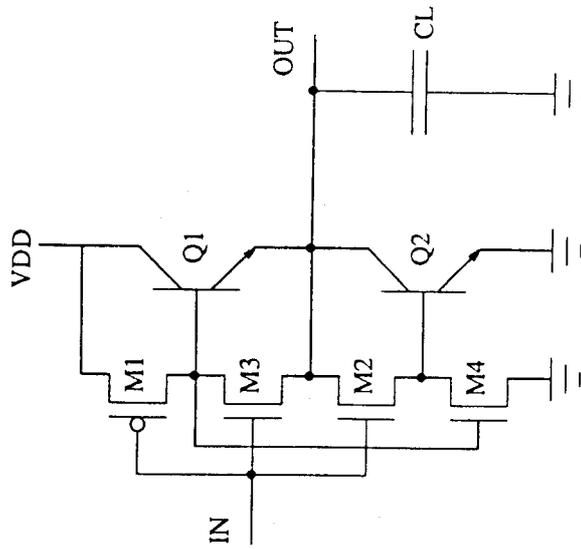
NPN-Transistor mit $f_T = 7 \text{ GHz}$

Eingangsseitig 1 CMOS-Load

Ausgangsseitig hohe Treiberleistung, da niederohmig

wirksam in Applikationen, die hohes Fan-Out erfordern

Geschwindigkeitserhöhung von Decodern und Sense-Amplifiern in SRAM

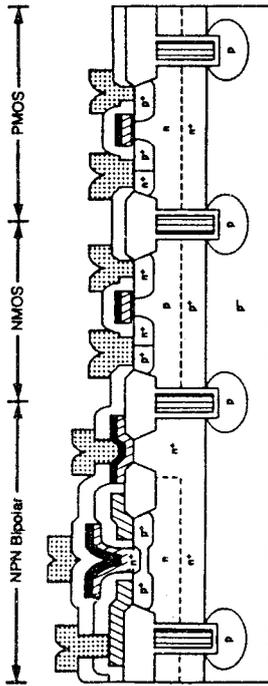


BiCMOS - Driver

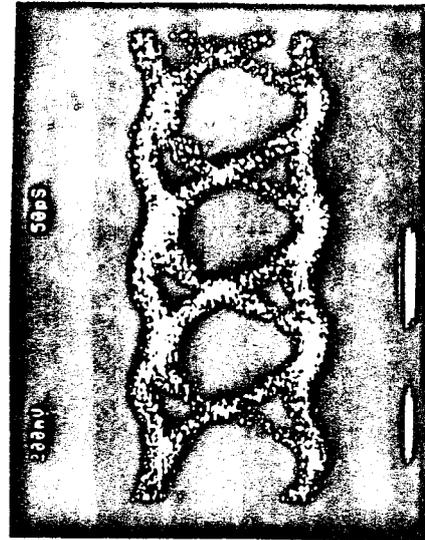
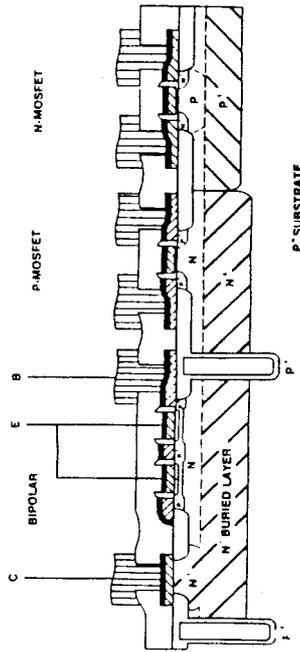
SRAM - Read Access Time (2k x 21, 5V)

Module	CMOS	BiCMOS
Address Buffer	1.0	0.8
Word Decode	2.4	1.2
Cell	1.0	1.0
Sense Ampl. + Output Buffer	3.5	1.5
Summe	7.9 ns	4.5 ns

Quelle: AT&T, T.E. Ham et al.



SiO₂
 Undoped Poly-Si
 P+ Poly-Si
 N+ Poly-Si



8 Gbit/sec - Entscheider

BiCMOS kombiniert Eigenschaften von CMOS und Bipolar-Technik

höchste Geschwindigkeiten durch ECL-artige Schaltungstechnik erreichbar bei gleichzeitig niedrigem Leistungsverbrauch

Schlüsselement: schneller Bipolar-Transistor in CMOS-verträglicher Technologie

Super self aligned BiCMOS-Prozess, minimisiert die Basis-Kollektor-Kapazität C_{BC} und den Basis-Widerstand R_B

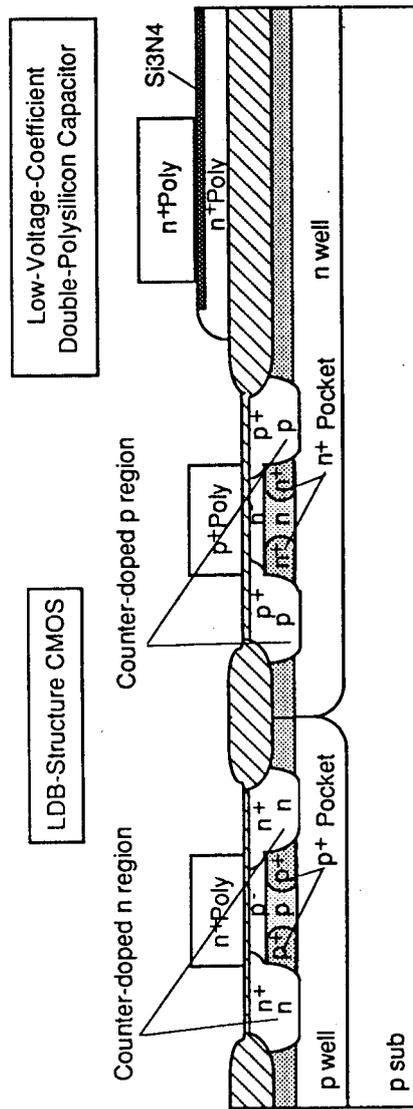
Eigenschaften:

- ↘ 0.5 μm - Lithographie
- ↘ Transitfrequenz $f_T = 34 \text{ GHz}$
- ↘ CMOS - Gate - Delay $t_{pd} = 50 \text{ ps}$

Quelle: AT&T, T.M. Liu et al.

2.3 Technologische Trends: 0.3 μm CMOS - Technologie (Hitachi)

- 0.3 μm - Kanallänge
- MOSFET mit laterally doped buried layer (LDB)
- für niedrige Betriebsspannung
- geringe Variationen der Schwellspannung V_{th}
- hohe Stromtreiber-Fähigkeit
- 1.5 V ... 3 V Betriebsspannung
- für digital & analog geeignet



Querschnitt durch 0.3 μm CMOS-Technologie für gemischte analog/digitale Schaltungen

Daten:

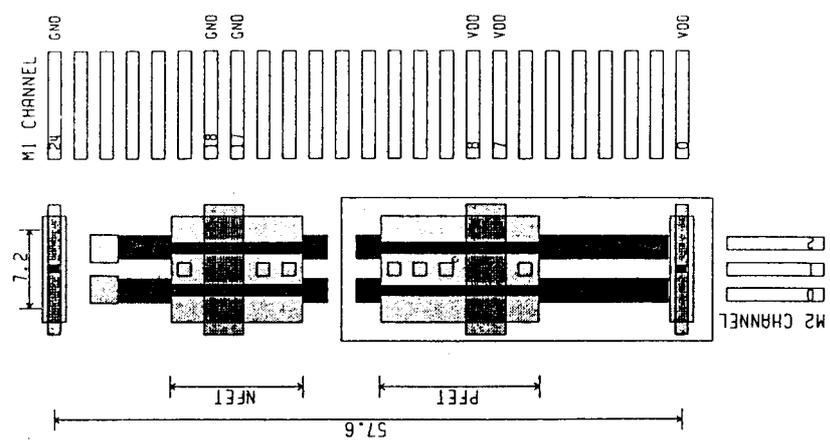
L_{eff}	= 0.3 μm
t_{ox}	= 6 nm
V_{tn}/V_{tp}	= 0.5 / -0.5 V
$I_{d satn}$	= 0.17 mA/ μm @ 1.5 V
$I_{d satp}$	= 0.08 mA/ μm @ -1.5 V
V_{DSSmax}	= 5.8 V
Insulator	= Si_3N_4 38 nm

Quelle: ULSI Research Center, Hitachi
Masafumi Miyamoto et al.

3.1 Gate Arrays:

CMOS Gate Arrays 250 k

(IBM - Nippon)



- 3.3 V Betriebsspannung (2.7 ... 3.8 V) CMOS - Sea of Gates - Array
- 0.45 μm effektive Kanallänge, $t_{\text{ox}} = 11.5 \text{ nm}$
- 2.4 μm Metal Pitch - Verdrahtungsraster
- 3 Metallisierungsebenen
- 0.8 μm Lithographie für Low Cost/Low Power Circuits
- 260 000 Gatter (NAND-Äquivalente)
- I/O-Zellen mit ESD-Protection (3 kV) und Slew Rate Control
- Umfangreiche Standard-Bibliothek incl. Macro-Zellen verfügbar

Quelle: Makoto Ueda et al.

IBM Yasu Technology Application Laboratory

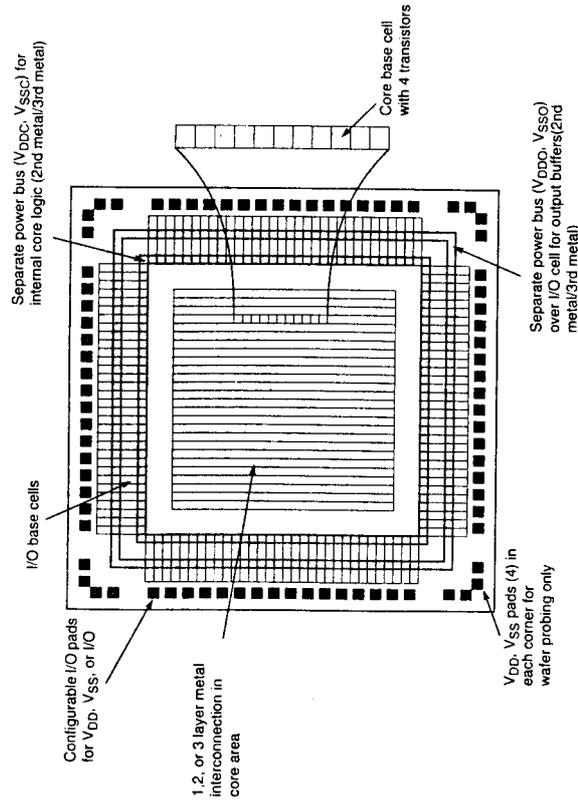
Flüß, Prof. Dr.-Ing. D. Jansen, 1993

Gate Array Basic Cell Structure

Daten:

- 2 - NAND
- $t_{\text{pd}} = 245 \text{ ps @ } 3.3 \text{ V}$
- Leistungsaufnahme:
- 1.3 $\mu\text{W}/\text{MHz}/\text{Gate}$

- 500 k nutzbare Gatter (10 k ... 774 k raw gates)
- 0.5 μm - 3.3 V - CMOS - Prozeß
- Sea of Gates Struktur
- 3 Lagen-Verdrahtung (Metall)
- 110 psec Gatterlaufzeit (2-NAND)
- spezielle Taktverteilungszellen mit geringem skew
- RAM bis 144 kBit auf dem Chip möglich
- Umfangreiche Bibliothek einschließlich Makrozellen wie PCI, UARTs, SCSI
- JTAG Boundary Scan und Scan Path ATVG
- für alle modernen Entwicklungsplattformen, u.a. Mentor



MSM10R0000 Array-Architektur

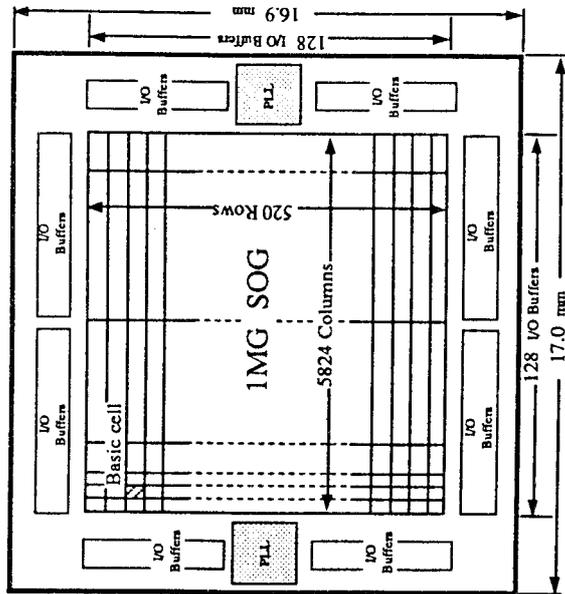
Quelle: OKI Data Sheet, MSM10R0000 Family

3.3 Gate Arrays:

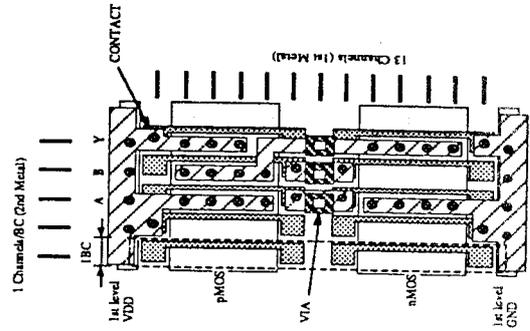
CMOS Gate Array (SOG)

(Mitsubishi)

- 1 Mio. Gates, 3 Mio. Basic Cells
- Cell-Größe $2.5 \times 28 \mu\text{m}^2$
- 0.5 μm CMOS 1 Poly - 3 Metall-Lagen - Prozeß
- Chip-Größe $17 \times 16.9 \text{ mm}^2$
- 512 I/O-Cells, 1 PLL-Cell (analog)
- 85 ps (F.O. = 1) Delay, > 100 MHz Operating Frequency
- RAM-Cells 64×64 mit 4.6 nsec t_{acc}
- 0.9 $\mu\text{W}/\text{MHz}/\text{Gate}$ Power Dissipation
- 3.3 V Betriebsspannung



Chip-Floorplan eines 1 MG CMOS SOG

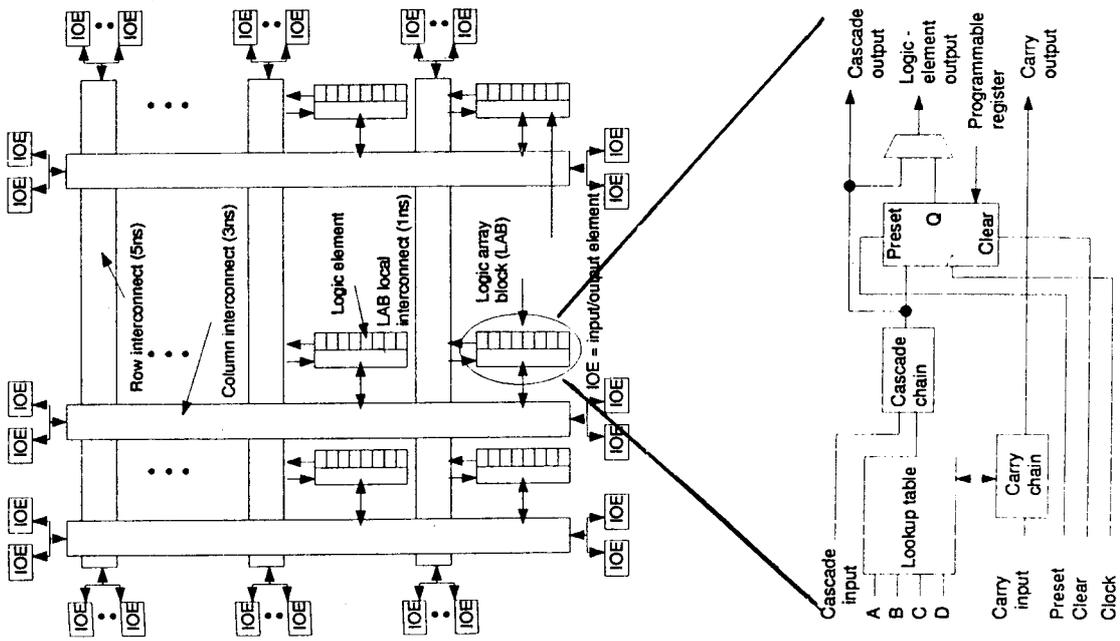


Layout eines 2-NAND

Quelle: Nobuyuki Ikeda et al.

ASIC Design Engineering Center, Mitsubishi El. Corp.

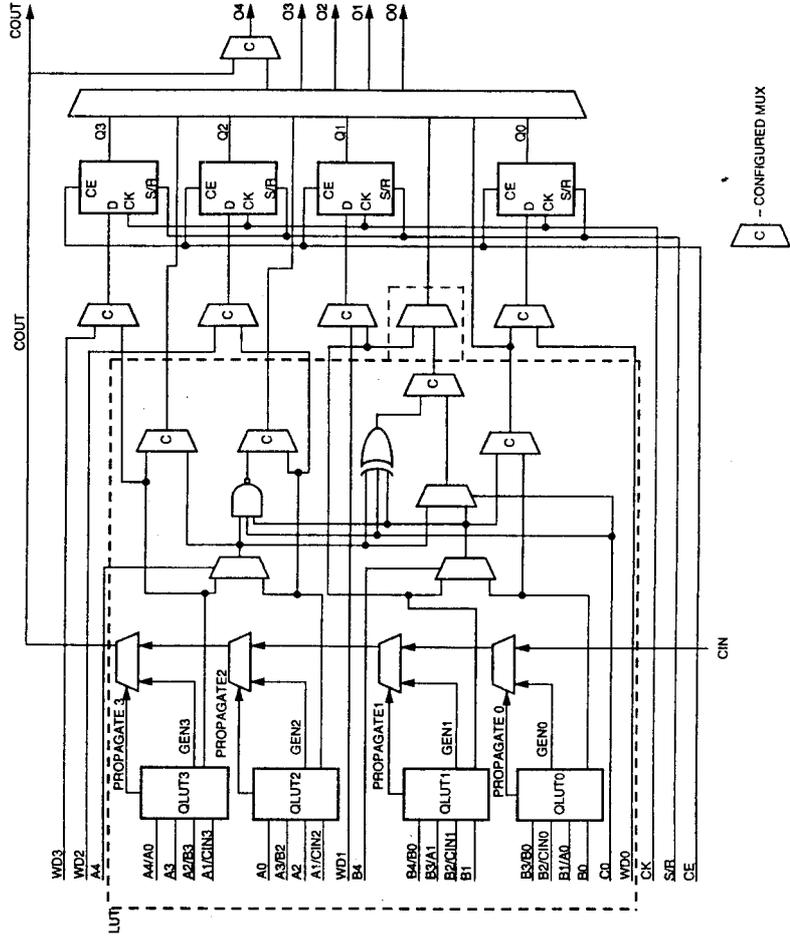
FLEX = Flexible Logic Element Matrix (8000er - Serie)



- "Dual Granularity" Architektur
- 1 Bit Logical Element (LE) zu je 8 in einem LAB (Logical Array Block) zusammengefaßt
- lokales Routing im LAB
- LABs über globales Routing miteinander verbunden
- Baustein EPL81188 enthält 1188 Register und ca. 12000 Gatter, organisiert als 6 rows x 21 columns
- 180 I/O - Elemente, 232 Pin PGA / 240 Pin QFP
- Delays: lokal 1 nsec, horizontal 6 nsec, vertikal 9 nsec
- hohe Systemtaktrate, hohe Dichte, gute Entwurfs-systematik
- Software für PC

Quelle: Richard Cliff et al.
Altera Corporation.

ORCA = Optimized Reconfigurable Cell Array



- optimiert für Datenpfad / RAM-ROM-Strukturen
- FPU ist 4 Bit breit (Nibble-Breite)
- konfigurierbar als:
 - 4 Bit Adder/Subtractor
 - 4 Bit Counter
 - 4 Bit Comparator etc. mit fast carry
- oder für Kontrollfunktionen:
 - 4 x 4 Bit - Funktion
 - 2 x 5 Bit - "
 - 1 x 6 Bit - "

- RAM/ROM mit 16 x 4 Bit
- Multi-Level-Routing
- RAM-based Architektur in 0.6 μm Technologie
- Download bei Anlegen der Betriebsspannung (ähnlich XILINX)
- von 3500 ... 22 000 Gattern,
- max 2304 Register / 36 864 RAM-Bits
- Array-Größe von 10 x 10 ... 24 x 24
- 33 MHz ... 80 MHz Taktfrequenz
- Software für PC und SUN Workstations

- **Multimedia, insbesondere Video-Compression und Codec-Chips**
 - ↳ **Motion Video Compression LSI (OKI)**
 - ↳ **Motion Estimation Processor LSI (Mitsubishi)**
 - ↳ **Multistandard Video Codec (C-Cube)**
- **Microprozessoren und Signalprozessoren**
 - ↳ **3.3 V, 2.8 Mio. Transistors - BiCMOS - RISC - Processor (Hitachi)**
($f_T = 15 \text{ GHz}$, $L_{\text{eff}} = 0.6 \mu\text{m}$, $T_{\text{ox}} = 9 \text{ nm}$, 4-Lagen Metall-Verdrahtung, 120 MHz)
 - ↳ **2.4 nsec - 16 Bit - Arithmetic Logic Unit (NEC)**
($0.5 \mu\text{m}$ CMOS, 3-Lagen Verdrahtung, 3.3 V, 200 mW @ 300 MHz)
 - ↳ **Font Rendering Processor mit RISC CPU (Fuji XEROX Co.)**
($0.8 \mu\text{m}$ CMOS, 20 MHz, 127 k Transistoren, für Drucker)
 - ↳ **Signal Processor Core LSI (NEC)**
($0.5 \mu\text{m}$ BiCMOS, 300 MHz, 16 Bit, DCT, Video Compression)

5.2 Herausragende Anwendungen (high performance circuits)

- **Communication ASIC's**
 - **Direct Sequence Spread Spectrum Transceiver Chip (Cylink)**
(Correlator, Digital PLL, Detection Logic)
 - **300 MHz BiCMOS Serial Data Transceiver (MIT)**
(1.2 μm BiCMOS, 2 d PLL's, 30 MHz parallel Interface, 1 W, Jitter 63/50 ps)

- **Embedded Processors (Motorola) mit Gate Array**
 - **68 000 kombiniert mit CMOS Gate Array und Interface Strukturen**

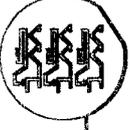
- **1.16 GHz Dual-Modulus 1.2 μm CMOS Prescaler (Swiss Fed. Inst. of Techn., ETH Zürich)**
 - **Standard CMOS Prozeß verwendet, leicht integrierbar**

- CICC 93 gibt Einsicht in neuesten Stand der ASIC-Entwicklung
- Technischer Stand:
 - ↳ Gate Arrays: > 1 Mio. Gates, < 100 psec Delay
 - ↳ Technologie < 0.5 μm CMOS
 - ↳ BiCMOS zukünftige Technologie
 - ↳ Taktfrequenzen bis 300 MHz
 - ↳ FPGA's für Anwendungen < 10 000 Gates
- Es gibt fast nichts, was nicht in Gate Arrays (digital) paßt !
- Technologietreiber: Multimedia, Kommunikationstechnik, Prozessoren
- Europäische Industrie praktisch nicht vertreten (und nicht konkurrenzfähig ?!)

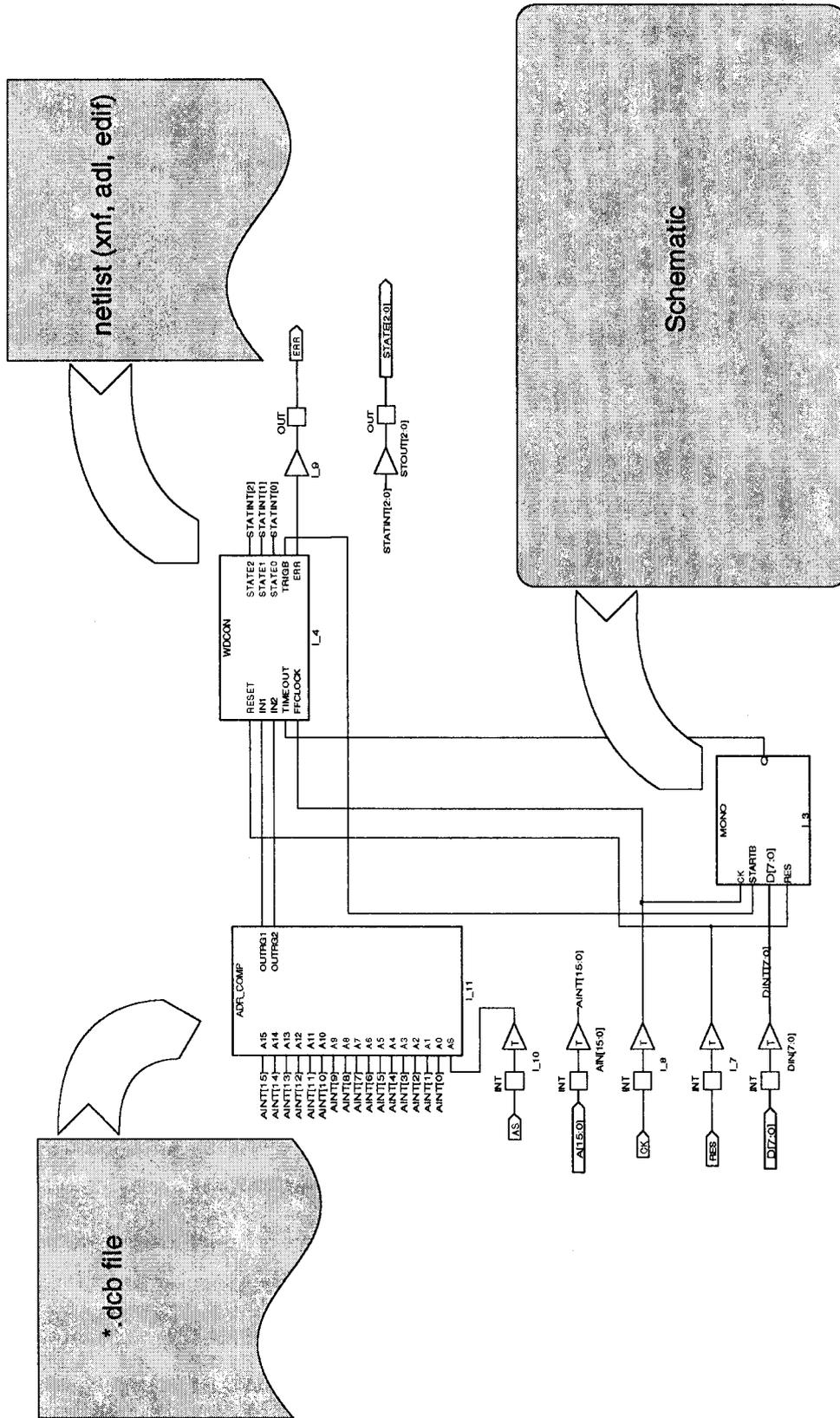


*Technologie-transparentes ASIC Design
auf einem Low-Cost Entwicklungssystem
MIGRATE am Institut für Mikroelektronik Stuttgart*

J. Gandowitz, IMS Stuttgart



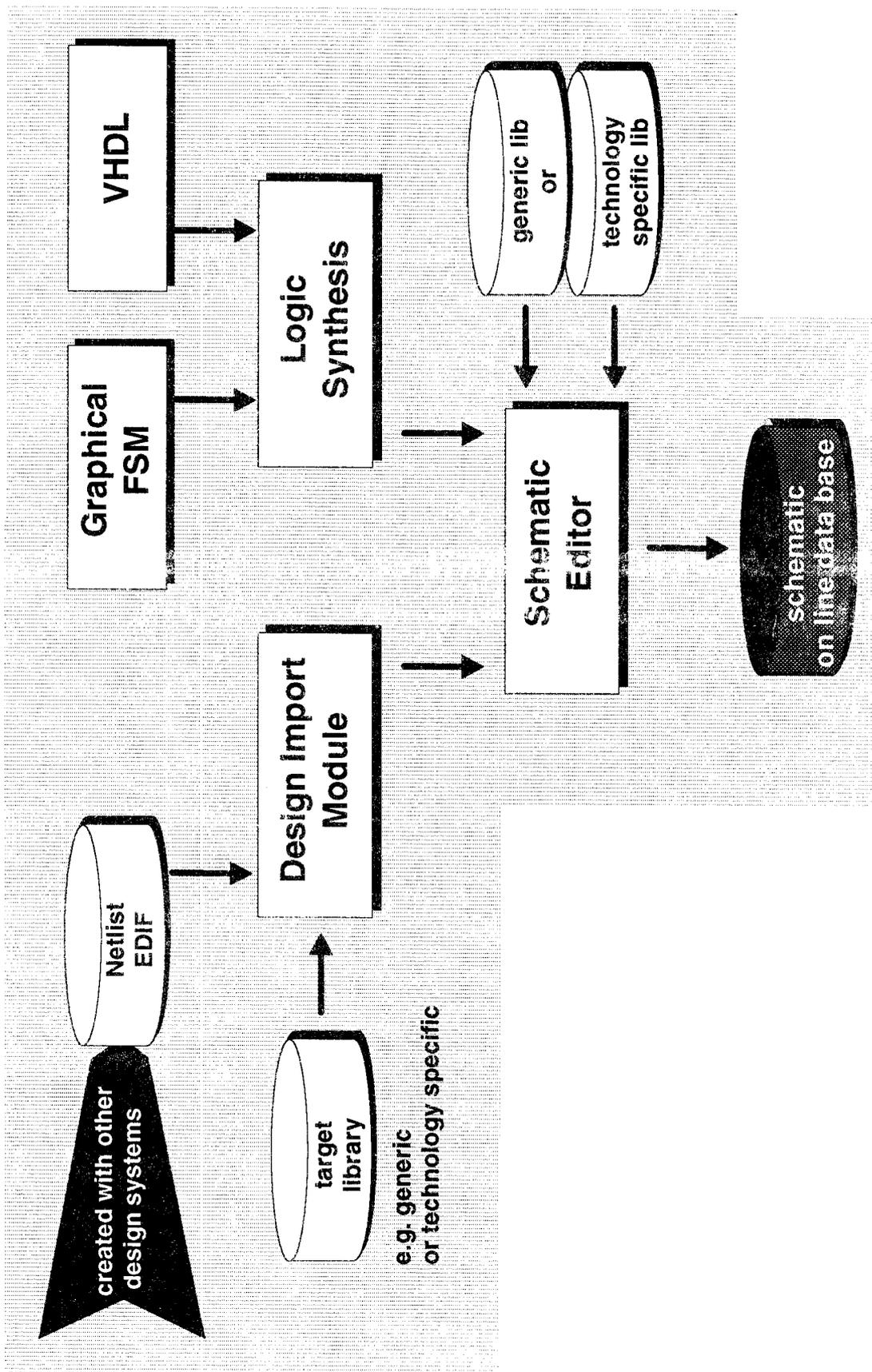
- **universell**
PLD - FPGA - GA - Standard Cell IC
- **Standard Interfaces**
EDIF 200 Schematic in/out, EDIF 200 Netlist In/Out
VHDL usw.
- **Platforms**
Work Stations, PCs (IBM compatible) und Sun
- **Standard Windowing Environment**
Windows-Dos, Motif/OpenWindows- Sun OS
- **Ease of Use**
Bedienung durch Dialog-Boxen
DDE, Interaktion zu On-Line Data Base
- **Offen:**
Prozedurale C-Schnittstelle

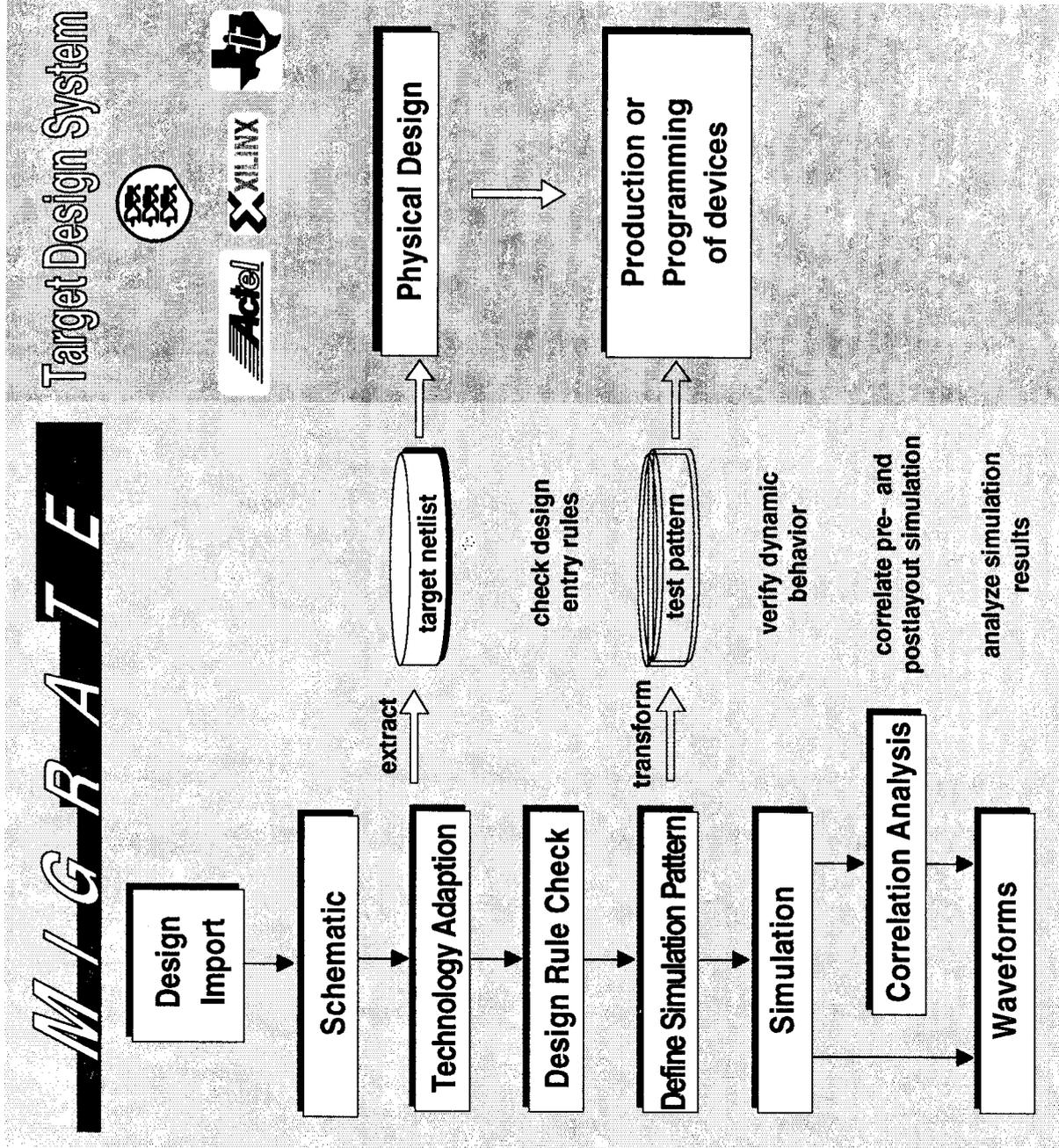
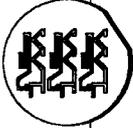


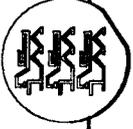
Design Capture



MIGRATION TECHNOLOGY





**■ MPROJECT**

*Technology selection
Package selection
Operating condition*

■ M2TECH

*Technology adaption
Cell mapping
Technology data
Operating point
Target netlist*



M2TECH

Mode
 Unit
 Min
 Typ
 Max

Wire delay calculation
 Prelayout Postlayout

Operating Point
 Voltage: 5.00 Volt
 Temperature: 25.0 °C

Project data
 Vendor: ims Derating: 1.000
 Technology: gtf012 Process: IND
 Master: gf9g1 Package: CLCC68-1

Write Target Netlist

Run Cancel Project Help

MPROJECT

Project Name: DEMO

ASIC Vendor: IMS

Technology: GTF012

Master: GF9G1

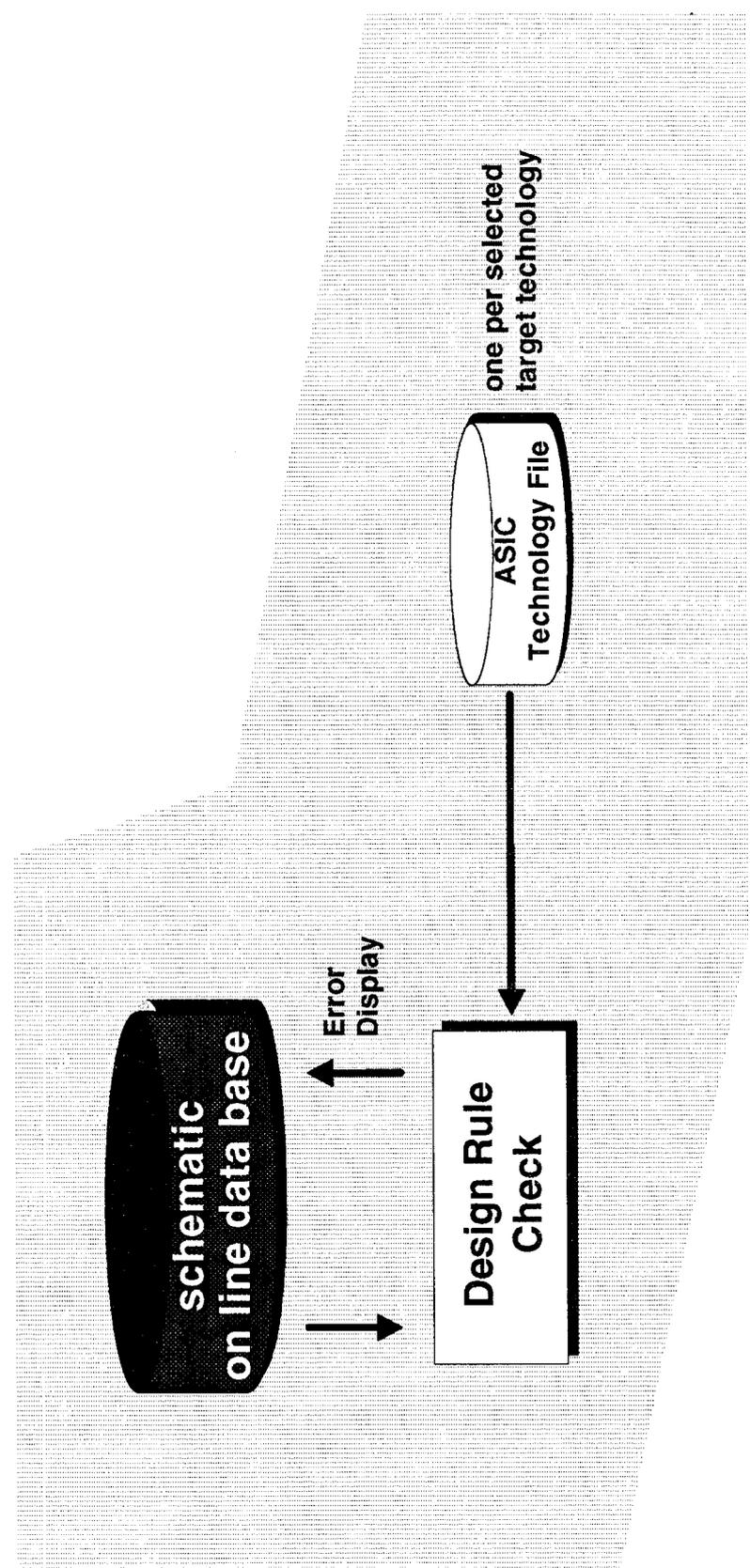
Package: CLCC68-1

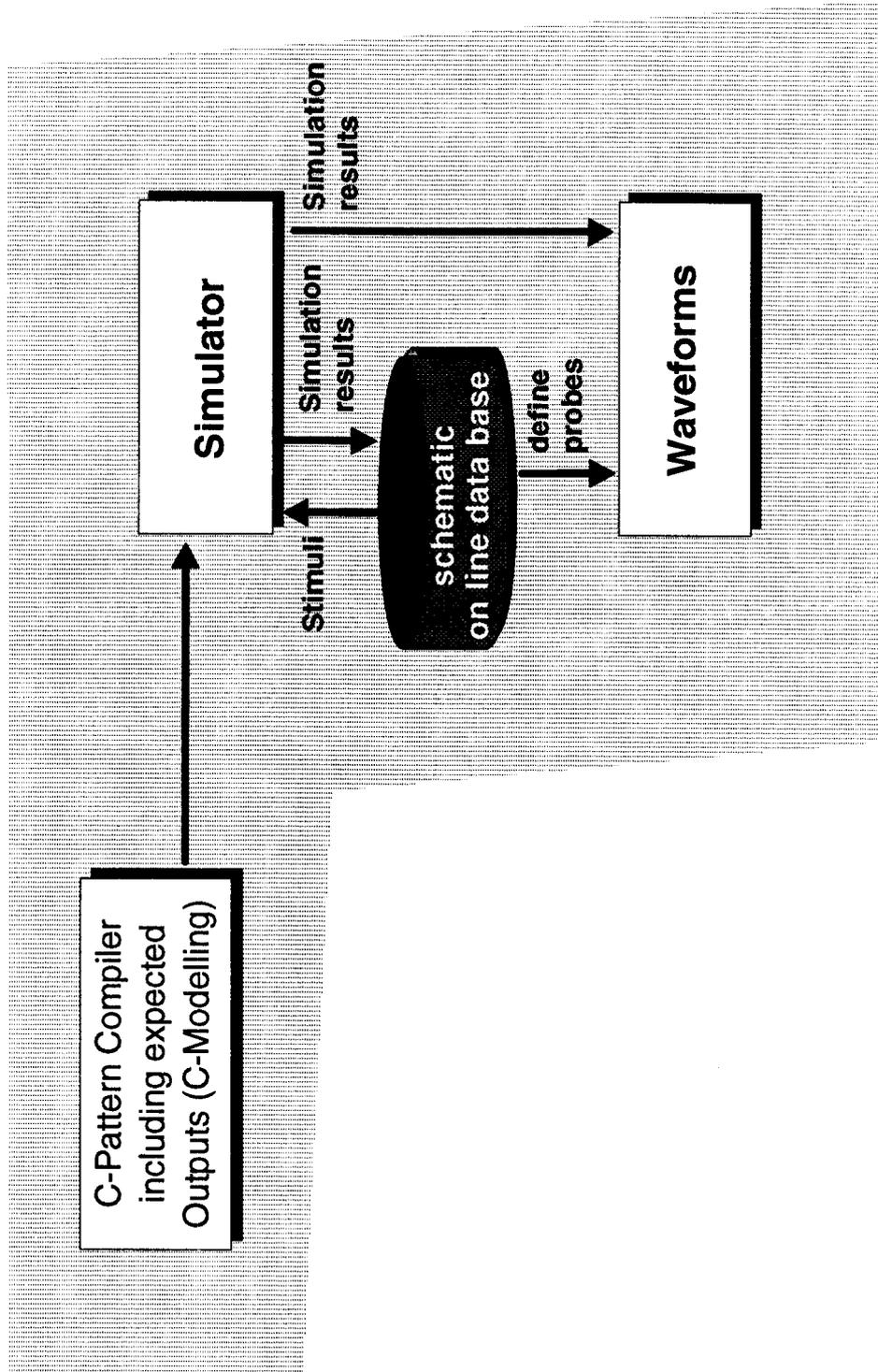
Process
 COM MIL. IND

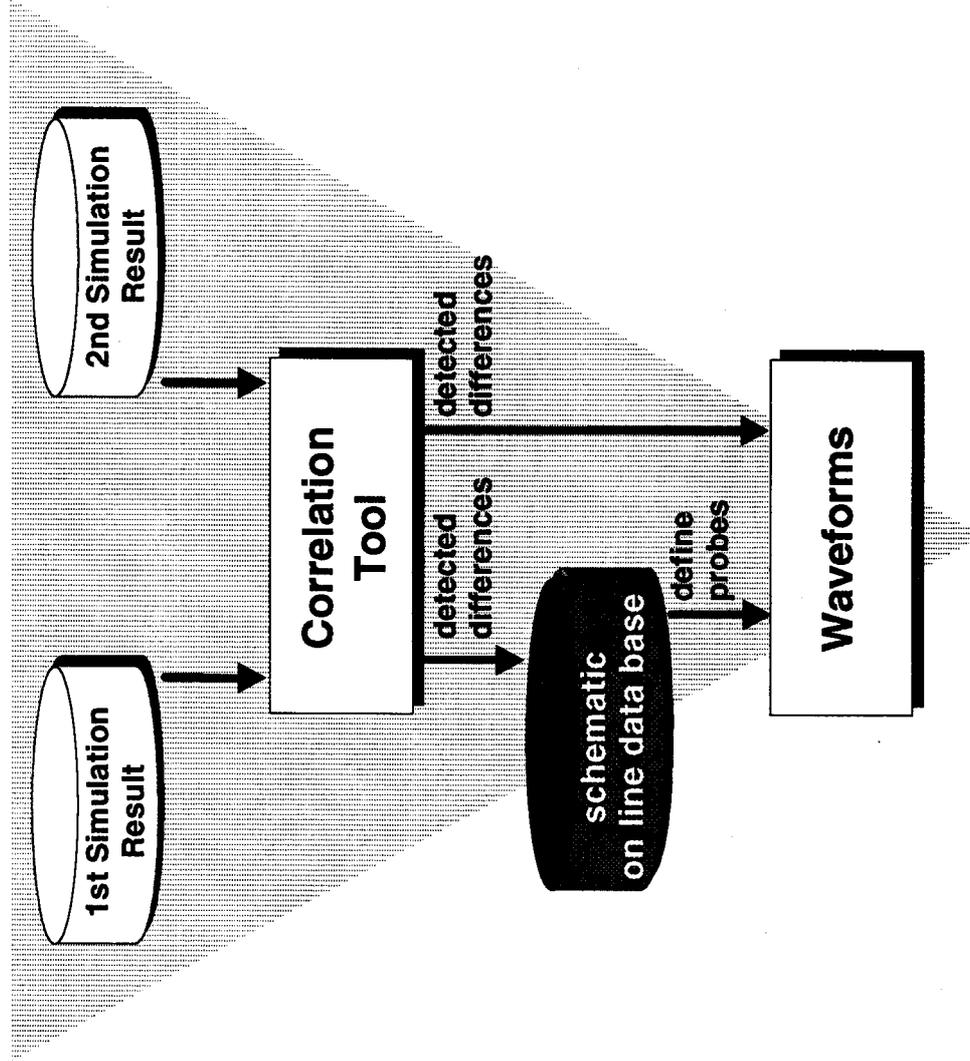
012 Sheet 1 of 1 Help F1

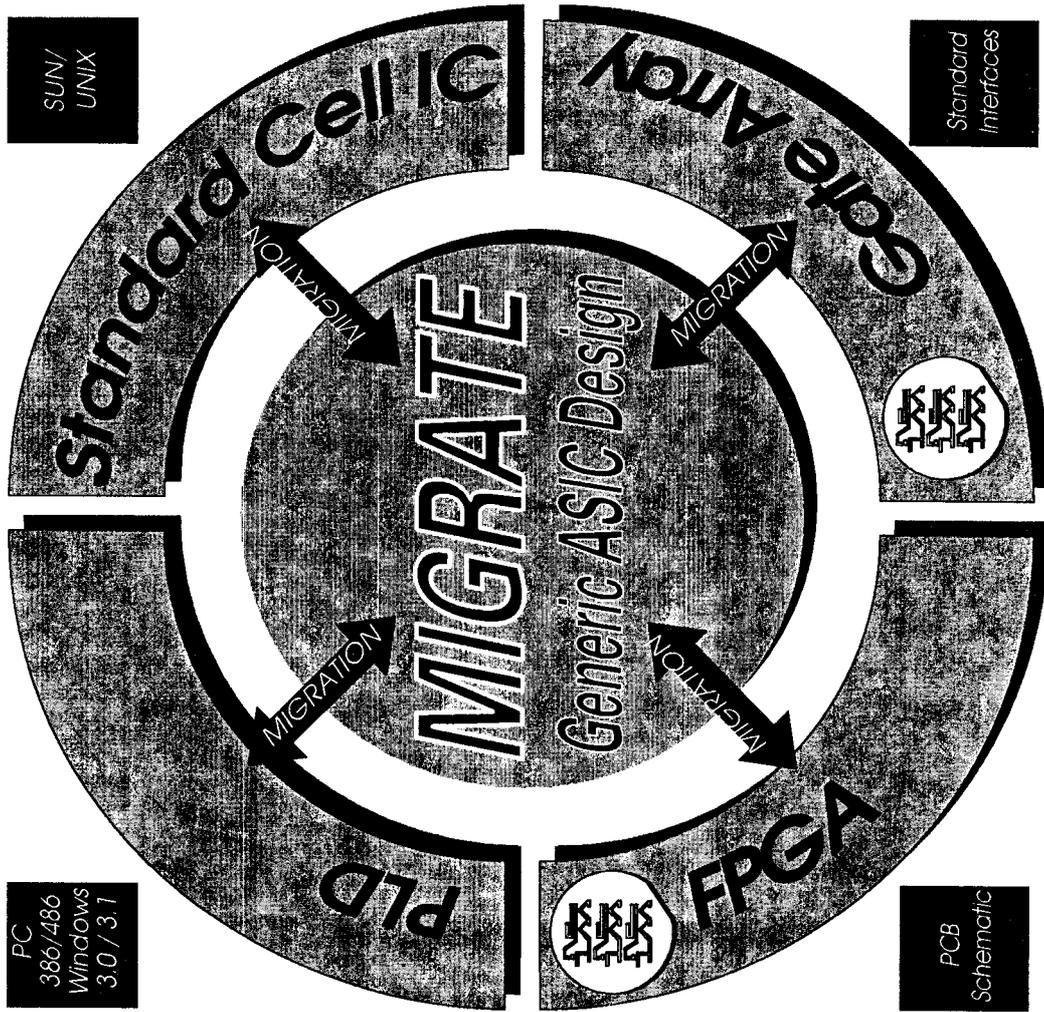
Process XSIM

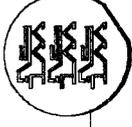
Zoom In - Pick Center Point or Corner of Box











Aktuelles Beispiel am IMS

- **ASIC integriert mehrere Funktionen**
Prozessorprotokolle, Zähler, Speicheransteuerung
- **Frühzeitige Verifikation von Teilfunktionen mit FPGAs**
ACT1, A1020, PLCC84
- **Schematic Entry und Simulation**
Auf 486 PC am IMS
- **Problemlose Übernahme von Netzliste und Pining in die Compass Tools**
Layout, Postlayout, Simulation
- **ASIC-Prototypenfertigung**
GF4-Master, 6.4K Gatteräquivalente, 82 % Belegung, CLCC84, Pin kompatibel zu FPGA

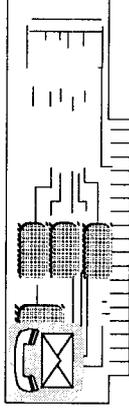
Kunde:

Dr. Neuhaus KG



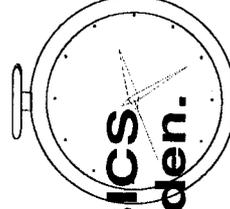
Applikation:

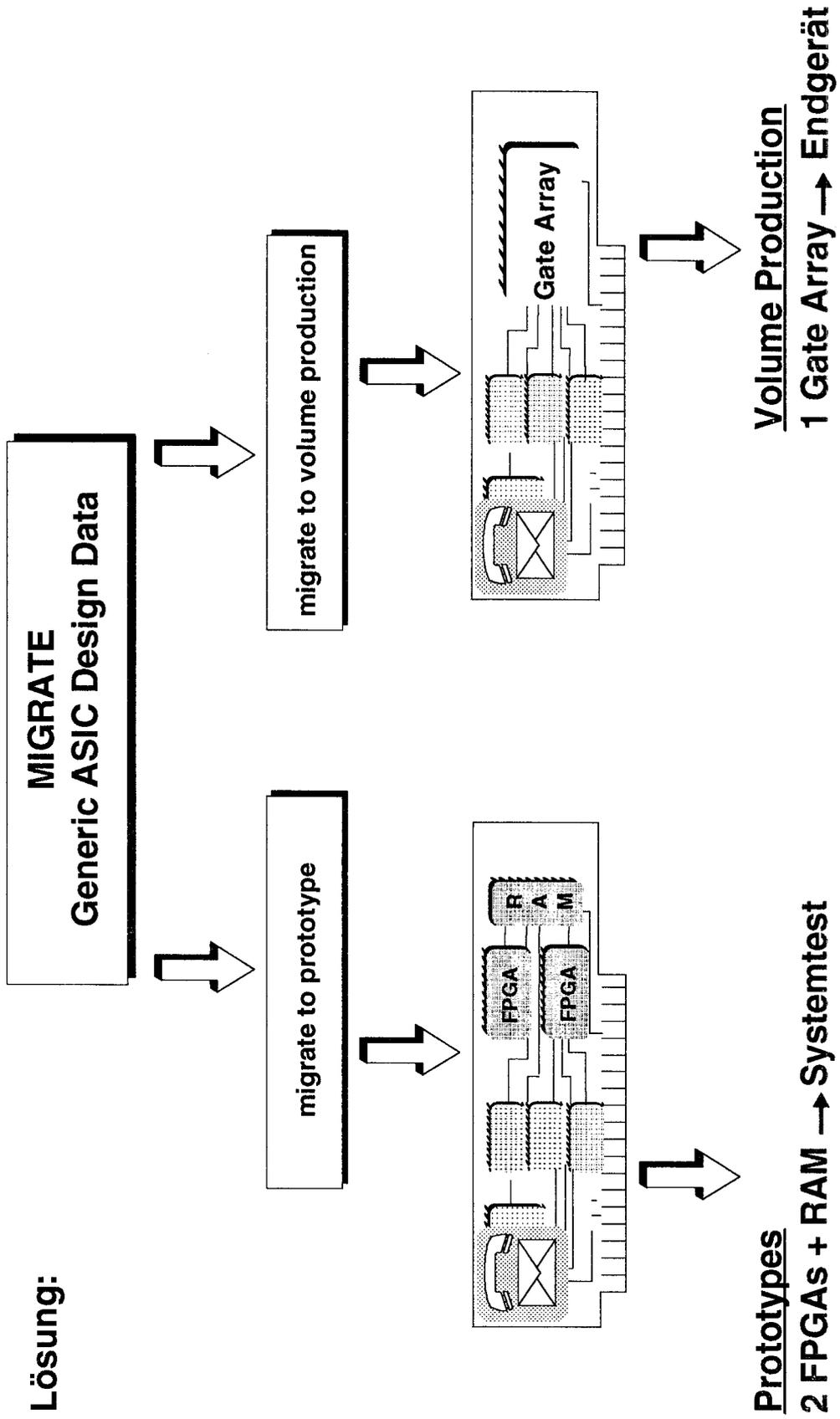
Modem/Fax

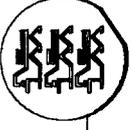


Randbedingungen:

Um den Zieltermin einzuhalten, mußte ein Redesign des ASICs abgeschlossen werden.







Das IMS bietet

- **Beratung**
ASIC-Realisierung, Partitionierung, Prototypen
- **Schulung**
ASIC Design mit VHDL, Design for Testability, ASIC-Praxis mit Übungen
- **Schematic Entry, Logic Synthesis und Simulation**
Auf 486 PC und Workstations am IMS
- **Layout, Postlayout**
- **FPGA - Programmierung**
Actel ACT1 und ACT2 / Xilinx XC2000, XC3000 und XC4000
- **ASIC-Prototypenfertigung**

Realisierung großer Netze

M. Faas, P. Leber, W. Rülling
FH Furtwangen, Mikrosystemtechnik, IIT

Zusammenfassung

Semi-Custom Entwurfssysteme haben Schwierigkeiten mit der Realisierung großer zeitkritischer Netze. In der vorliegenden Arbeit wird daher ein neues Optimierungstool vorgestellt, mit dessen Hilfe sich große Netze layoutabhängig in einfacher realisierbare Teilnetze zerlegen lassen. Damit werden auch komplexe zeitkritische Schaltungsentwürfe ohne manuelle Eingriffe umsetzbar.

1 Einführung

Beim Entwurf komplexer integrierter VLSI-Schaltungen stellt sich oft das Problem, daß viele Gatter mit dem gleichen elektrischen Signal versorgt werden müssen [1]. In der vom Logikdesigner entworfenen Netzliste sieht ein solches Netz zum Beispiel wie in Abbildung 1 dargestellt aus.

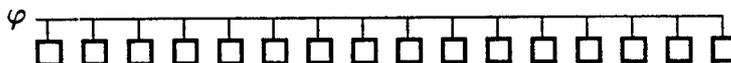


Abb.1 Netzdarstellung in Semi-Custom Netzliste

Werden an diese Netze zusätzlich hohe Geschwindigkeitsanforderungen gestellt, müssen sie mit Hilfe von Baumstrukturen mit zusätzlichen Treibern realisiert werden, damit die Signale mit ausreichender Geschwindigkeit und möglichst zeitgleich bei allen zu treibenden Gattern eintreffen. Eine solche Realisierung könnte folgendermaßen aussehen.

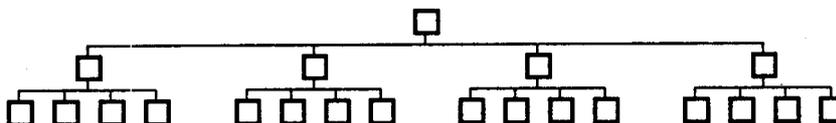


Abb.2 Netzrealisierung als Baumstruktur mit Treibern

Bei einer gleichmäßigen Verteilung der Gatter auf einem Chip kann die Baumstruktur beispielsweise über einen balancierten H-Baum (Abb.3) günstig realisiert werden. Bei dieser Anordnung entstehen minimale Laufzeitunterschiede und jeder Treiber hat eine gleichgroße Fläche zu versorgen.

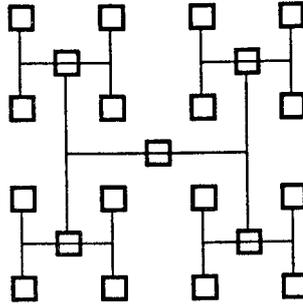


Abb.3 H-Baum Realisierung im Layout

Im allgemeinen wird die Baumstruktur jedoch unregelmäßiger aussehen, da die Treiberanordnung und Dimensionierung jeweils den Layoutpositionen und dem jeweiligen Fanin der zu treibenden Gatter angepaßt werden muß.

Da bei zeitkritischen Anwendungen insbesondere die Leitungskapazitäten minimiert werden müssen, ist es entscheidend, daß jeweils im Layout benachbarte Gatter von einem gemeinsamen lokalen Treiber versorgt werden. Damit ist die zu realisierende Netzliste der Schaltung offensichtlich vom Layout der Schaltung abhängig.

2 Problemstellung

Beim Chipentwurf mit Hilfe von Semi-Custom Entwurfssystemen geht man typischerweise so vor, daß der Schaltungsentwurf in eine Gatternetzliste konvertiert wird und man sich daraus automatisch durch Platzierungs- und Verdrahtungsalgorithmen ein Layout generieren läßt. Dabei wird der Designer keinen direkten Einfluß auf die Layoutgeometrie haben und insbesondere auch keinen Überblick über das entstehende Layout bekommen. Daraus ergibt sich als direkte Konsequenz, daß der Designer bei der Erstellung der Netzliste keine Layoutabhängigkeiten berücksichtigen kann.

In den üblichen Semi-Custom Entwurfstools sind zwar Sonderbehandlungen für große Netze wie Taktsignale und Stromversorgungen realisiert, es hat sich jedoch gezeigt, daß diese Sonderfälle das allgemeine Problem, z.B. die Realisierung dynamischer Logik, schaltungsintern erzeugter Steuersignale komplexer Rechenwerke oder auch Mehrphasenclocks (z.B. 16-phasig) nicht ausreichend beschreiben. Als Beispiel ist in Abbildung 4 die Skizze eines seriellen Multiplizierers dargestellt. Ein dabei oft auftretendes Steuersignal ist das *reset*-Signal, welches an jede Volladdierer-Grundzelle geführt werden muß. Um bei dieser Anordnung der Grundzellen eine gleichmäßige Aufteilung der Leitungsführung zu erhalten, müssen die Baumstrukturen zur Verstärkung des Signals layoutabhängig realisiert werden.

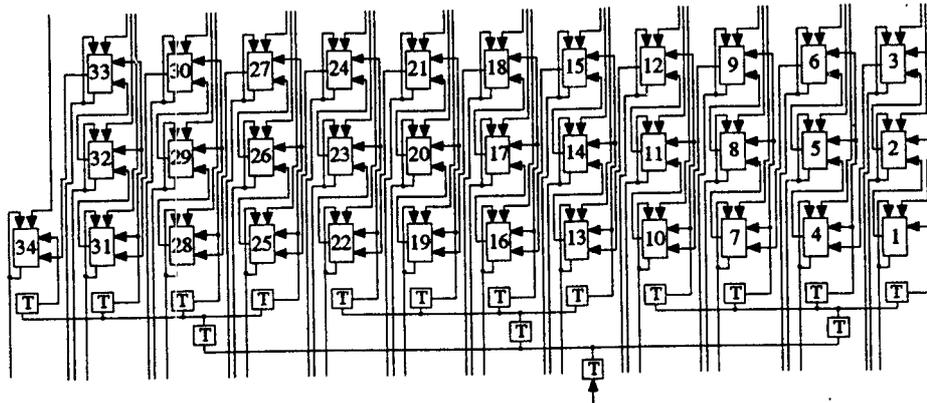


Abb. 4 Baumstruktur zur Verstärkung des *reset*-Signals eines seriellen Multiplizierers

Ein Beispiel eines automatisch generierten Layouts in welchem große Netze realisiert wurden, kann in der folgenden Verdrahtungsskizze in Abbildung 5 betrachtet werden.

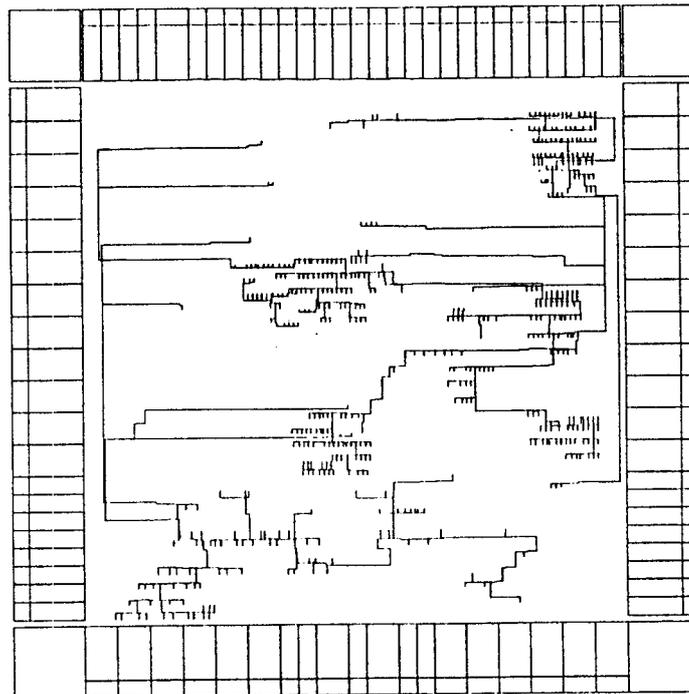


Abb. 5 Automatisch generiertes Layout

Hier wurden die von einem großen Netz zu versorgenden Gatter der Schaltung jeweils, entsprechend ihrer Reihenfolge in der Netzliste, verschiedenen Treibern

zugeordnet. Anschließend wurde die Netzliste automatisch vom CADENCE-Entwurfssystem [2] plaziert und verdrahtet. Wie die Abbildung zeigt, haben sich bei dieser Vorgehensweise recht lange Signalleitungen ergeben. Die Netze sind offensichtlich nicht lokal realisiert.

Zum Vergleich betrachte man die Realisierung aus Abbildung 6, bei der der Verdrahtungsaufwand erheblich geringer ist, da es gelungen ist, jeweils naheliegende Treiber zur Versorgung der Gatter zu benutzen.

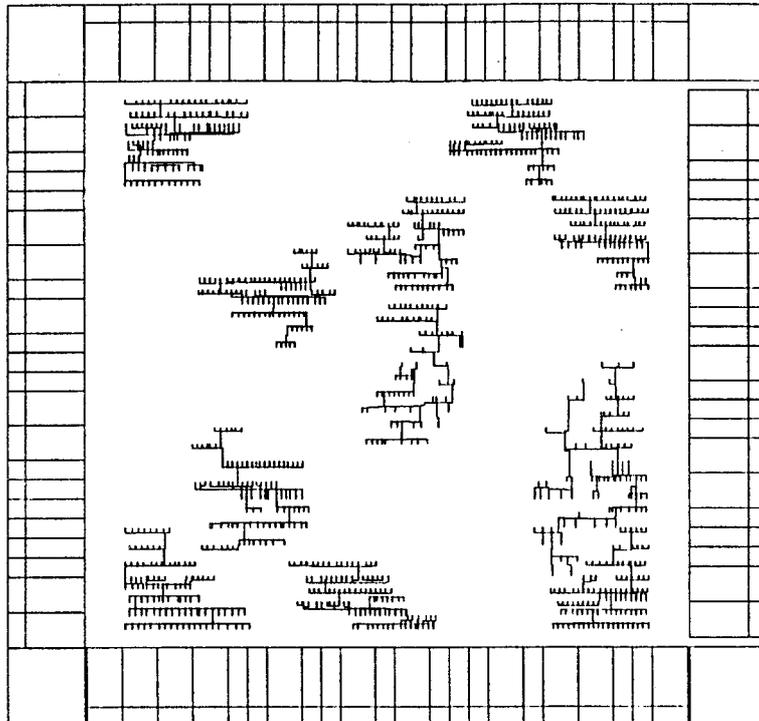


Abb.6 Layoutskizze großer Netze mit extrem kurzen Leitungen

Im folgenden wird erklärt, wie sich derartige Layoutverbesserungen der Netzliste mit Semi-Custom Entwurfssystemen realisieren lassen.

3 Lösungsansatz

Um die großen Netze einer Gatternetzliste layoutabhängig zu optimieren, verzichtet man zunächst darauf, zusätzliche Treiber einzusetzen und läßt als erste Näherung ein Layout ohne Baumstrukturen generieren. In dem so erzeugten Layout lokalisiert man dann die jeweils gemeinsam zu treibenden Gatter und konstruiert dabei layoutabhängig geeignete Treiberbäume, die dann in die ursprüngliche Netzliste integriert werden. Auf diese Weise ergibt sich eine Netzliste mit einer sinnvollen Realisierung großer Netze, die dann erneut vom Semi-Custom System in ein Layout überführt wird. Diese prinzipielle Vorgehensweise ist in Abbildung 7 skizziert.

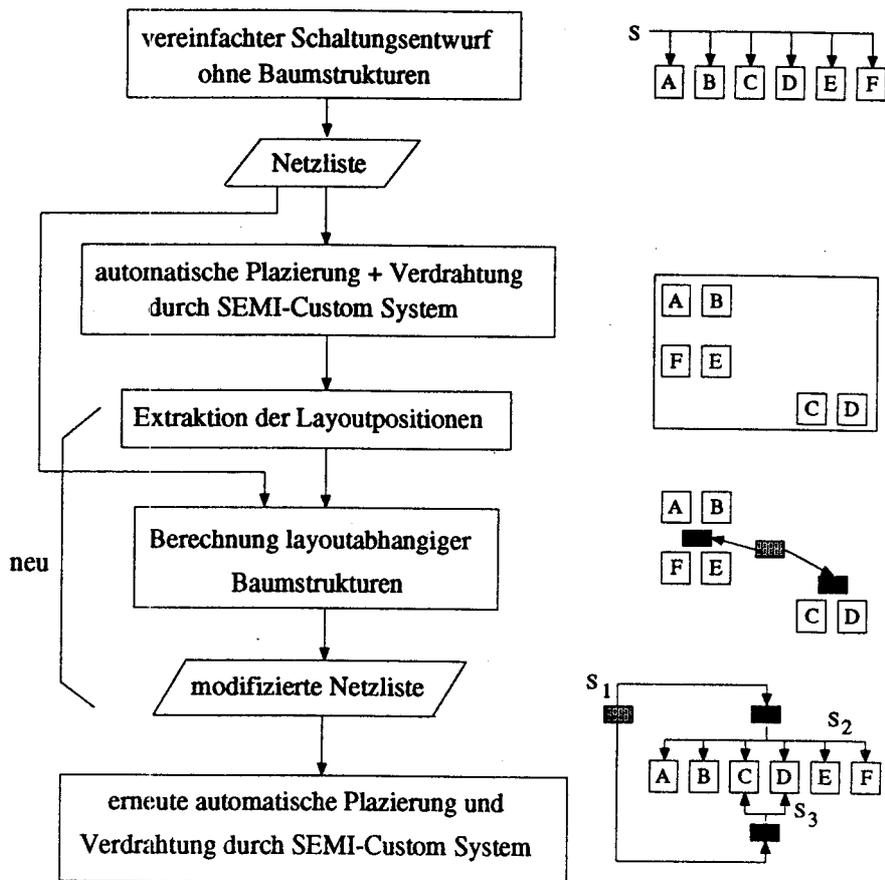


Abb.7 Layoutabhängige Optimierung einer Gatternetzliste

4 Ein Tool zur automatischen Netzlistenoptimierung

In [3] wurde ein Programmpaket entworfen und implementiert, mit dessen Hilfe sich die benötigte Netzlistenoptimierung automatisieren läßt. Als Eingabe werden die zu optimierende Netzliste und die automatisch erzeugten Platzierungsinformationen benutzt. Daraus ermittelt das Programm zunächst die Lage aller an einem Netz angeschlossenen Gatter. In Abbildung 8 ist exemplarisch eine solche Gattermenge dargestellt. Zusätzlich wird eine relativ feine Kachelung über das Layout gelegt und die Zuordnung der relevanten Gatter zu Kacheln wird ermittelt.

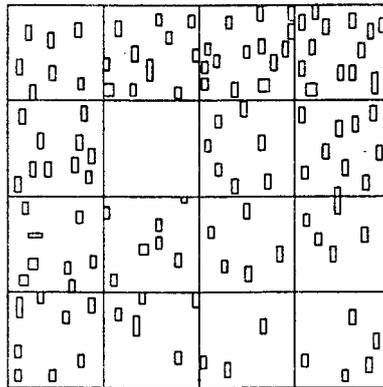


Abb.8 Gattermenge und Kachellung des Layouts

Anschließend wird die Anzahl der benötigten Treiber berechnet und durch Zusammenfassen benachbarter belegter Kacheln wird eine Aufteilung der Gatter auf eine den Treibern entsprechende Anzahl von Gebieten vorgenommen. Dabei wird mit Hilfe von Auswahlheuristiken erreicht, daß jeweils die Kacheln eines Gebiets möglichst nahe zusammen liegen. Die Gebietsmittelpunkte definieren dann die Treiberpositionen für die unterste Hierarchiestufe des Treiberbaums. In Abbildung 9 sind diese Positionen am Beispiel dargestellt.

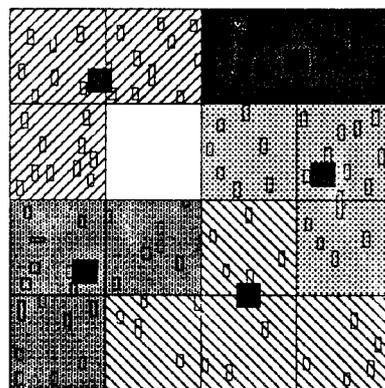


Abb.9 Treiberpositionen in den aufgeteilten Gebieten

Wendet man das Verfahren nun erneut auf diese Gatterpositionen an, so ergibt sich die nächst höhere Hierarchiestufe des Treiberbaums. Durch fortgesetzte Anwendung entsteht schließlich die vollständige Baumstruktur (Abbildung 10).

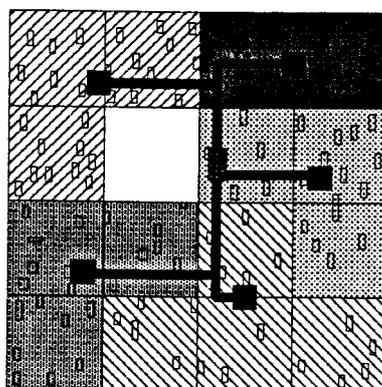


Abb.10 Vollständige Baumstruktur

Die Bäume sämtlicher großen Netze werden schließlich mit den dazugehörigen Layoutpositionen in die geplante Gatternetzliste eingebaut. Mit Hilfe der so modifizierten Netzliste läßt sich dann vom Semi-Custom System ein verbessertes Layout generieren. Gegebenenfalls kann der gesamte Vorgang mehrfach iteriert werden, bis keine Verbesserungen mehr erreicht werden.

5 Ergebnisse

Die bisher durchgeführten Experimente mit der CADENCE-Entwurfsumgebung und dem neuen Optimierungstool haben gezeigt, daß im allgemeinen eine Iteration ausreicht, um zu guten Ergebnissen zu kommen. Dabei ist es allerdings empfehlenswert die berechneten idealen Treiberpositionen dem Platzierungsalgorithmus als Grobplatzierung vorzugeben. Andernfalls entstehen durch ungünstigere Platzierungen weniger gute Verdrahtungen. Als Beispiel für diesen Effekt ist eine entsprechende vom CADENCE-System erzeugte Verdrahtung in Abbildung 11 angegeben.

Außerdem muß bei der Verdrahtung verhindert werden, daß große Netze mit der Sonderbehandlung für Taktsignale realisiert werden, da sonst die Verdrahtung extrem ungünstig über die komplette Chipbreite realisiert wird. In Abbildung 12 ist als Beispiel ein Netz mit der Sonderbehandlung *clock* skizziert. Man erkennt deutlich, daß durch die unflexible Struktur extrem lange und unnötige Leitungsführungen entstehen.

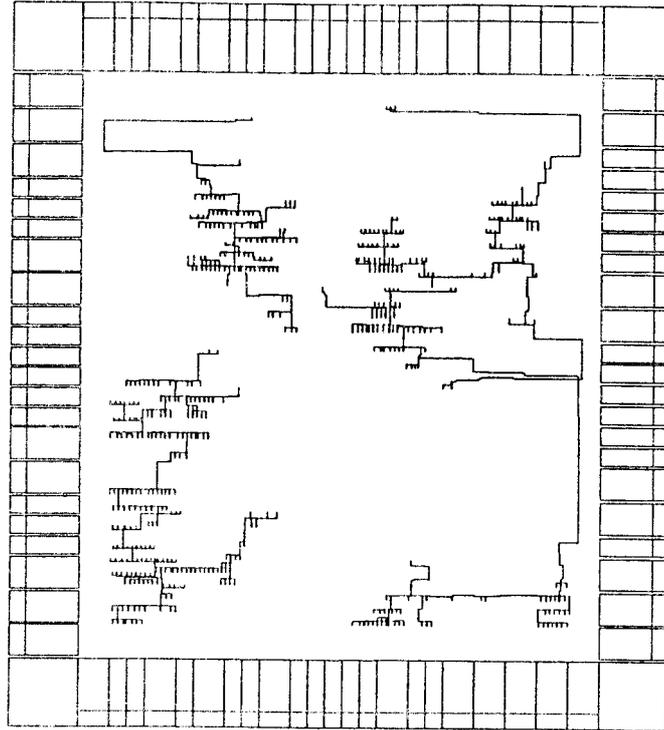


Abb.11 Automatisch generiertes Layout ohne Grobplazierung der Treiber

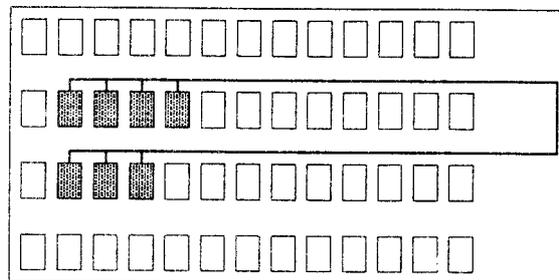


Abb.12 Realisierung eines großen Netzes, mit der Sonderbehandlung für Clock-signale

Der Vergleich der verschiedenen automatisch generierten Layoutskizzen mit der Lösung aus Abbildung 6 demonstriert somit deutlich, wie notwendig es ist, vorhandene Semi-Custom Systeme um zusätzliche Tools zu ergänzen, um auch für komplexe zeitkritische Chipentwürfe zu akzeptablen Layoutlösungen zu kommen.

Literaturverzeichnis

- [1] W. Rülling P. Leber. *Floorplan- und Layouterstellung für einen Chip zur Berechnung von Faltungintegralen*. Interner Bericht, Fachhochschule Furtwangen, 1993.
- [2] CADENCE. *Reference Manual, Automatic Place and Route, Volume 1 and 2*. CADENCE, 1989.
- [3] M. Faas. *Layoutgenerierung großer Netze für schnelle Signalprozessoren*. Diplomarbeit, Fachhochschule Furtwangen, 1992.

MPC-Gruppe der baden-württembergischen Fachhochschulen
Sommersemester-Workshop 1993

Ein Demonstrationssystem für die Bilddatenübertragung

Prof. Dr.-Ing. Norbert Höptner
Fachhochschule Pforzheim

Kurzfassung:

Im Rahmen eines "Innovativen Projekts" ist ein ASIC entwickelt und hergestellt worden, der eine serielle adressierbare Datenübertragung einfach und kostengünstig mit dem DQDB-Zugriffsverfahren ermöglicht. Eine Besonderheit dieses Bausteins ist die Echtzeitfähigkeit der Datenübertragung, d. h. ein Teilnehmer kann erst dann zum zweiten Mal senden, wenn alle anderen sendewilligen Teilnehmer ihren Datenrahmen absenden konnten. Die dazu erforderliche Arbitrierung erfolgt ohne zusätzlichen Controller, sondern ist auf alle Busteilnehmer verteilt (verteilte Warteschlange = distributed queue) und erfordert eine Doppelbus-Verbindung (dual bus). Das Übertragungsmedium können Kupfer- (BNC-) und Lichtwellenleiter sein. Auch eine gemischte Verkabelung ist ohne Zusatzaufwand möglich. Die dazu erforderlichen Codierungen sind im ASIC integriert, wobei eine Verbesserung gegenüber dem FDDI-Standard erreicht wurde. Die parallele Anwenderschnittstelle ist wahlweise auf 8, 16, 32 und 64 Bit einstellbar. Beispielsweise ist so eine Datenformat-Konvertierung zwischen unterschiedlichen Prozessorarchitekturen (8 Bit --> 64 Bit) durch die Datenübertragung ohne Zusatzaufwand möglich! Dieser Entwicklungsstand wurde auf dem letzten MPC-Workshop (Wintersemester 1992/93) vorgestellt (Larsch, Höptner).

Die nun bis zum Abschluß des Projektes (30.6.93) erfolgte Weiterentwicklung betrifft drei Punkte:

- 1) Das ASIC-Design wurde im Rahmen einer Diplomarbeit auf LCA-Design (Xilinx X4000-Serie) übertragen.

Der erwartete Vorteil war die Weitergabe eines Grund-Designs an interessierte Anwender, die nun selbständig ihre spezielle Applikationsschnittstelle dazu entwickeln können. Diese Vorgehensweise stellte sich jedoch durch gravierende Nachteile als unbrauchbar heraus:

- statt 1 ASIC werden 2 LCAs und Zusatzhardware benötigt
- die Kosten einer Schnittstelle steigen auf das 10fache
- die Design-Umsetzung ASIC --> LCA ist sehr aufwendig.

2) Nachweis der Funktionalität des ASICs. Dazu wurde ein Bilddatenübertragungssystem realisiert, mit dem die Übertragung von Bilddaten zwischen bis zu 256 PCs möglich ist. Das Bilddatenübertragungssystem besteht aus jeweils:

- 1 Transputer-PC-Board (Eigenentwicklung)
- 1 ASIC-Interface-Board (Eigenentwicklung)
- Software für Steuerung und Systembedienung.

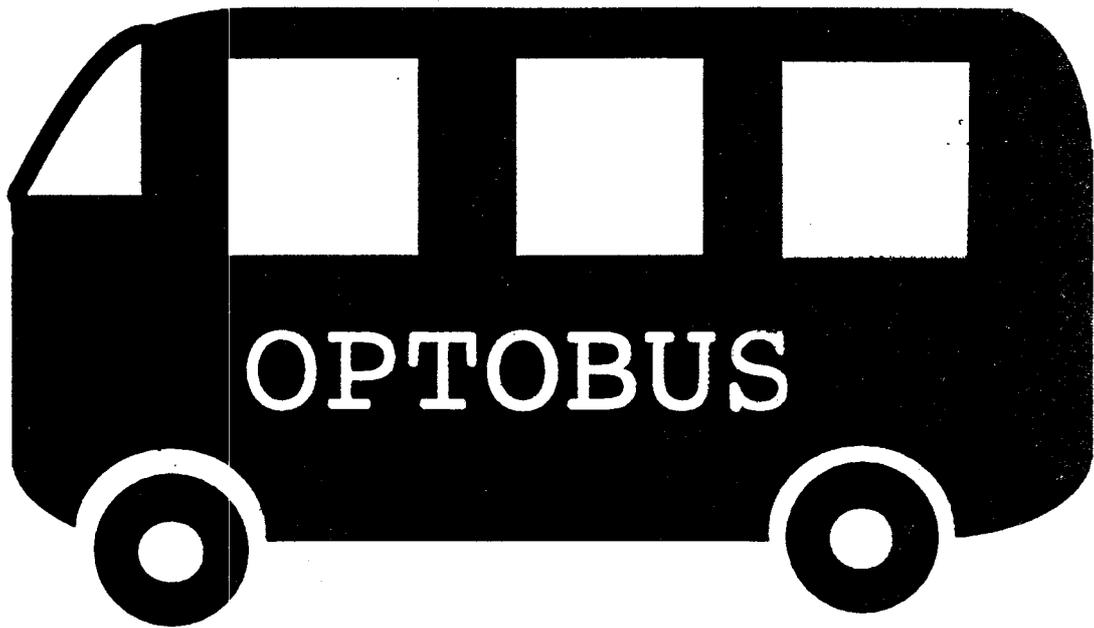
Die Bedienoberfläche erlaubt den Transfer unterschiedlicher Bilddaten zu einstellbaren Bus-Teilnehmer-Adressen. Damit konnte der Nachweis erbracht werden, daß der ASIC richtig funktioniert. Weiterhin steht in Form der entwickelten Platinen eine PC-Anwender-Schnittstelle für beliebige Datenübertragung über dieses Bussystem zur Verfügung.

Im Rahmen des Workshops wird ein Bilddatenübertragungssystem mit 3 vernetzten PCs demonstriert.

3) Durch das Bilddatenübertragungssystem konnten die tatsächlichen Grenzwerte und Funktionsfehler des ASICs ermittelt werden:

- Die maximale Taktrate beträgt bei 5 Volt Versorgungsspannung und 27 Grad Celsius Raumtemperatur 48 MHz (statt erwarteten 50 MHz). Dies ist durch Addierung von Laufzeiten bei der im ASIC realisierten Ablaufsteuerung erklärbar. Die unidirektionale Nutzübertragungsrate beträgt damit 3.5 MByte/s, die bidirektionale Übertragungsrate erreicht 7 MByte/s.
- Die Codierung des "Token" (5 aufeinanderfolgende "1"), mit dem eine Funktionsprüfung des Systems erreicht wird, erwies sich als nicht eindeutig. Um diesen Konzeptfehler zu umgehen, dürfen derzeit die letzten 3 Nutzdatenbits nur den Inhalt "0" haben!
- Die Teilnehmerstation mit der Adresse "1" reagiert auf viele andere Adressen im Datenrahmen. Die Ursache hierfür konnte noch nicht ermittelt werden.

Zusammenfassend kann festgestellt werden, das in diesem umfangreichen Projekt die gesteckten Ziele erreicht wurden und nun ein industriell verwendbares Datenübertragungssystem zur Verfügung steht. Durch die Einbindung vieler studentischer Arbeiten und das Gewinnen fachlicher Erkenntnisse beim Projektleiter bedeutet dieses Projekt auch einen beachtlichen Zugewinn an fachhochschulspezifischem Lehr-Potential.



IIT FH Karlsruhe

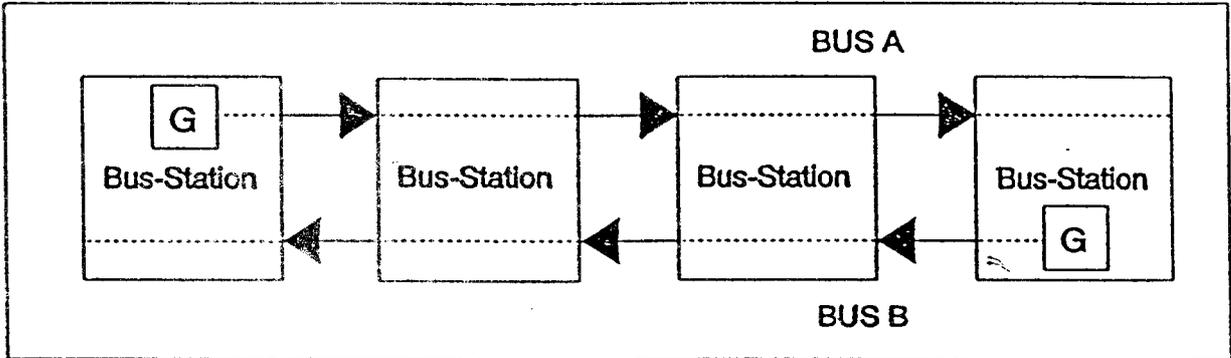


Bild 2: DQDB - Bus-Topologie

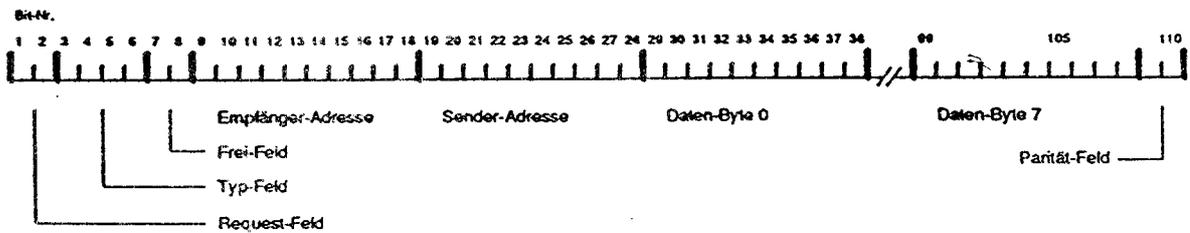


Bild 3: Format des Opto-Bus-Datenrahmens

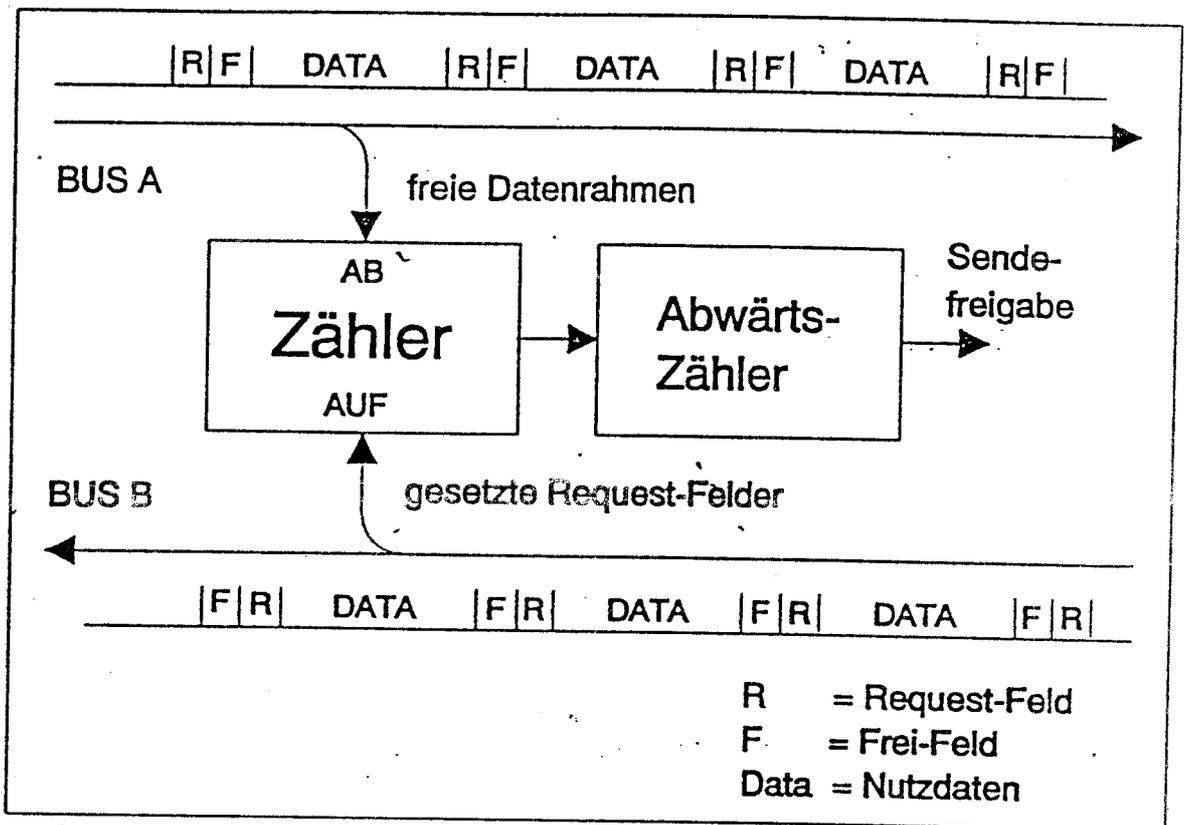


Bild 4: Das Prinzip der verteilten Warteschlange

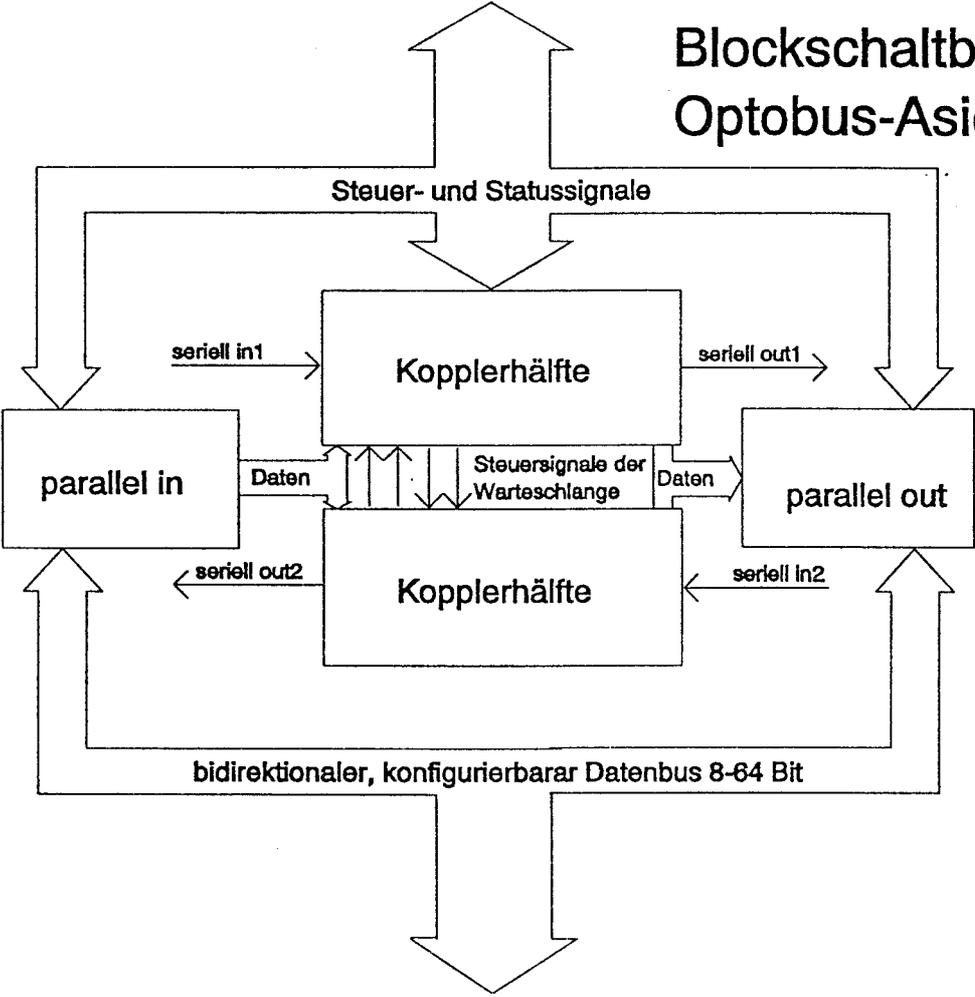
Realisierung der integrierten Schaltung

Design: MENTOR-Graphics CAEE-Labor
auf Apollo/Domain - HP Prof. Ritzert, FH KA

Herstellung: IMS, Stuttgart MPC-Gruppe
INTERMETALL, Freiburg Prof. Führer, FH ULM

Prozess: 1,2 μ m IMS-Gate-Forrest
144 pin Gehäuse (Keramik)
9600 Gatterfunktionen
53000 aktive Transistoren
---> 76% Chip-Ausnutzung

Blockschaltbild Optobus-Asic



System-Merkmale:

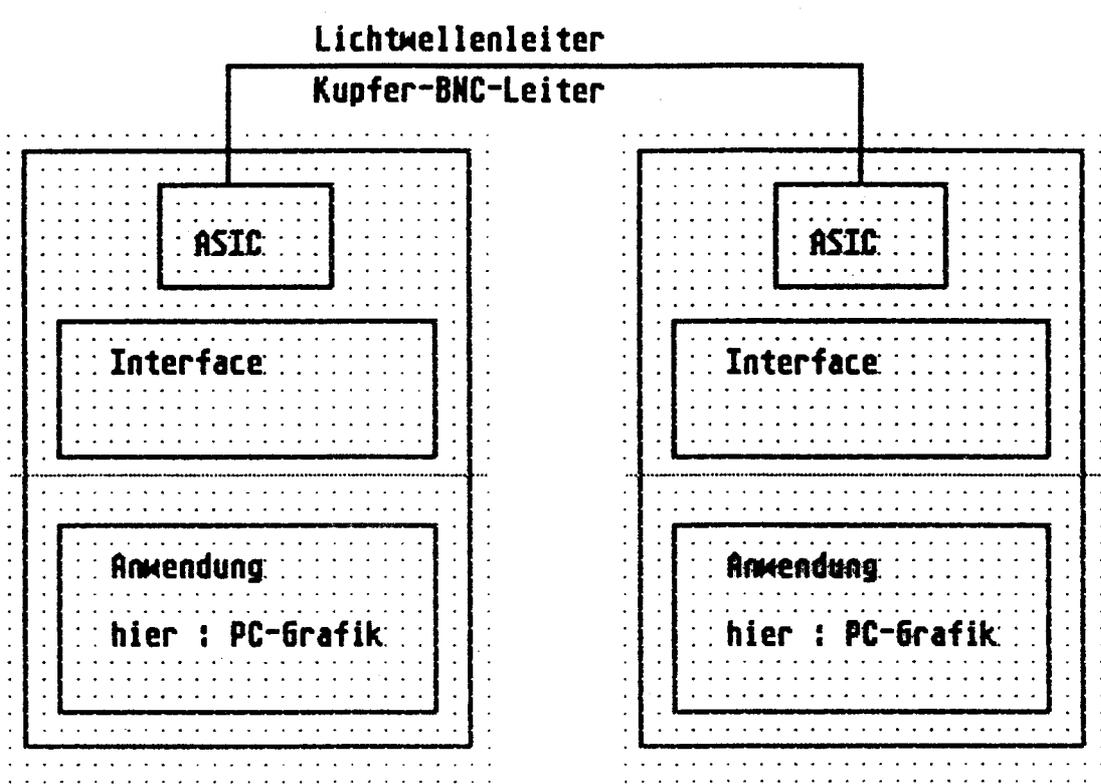
- **50 MBit/s Brutto- 29 MBit/s Netto-Übertragungsrate**
 - **freie Adressierbarkeit**
 - **2 - 256 Stationen**
 - **gleiche Bandbreitenverteilung auf alle Stationen (DQDB-Verfahren)**
 - **echtzeitfähig**
- **parallele Schnittstelle, 8 - 64 Bit breit konfigurierbar**
 - **automatische Initialisierung und Einsynchronisation aller Stationen nach dem Einschalten**
 - **automatische Fehlererkennung und Auftrennung des Bussystems an der Fehlerstelle**
 - **automatische Neuinitialisierung nach Beseitigung des Fehlers**

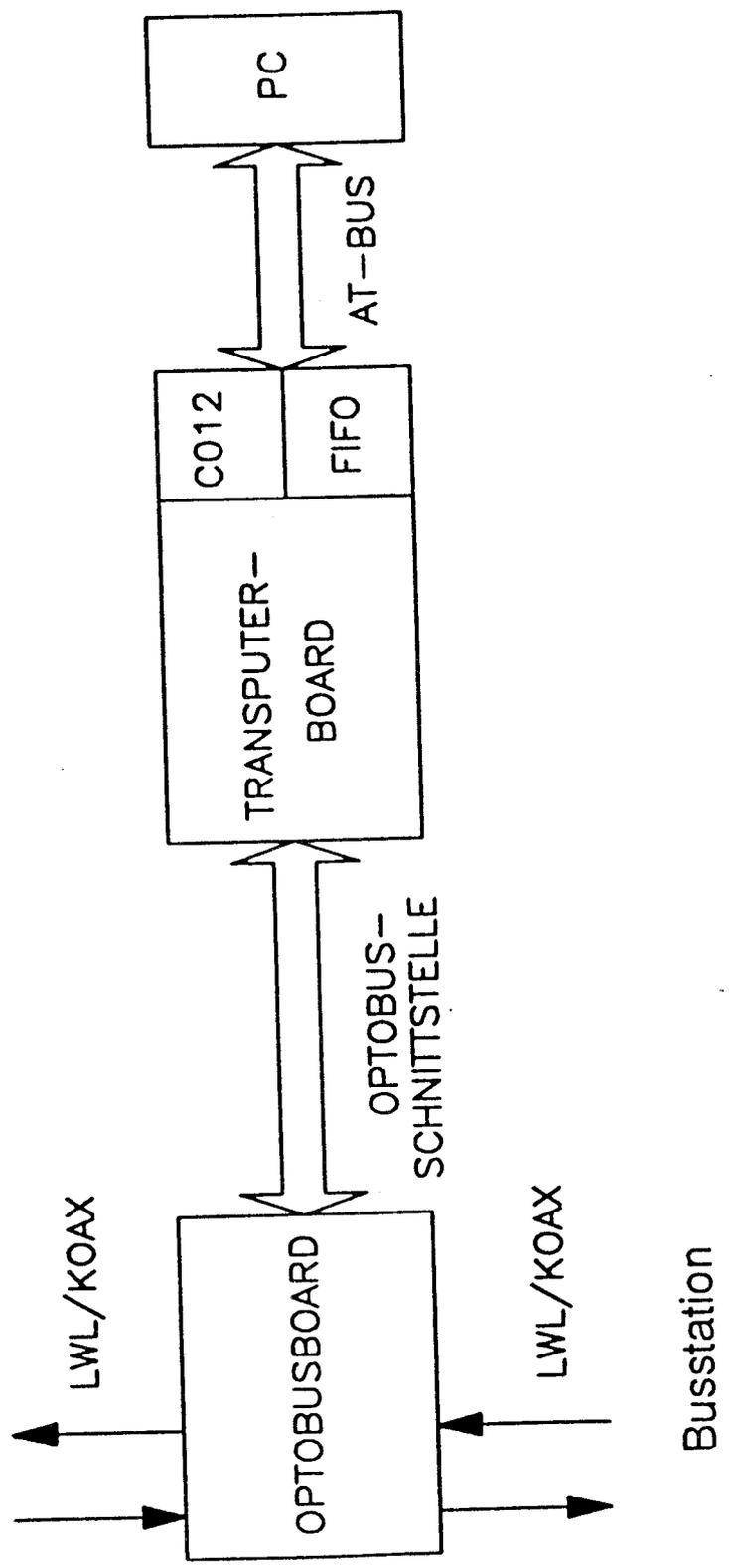
Vergleichs-Design-LCA-X4000

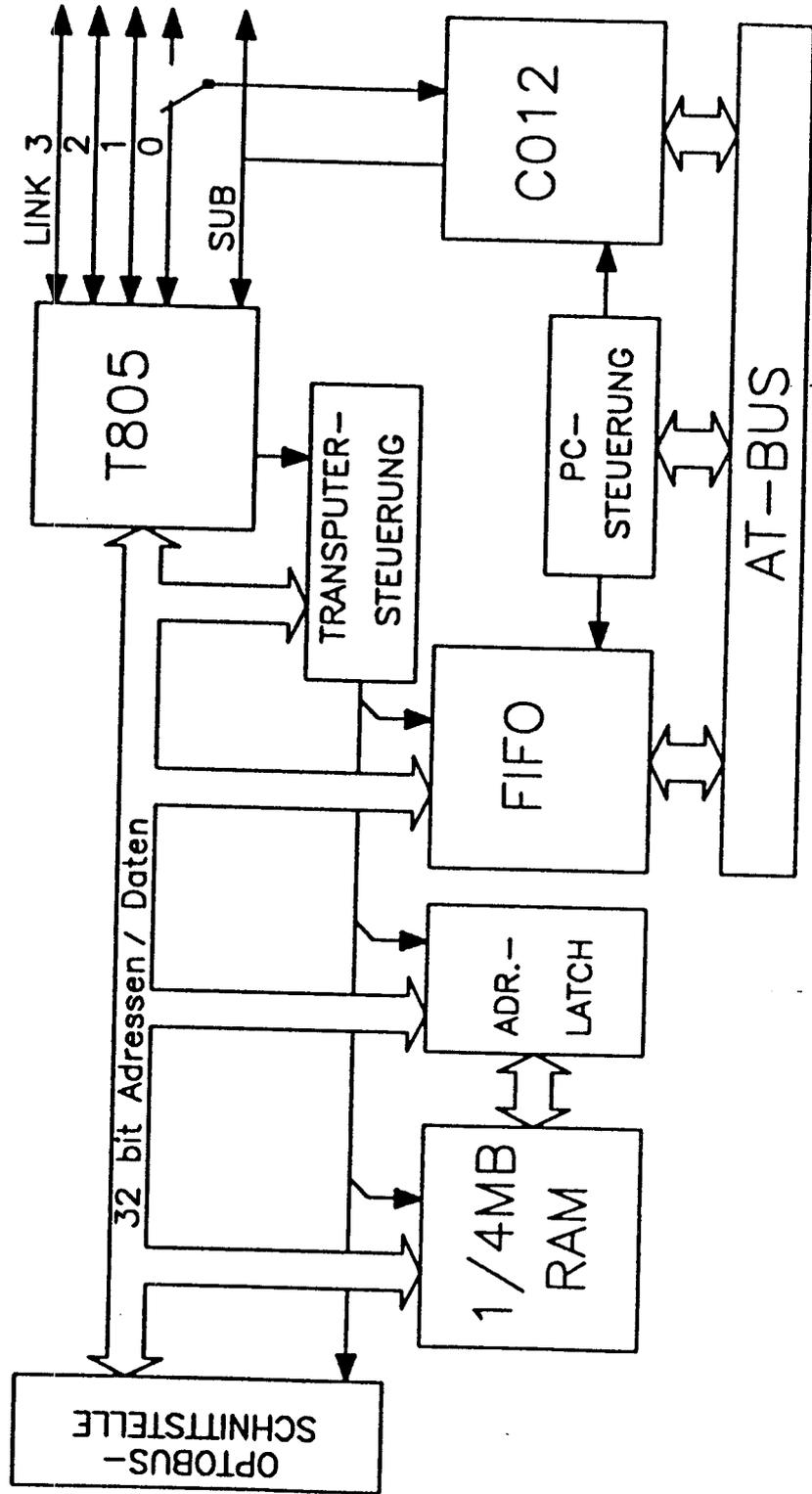
Vorteil: Schaltfunktionen sind programmierbar
und dynamisch ladbar
---> Anwender kann eigene Schnittstelle
selbst erstellen

Nachteil: ASIC-Design nur mit Schwierigkeiten auf LCA
übertragbar

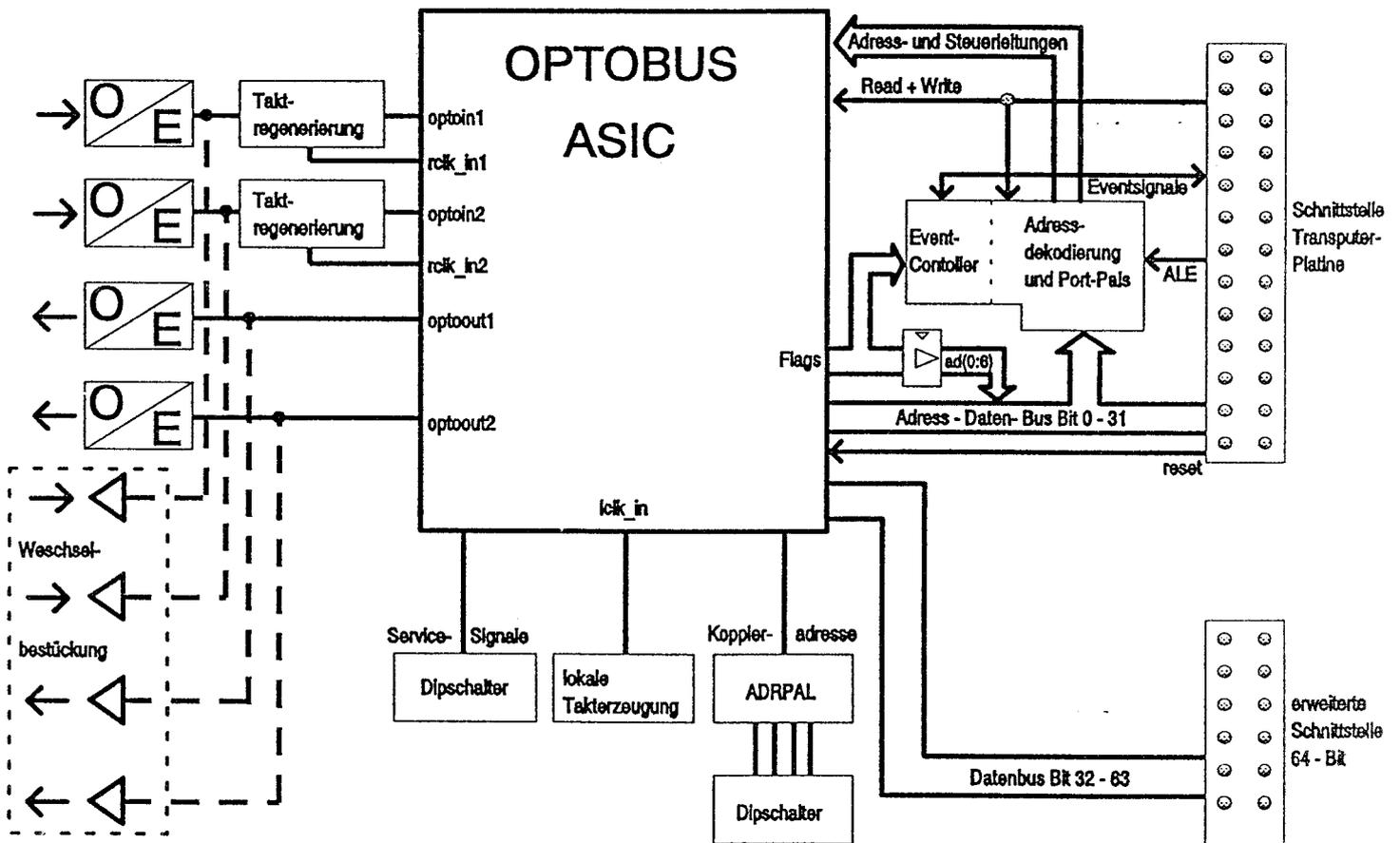
1 ASIC --->	2 LCA X4000
	2 Coder/Decoder-Speicher
	nur 8-Bit-Anwender-Schnitt- stelle
	4fache Layout-Fläche
	höhere Kosten bei geringerer Funktionalität:
ca. 100 DM	ca. 2000 DM

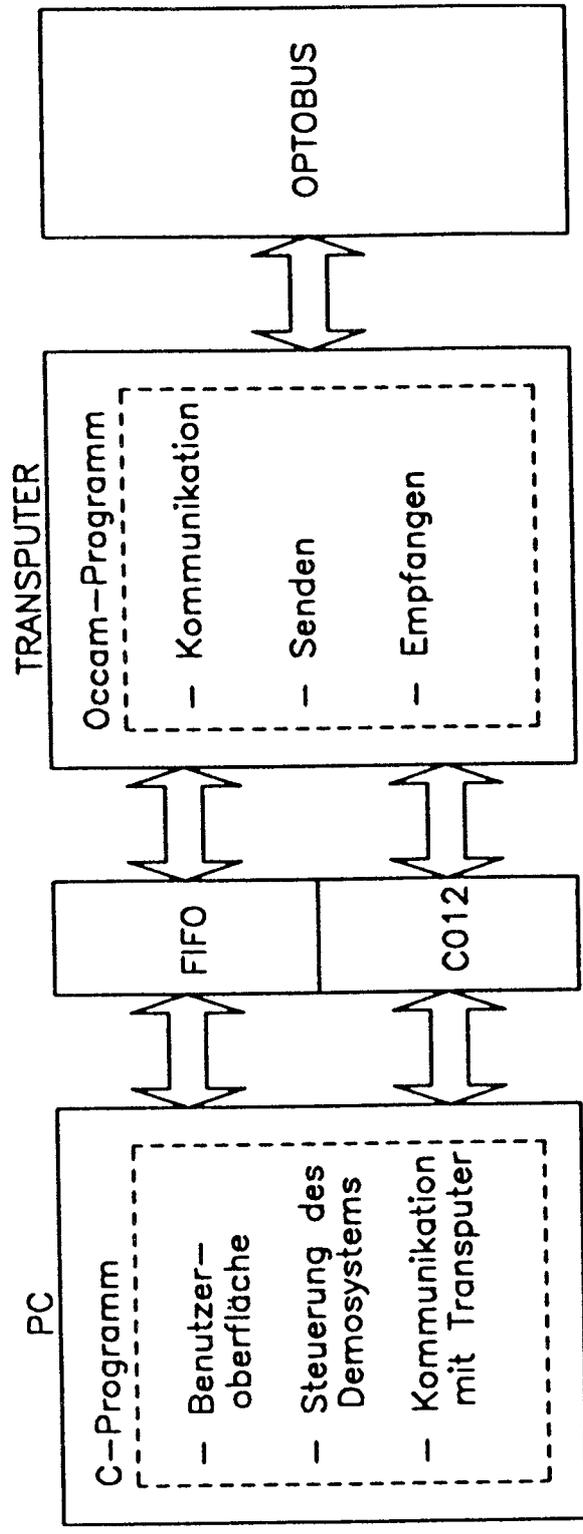




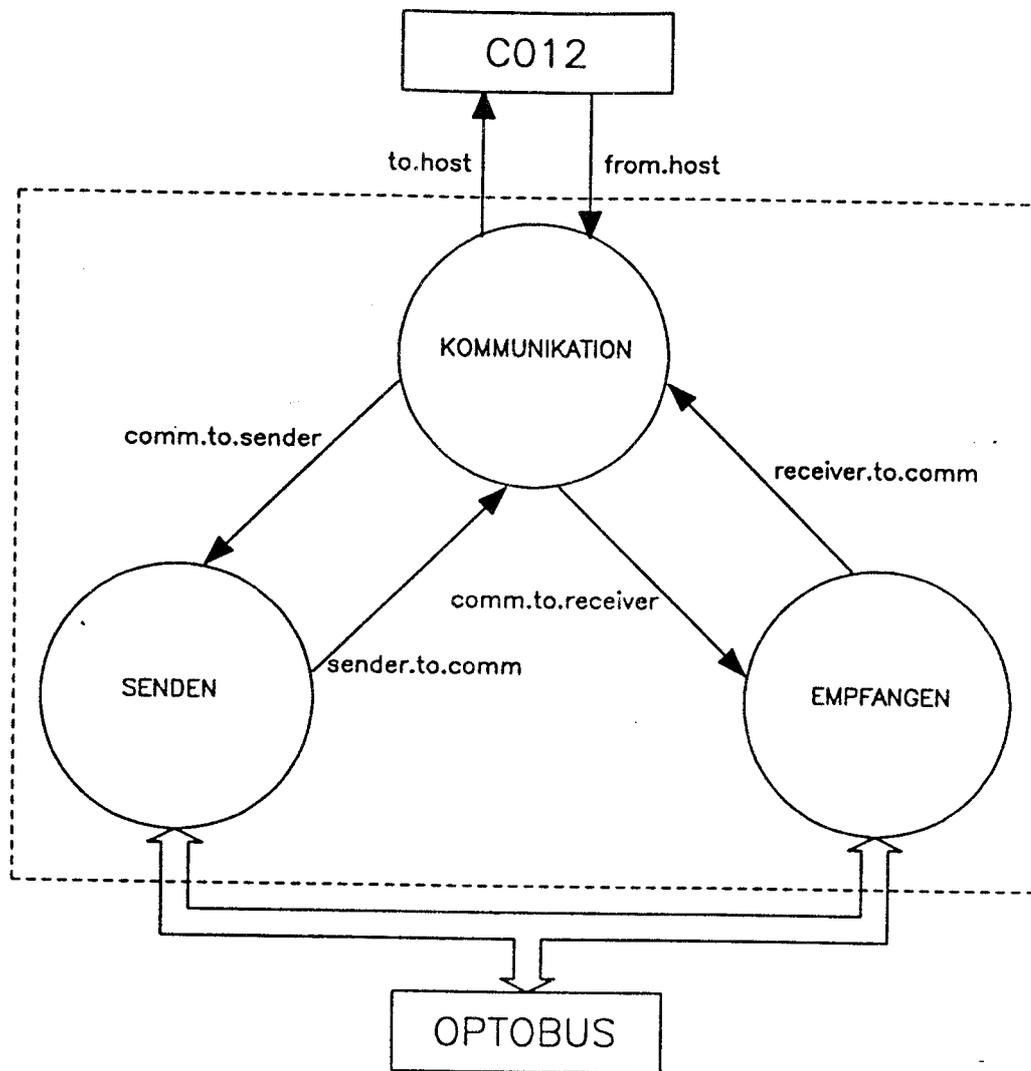


Blockschaltbild des Transputerboards

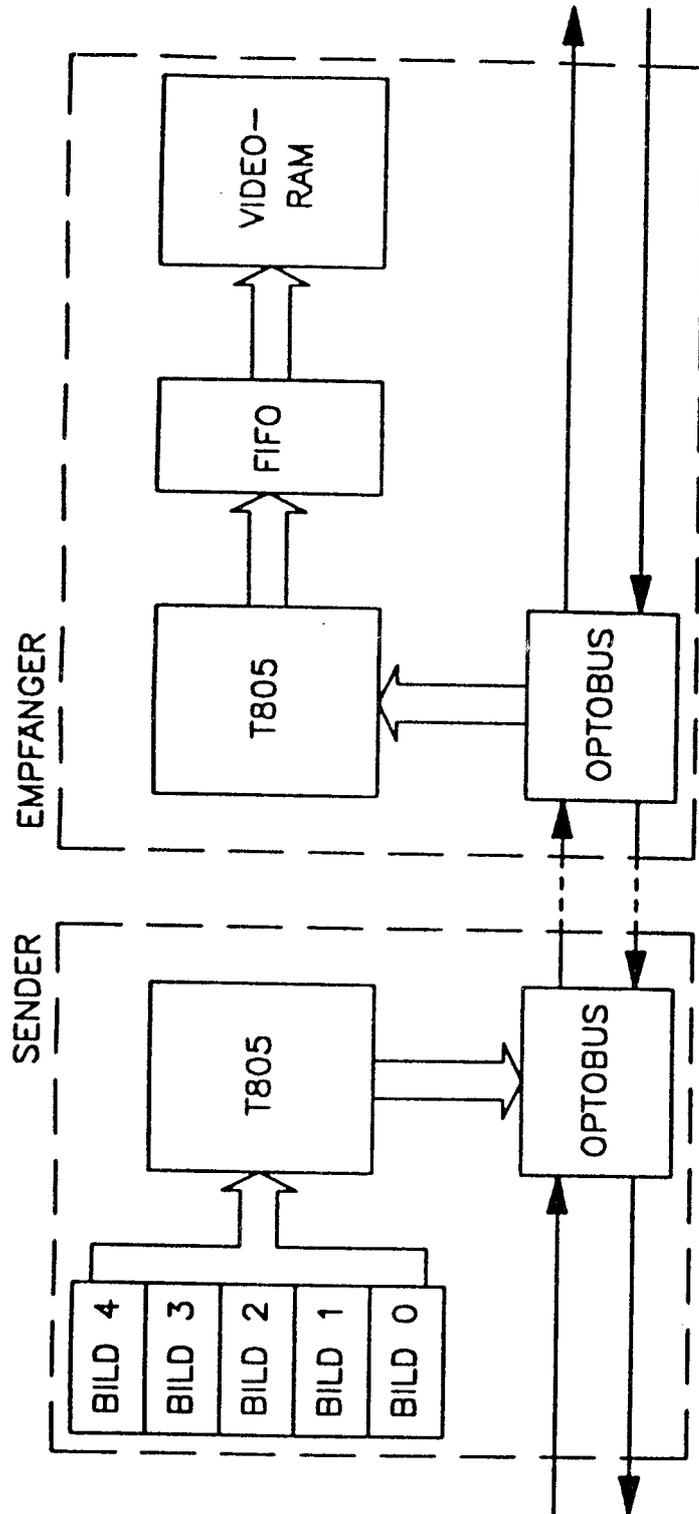




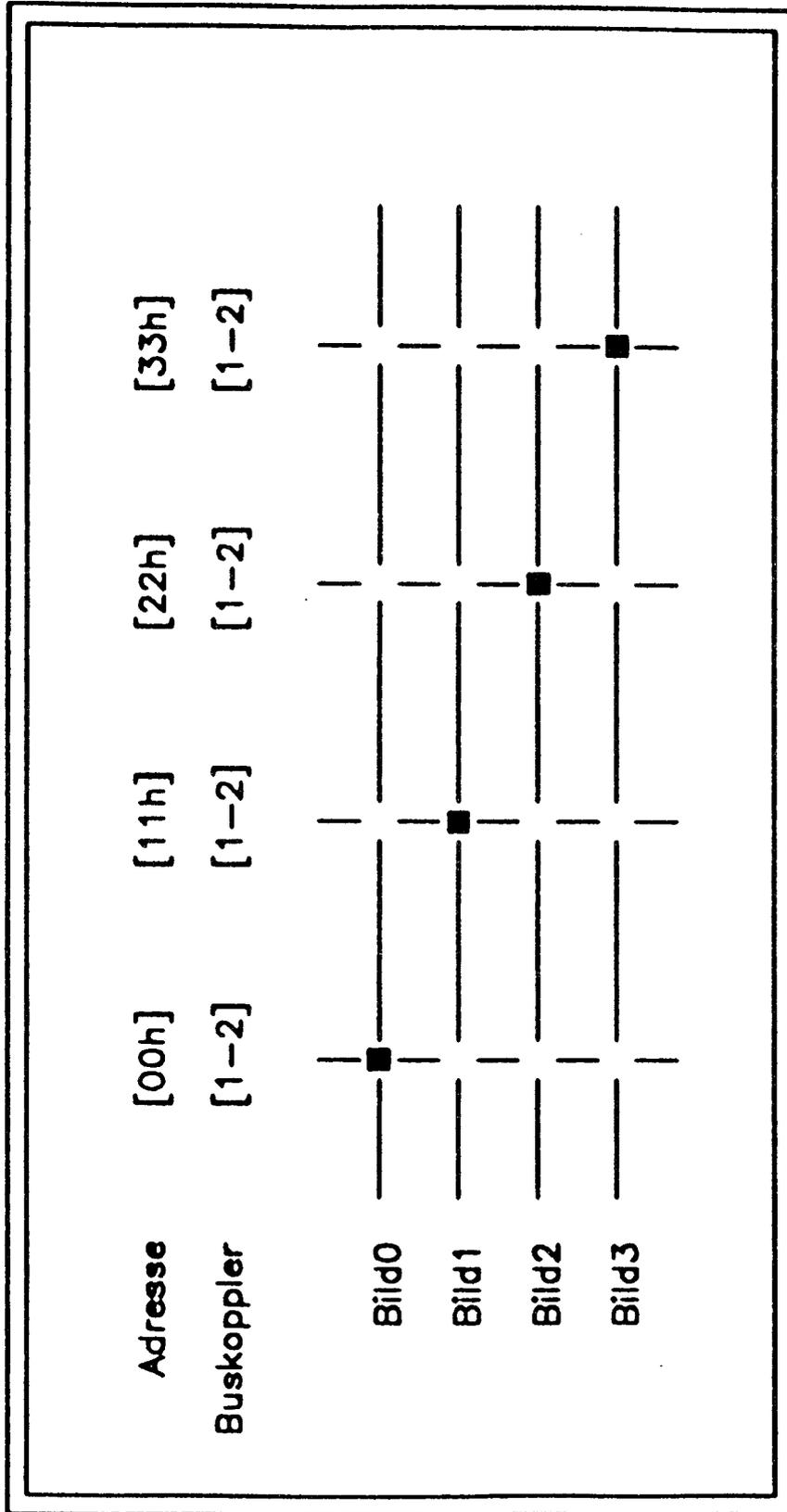
Software-Struktur



Struktur Occam-Software



Datenfluß im Demosystem



Aktionsfeld des Sendeprogramms

**Aufbau und Spice - Simulation von
nichtlinearen chaotischen Schaltungen
am Beispiel von Chua's Circuit**

Bernd Maurer und Ralf Hägele

**Fachhochschule für Technik Esslingen
Außenstelle Göppingen
Fachbereich Elektronik/Mikroelektronik**

Betreuer

Prof. Dr.-Ing H. Khakzar

Inhalt

I. Einführung Chaos

II. Nichtlinearitäten

III. Chua's Circuit

IV. Demonstration

I. Einführung Chaos

In diesem Abschnitt werden die Fragen

- Was ist Chaos ?
- Wie entsteht Chaos ?

behandelt.

Was ist Chaos ?

Wir reden dann von Chaos, wenn:

- das Ergebnis einer Funktion für längere Zeitperioden nicht mehr vorhersagbar ist.
- kleinste Abweichungen in den Eingangsgrößen große Unterschiede in den Ausgangsgrößen verursachen.

Beispiele für chaotisches Verhalten sind:

- Wetter
- Weltwirtschaft
- Bevölkerungsmodelle
- chemische Reaktionen

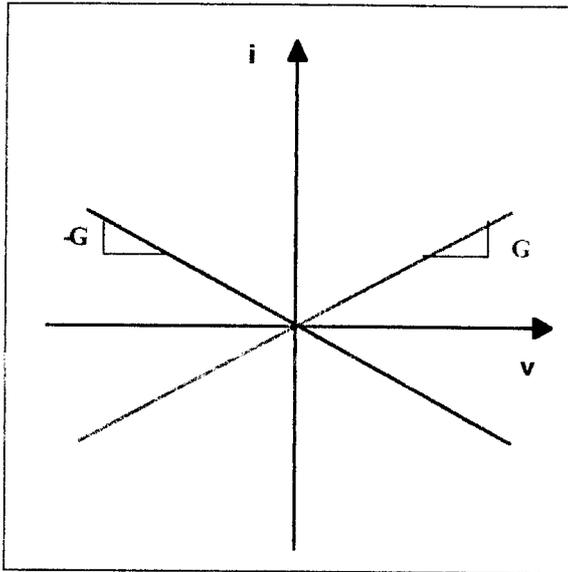
Wie entsteht Chaos ?

Die Entstehung von Chaos konnte bisher nur in nichtlinearen dynamischen Systemen nachgewiesen werden. In diesen Systemen führt der Weg ins Chaos immer über ein bestimmtes vergleichbares Verhaltensschema. Das chaotische Verhalten ist selbstverständlich immer unterschiedlich. Das Ziel der Chaosforschung ist es den Übergang von Ordnung ins Chaos zu studieren. Dabei sollen Erkenntnisse über die Mechanismen der Selbstähnlichkeit und der Selbstorganisation gewonnen werden.

II. Nichtlinearitäten

Erzeugung eines negativen Widerstandes:

Ein Widerstand kann durch seine Strom - Spannungs - Kennlinie dargestellt werden. Demnach läßt sich der Leitwert eines Widerstandes in einer Spannungs - Strom - Kennlinie darstellen.

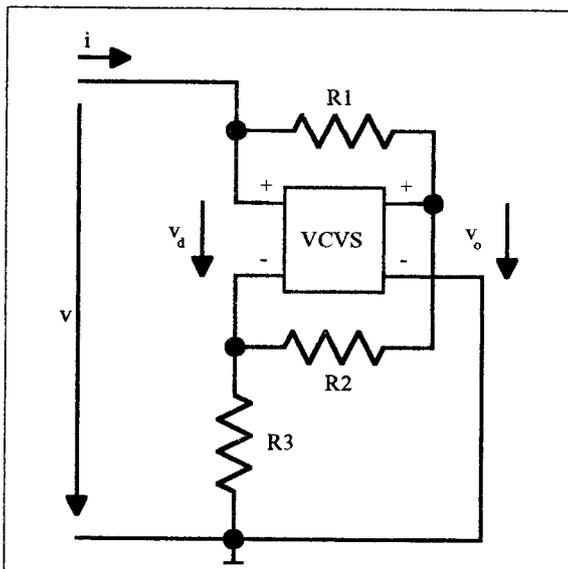


Der positive Leitwert G entspricht der Steigung der grünen Kennlinie.

Der negative Leitwert $-G$ entspricht der Steigung der roten Kennlinie.

Wenn man nun einen negativen Widerstand erzeugen will, so muß man versuchen die rote Kennlinie nachzubilden.

Dies gelingt, wenn man eine spannungsgesteuerte Spannungsquelle wie folgt beschaltet:



Nach der Knotenregel gilt:

$$i = \frac{1}{R_1} * (v - v_o)$$

Nach der Maschenregel gilt:

$$v = v_d + \left(\frac{R_3}{R_2 + R_3} \right) * v_o$$

Die Übertragungsfunktion der spannungsgesteuerten Spannungsquelle lautet:

$$v_o = A * v_d \Rightarrow v_d = \frac{v_o}{A}$$

Wenn man nun die Übertragungsfunktion in die Maschenregel einsetzt, so erhält man folgende Gleichung:

$$v = \frac{1}{A} * v_o + \left(\frac{R_3}{R_2 + R_3} \right) * v_o = \left(\frac{1}{A} + \frac{R_3}{R_2 + R_3} \right) * v_o$$

oder anders dargestellt:

$$v = \left[\frac{(R_2 + R_2) + R_3 * A}{A * (R_2 + R_3)} \right] * v_o = \left[\frac{R_2 + R_3 * (1 + A)}{A * (R_2 + R_3)} \right] * v_o$$

nach v_o umgestellt ergibt dies:

$$v_o = \left[\frac{A * (R_2 + R_3)}{R_2 + R_3 * (1 + A)} \right] * v$$

Wird diese Gleichung in die Knotenregel eingesetzt erhält man eine Gleichung vom Typ $y=mx$:

$$i = \frac{1}{R_1} * \left\{ v - \left[\frac{A * (R_2 + R_3)}{R_2 + R_3 * (1 + A)} \right] * v \right\} = \frac{1}{R_1} * v * \left[1 - \frac{A * (R_2 + R_3)}{R_2 + R_3 * (1 + A)} \right]$$

$$i = \frac{1}{R_1} * \left[\frac{R_2 + R_3 * (1 + A) - A * (R_2 + R_3)}{R_2 + R_3 * (1 + A)} \right] * v$$

$$i = \frac{1}{R_1} * \left[\frac{R_2 + R_3 + R_3 * A - R_2 * A - R_3 * A}{R_2 + R_3 * (1 + A)} \right] * v = \frac{1}{R_1} * \left[\frac{R_2 + R_3 - R_2 * A}{R_2 + R_3 * (1 + A)} \right] * v$$

$$i = \frac{1}{R_1} * \left[\frac{R_2 * (1 - A) + R_3}{R_2 + R_3 * (1 + A)} \right] * v = \left[\frac{R_2 * (1 - A) + R_3}{R_1 * [R_2 + R_3 * (1 + A)]} \right] * v$$

Wenn die Verstärkung A als unendlich angenommen wird vereinfacht sich die Gleichung zu:

$$\lim_{A \rightarrow \infty} i \approx \left(-\frac{R_2}{R_1 * R_3} \right) * v$$

Wenn man zwei Widerständen den selben Wert zuordnet, so führt dies bei

- $R_1=R_2$ zu: $i \approx -\frac{1}{R_3} * v$
- $R_2=R_3$ zu: $i \approx -\frac{1}{R_1} * v$
- $R_3=R_1$ zu: $i \approx -\frac{R_2}{R_1^2} * v$

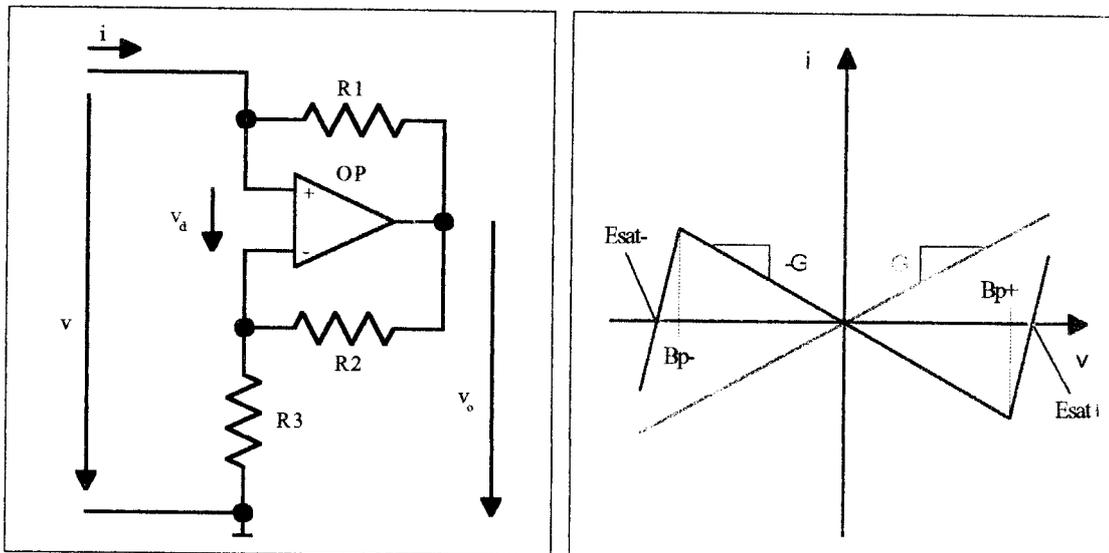
Die ersten beiden Fälle sind hilfreich da bei ihnen jeweils nur ein Widerstand in seinen negativen Wert gewandelt wird.

Eine spannungsgesteuerte Spannungsquelle ist ein ideales Bauelement. Dies bedeutet, daß man sie zuerst mit reellen Bauelementen nachbilden muß.

Ein Operationsverstärker ist in seinem linearen Bereich eine gute Annäherung an eine spannungsgesteuerte Spannungsquelle. Die Übertragungsfunktion eines Operationsverstärker ist dreigeteilt und lautet im:

- negativen Sättigungsbereich $v_o = \text{Esat-}$ $\frac{\text{Esat-}}{A} + v_{os} \geq v_d$
- linearen Bereich $v_o = A \cdot (v_d - v_{os})$ $\frac{\text{Esat-}}{A} + v_{os} \leq v_d \leq \frac{\text{Esat+}}{A} + v_{os}$
- positiven Sättigungsbereich $v_o = \text{Esat+}$ $v_d \geq \frac{\text{Esat+}}{A} + v_{os}$

Wenn man nun einen Operationsverstärker genauso beschaltet wie die zuvor beschriebene spannungsgesteuerte Spannungsquelle erhält man folgendes Ergebnis:



Für diese Spannungs - Strom - Kennlinie gilt:

$$i = \frac{1}{R_1} * \left\{ \left[\frac{R_3 + R_2 * (1 - A)}{R_2 + R_3 * (1 + A)} \right] * v + \left[\frac{A * (R_2 + R_3)}{R_2 + R_3 * (1 + A)} \right] * v_{os} \right\}$$

Für große Verstärkungen vereinfacht sich die Gleichung zu:

$$\lim_{A \rightarrow \infty} i \approx \left(-\frac{R_2}{R_1 * R_3} \right) * v + \left(\frac{R_2 + R_3}{R_1 * R_3} \right) * v_{os}$$

Der Wert des Breakpoint \$Bp+\$ berechnet sich wie folgt:

$$Bp+ = \left[\frac{R_2 + R_3 * (1 + A)}{A * (R_2 + R_3)} \right] * \text{Esat+} + v_{os} \quad \lim_{A \rightarrow \infty} Bp+ \approx \left(\frac{R_3}{R_2 + R_3} \right) * \text{Esat+} + v_{os}$$

Wegen der Symmetrie berechnet sich der Breakpoint \$Bp-\$ zu:

$$Bp- = -Bp+$$

Wenn nun mehrere negative Widerstände verwendet werden, lassen sich Reihenschaltungen und Parallelschaltungen bilden.

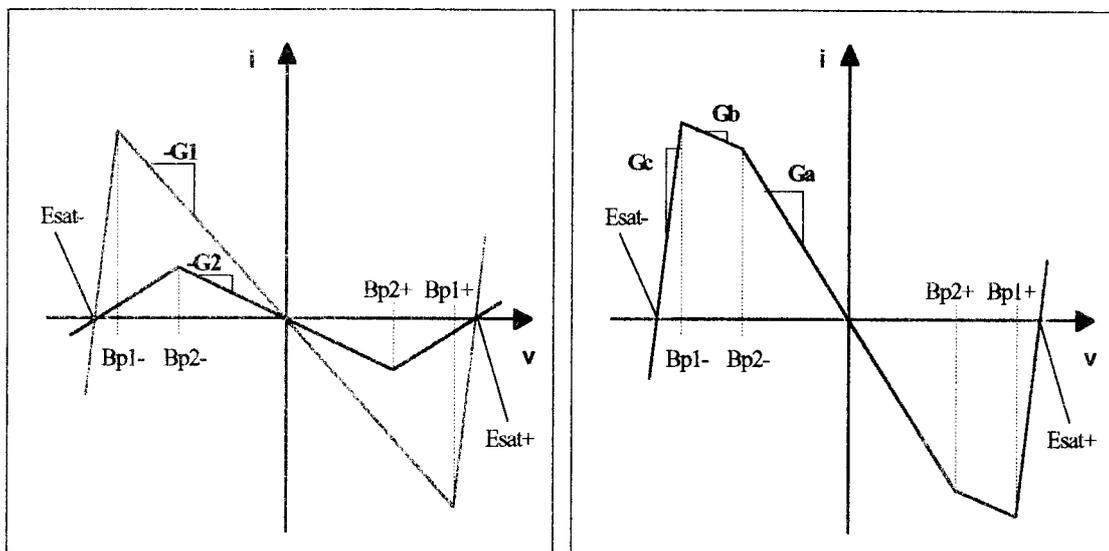
Für die Reihenschaltung gilt:

Die Spannungen der v_i - Kennlinie müssen addiert werden.

Für die Parallelschaltung gilt:

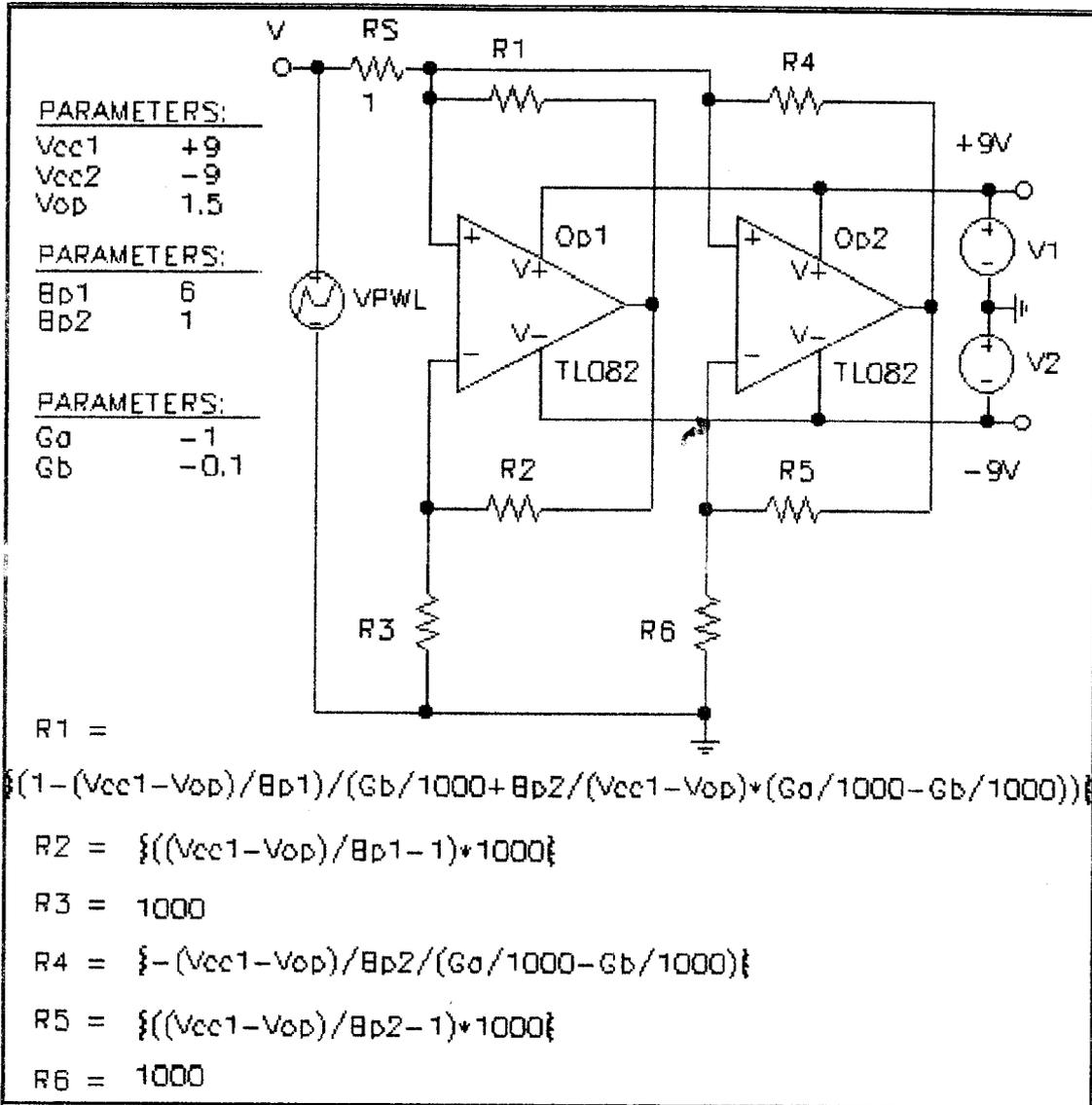
Die Ströme der v_i - Kennlinie müssen addiert werden.

Ein Beispiel zur Parallelschaltung sieht wie folgt aus:



In diesem Beispiel wurden die Betriebsspannungen der beiden OP's gleicher Art als identisch angenommen und somit sind auch die Sättigungsspannungen gleich. Wenn die Ströme der ersten Kennlinie mit denen der zweiten Kennlinie addiert werden, so erhält man eine fünfteilige stückweise lineare Funktion die den negativen Gesamtwiderstand der beiden negativen Widerstände darstellt. Die Breakpoints der beiden Kennlinien bilden die Berechnungsgrundlage für das Verhalten des Gesamtwiderstandes. Um einen möglichst großen Spielraum für diesen Widerstand zu erhalten, ist es günstig den Breakpoint Bp1 so dicht wie möglich an die Sättigungsspannung der OP's zu legen. Um das mathematische Modell zu vereinfachen sollte man den Breakpoint Bp2 auf 1V festlegen.

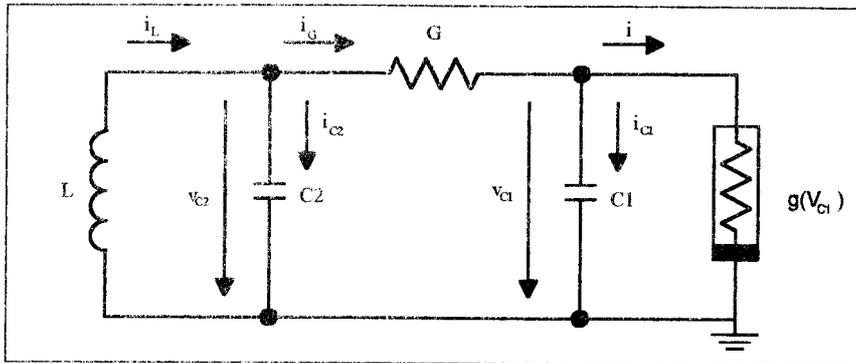
Für Chua's Circuit sind nur die Steigungen G_a und G_b interessant. Um diese beiden Steigungen beliebig und unabhängig voneinander einzustellen, wurde folgende Schaltung entwickelt:



Die 6 Widerstände werden für jede unterschiedliche Parametereinstellung neu berechnet. Dadurch wird die gewünschte Kennlinie erzeugt. In dieser Schaltung dürfen nur symmetrische Betriebsspannungen verwendet werden. Die Sättigungsspannung des TL082 ist immer um 1,5V niedriger als dessen Betriebsspannung. Der Parameter V_{op} ermöglicht es diesen Wert auch auf andere Operationsverstärker anzupassen.

III. Chua's Circuit

Chua's Circuit ist eine Schaltung in der chaotisches Verhalten auftreten kann. Die Schaltung besteht aus einer Spule, zwei Kondensatoren einen linearen und einen nichtlinearen Widerstand.



Die Schaltung kann durch folgende drei Differentialgleichungen beschrieben werden:

$$\frac{dv_{C1}}{dt} = \frac{1}{C1} * (i_L - i_{C2} - i) = \frac{1}{C1} * [G * (v_{C2} - v_{C1}) - g(v_{C1})]$$

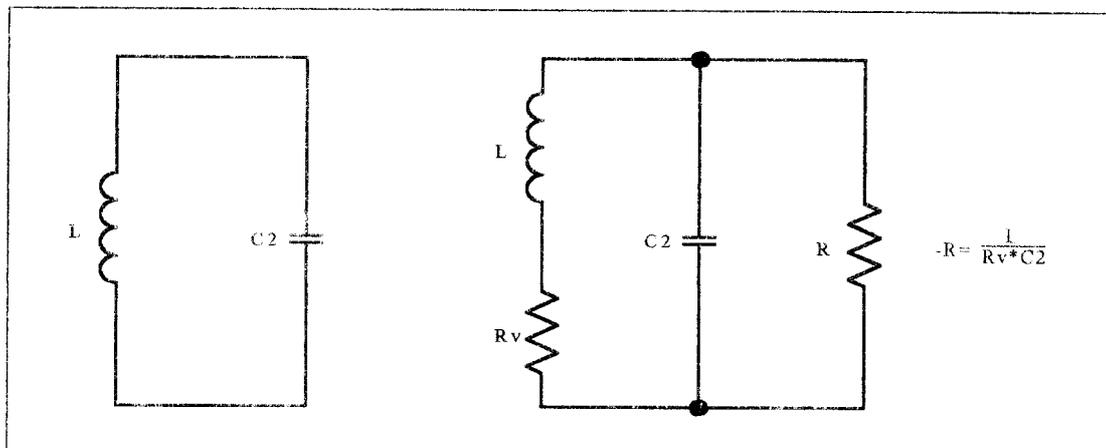
$$\frac{dv_{C2}}{dt} = \frac{1}{C2} * (i_L - i_{C1} - i) = \frac{1}{C1} * [G * (v_{C1} - v_{C2}) + i_L]$$

$$\frac{di}{dt} = -\frac{v_{C2}}{L}$$

Das Verhalten des nichtlinearen Widerstandes kann mit folgender Funktion beschrieben werden:

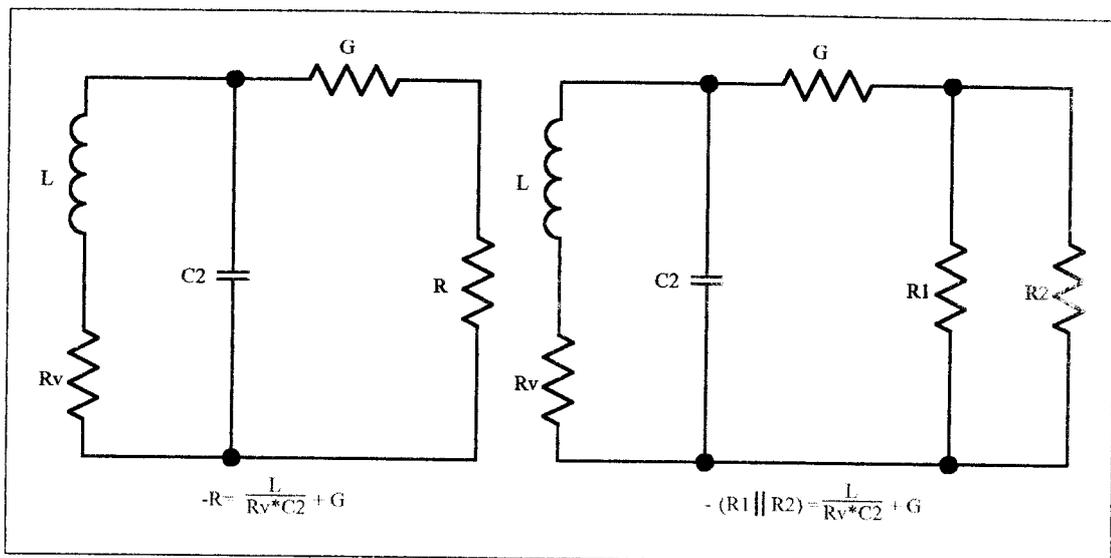
$$g(v_{C1}) = i(v_{C1}) = Gb * v_{C1} + \frac{1}{2} * (Ga - Gb) * (|v_{C1} + Bp| - |v_{C1} - Bp|)$$

Die Schaltung beginnt mit einem Parallelschwingkreis.

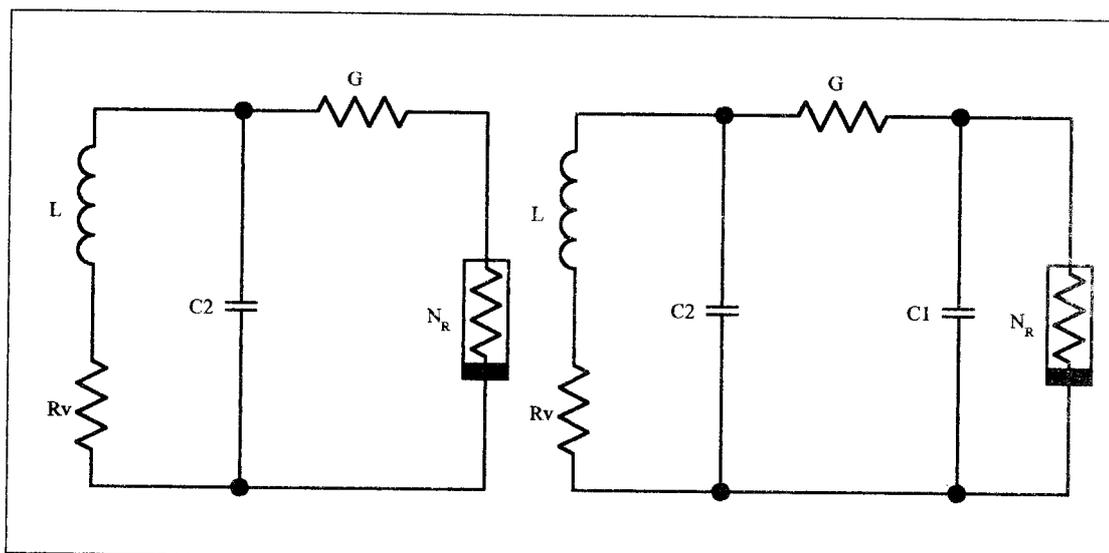


Da es aber keinen idealen undgedämpften Schwingkreis gibt, muß dieser in einen reellen Schwingkreis umgewandelt werden. Der Widerstand R_v ist der Schwingkreisverlustwiderstand. Der parallelgeschaltete Widerstand R verbraucht die Energie des Schwingkreises solange er einen positiven Wert besitzt. In dem Moment wo R negativ wird, wird Energie in den Schwingkreis eingespeist. Entspricht die eingespeiste Energie der verbrauchten Energie so hat man einen idealen Schwingkreis erzeugt.

Diese Schaltung läßt sich wie folgt verändern:



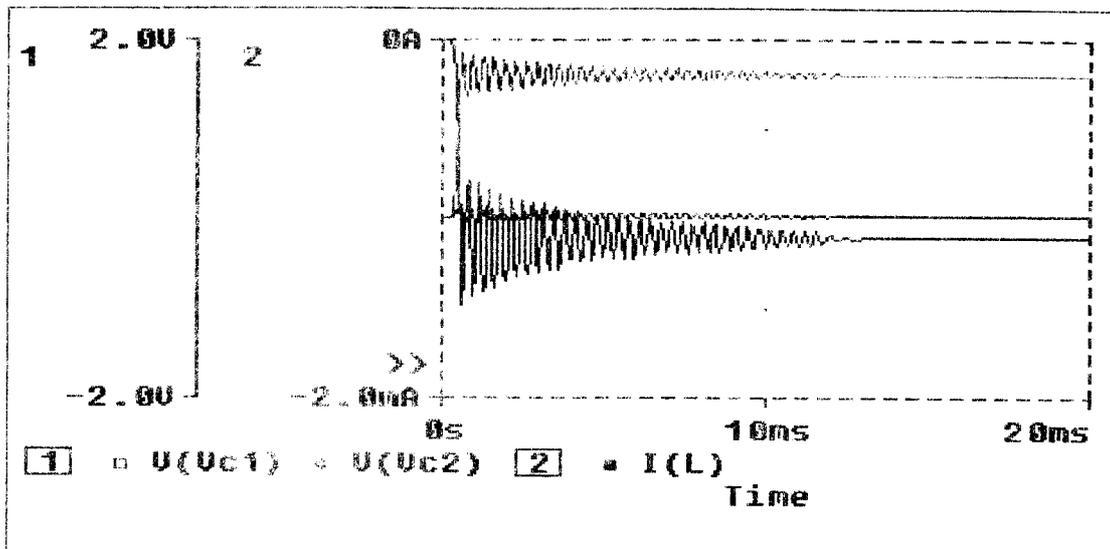
Wenn man nun für R_1 und R_2 einen negativen Widerstand einsetzt so erhält man den zuvor beschriebenen nichtlinearen Widerstand.



Die Schaltung verhält sich nach wie vor wie ein idealer Parallelschwingkreis. Erst wenn man parallel zum nichtlinearen Widerstand einen Kondensator hinzufügt kann ein chaotischen Verhalten entstehen.

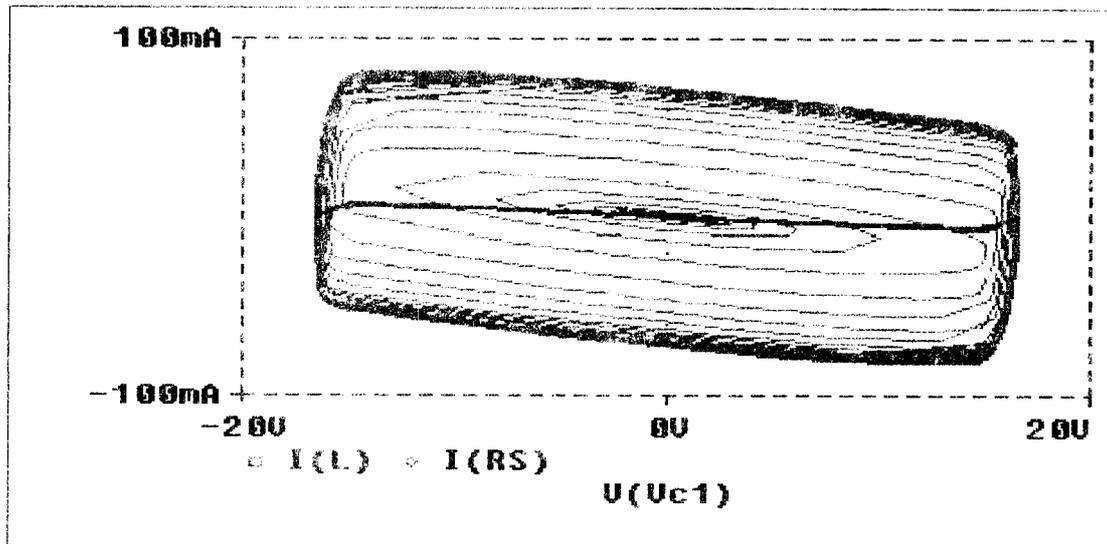
Chua's Circuit kennt zwei Grundzustände.

Dämpfung:



Die eingespeiste Energie reicht nicht aus um eine Schwingung aufrecht zu erhalten.

Sättigung:



Es wird zuviel Energie eingespeist. Die Schaltung geht in die Strombegrenzung.

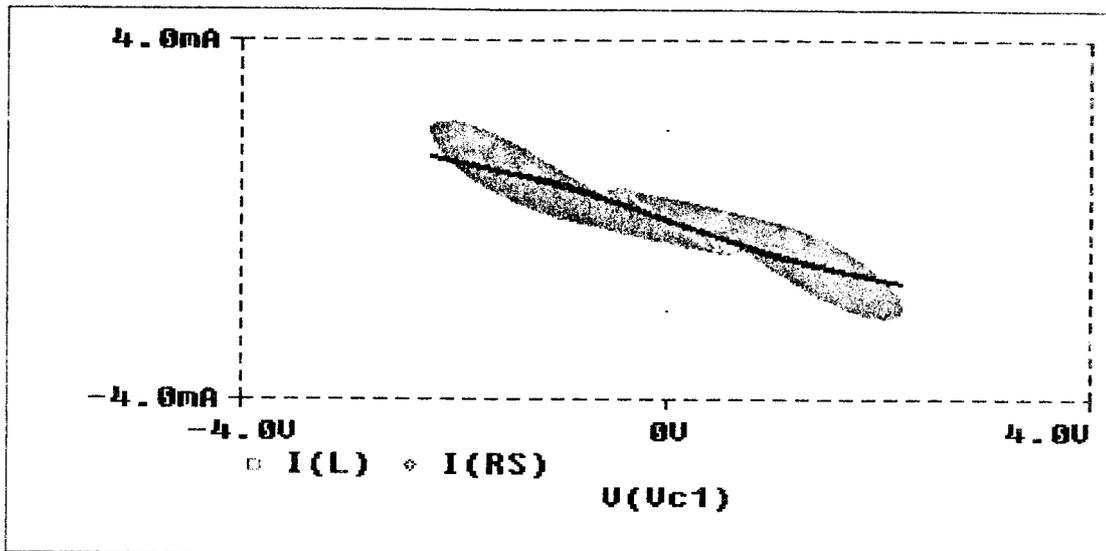
Zwischen diesen beiden Grundzuständen kann ein chaotisches Verhalten der Schaltung auftreten.

So entsteht z.B. ein Double Scroll Attraktor für die Werte:

$C_2=1\text{ F}$, $C_1=1/9\text{ F}$, $L=1/7\text{ H}$, $G=0.7\text{ S}$, $G_a=-0.8\text{ S}$, $G_b=-0.5\text{ S}$

Diese Werte müssen in der reellen Schaltung allerdings auf die Spannungs- und Strombegrenzung des Operationsverstärkers umdimensioniert werden.

Wenn dies getan wurde erhält man folgenden Attraktor:



Ein Attraktor ist die graphische Darstellung des Bereiches der Zustände in denen sich die Schaltung befinden kann. Dieser Bereich ist vergleichbar mit den Orbitalen von Atommodellen. Man kann vorhersagen, daß ein Zustand der Schaltung sich in diesem Bereich befindet. Aber zu welchem Zeitpunkt welcher Zustand angenommen wird ist nicht vorhersagbar.

Der Double Scroll Attraktor ist nur einer von vielen Attraktoren die in Chua's Circuit entstehen können. Die Attraktoren selbst wiederum bilden von ihrer Entstehung aus der Dämpfung bis hin zum Übergang in die Sättigung ein Feld, welches charakteristisch für Chua's Circuit ist. Es ist gleichgültig welchen Parameter man in der Schaltung verändert, die Veränderung bewirkt immer den selben Durchlauf dieses Feldes der Attraktoren.

Um einen gewünschten Attraktor zu erzeugen ist nützlich wenn man den Mittelpunkt um den sich die Kreisbahnen eines Attraktors bewegen berechnet.

Für die beiden Mittelpunkte des Attraktors gilt:

$$P1(X, Y) = \left(\left(\frac{|G_a| - |G_b|}{|G| - |G_b|} \right) * Bp2, -|G| * \left(\frac{|G_a| - |G_b|}{|G| - |G_b|} \right) * Bp2 \right)$$

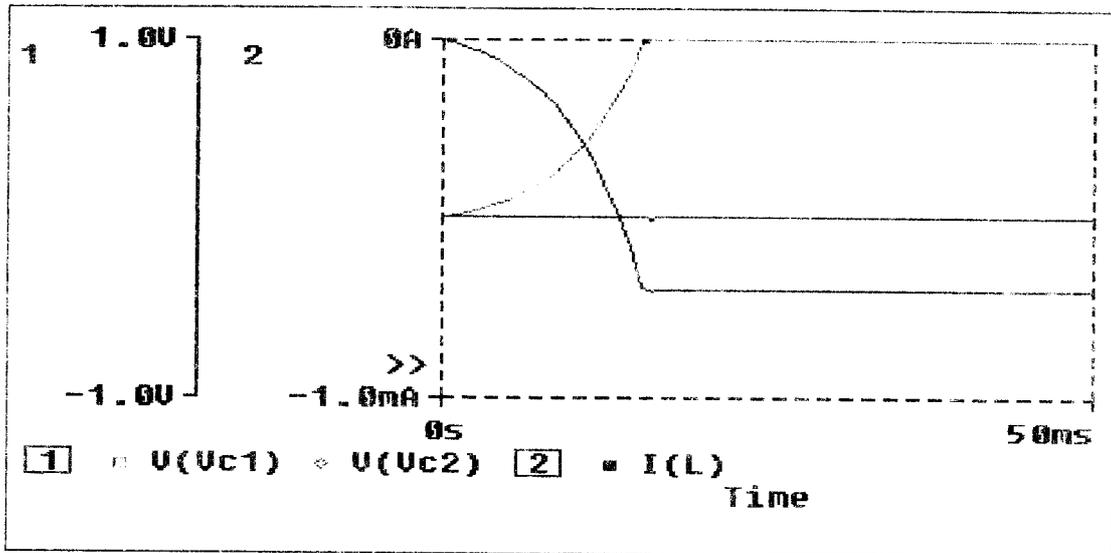
$$P2(X, Y) = \left(-\left(\frac{|G_a| - |G_b|}{|G| - |G_b|} \right) * Bp2, |G| * \left(\frac{|G_a| - |G_b|}{|G| - |G_b|} \right) * Bp2 \right)$$

In dieser Formel erkennt man zwei Sonderfälle.

Den Fall $G_a = G$ mit:

$$P1(X, Y) = (Bp2, -|G| * Bp2)$$

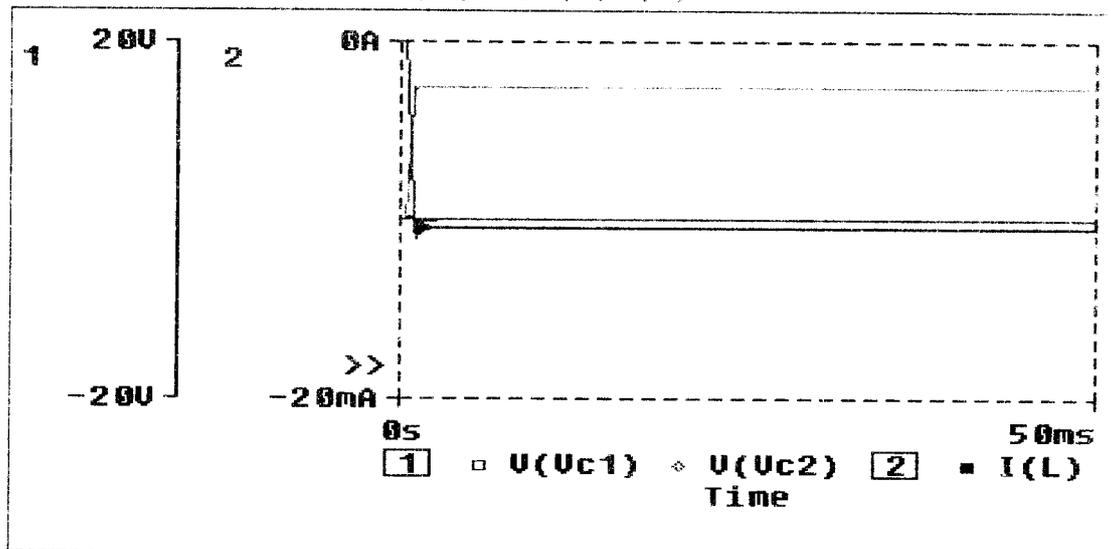
$$P2(X, Y) = (-Bp2, |G| * Bp2)$$



Den Fall $G_b = G$ mit:

$$P1(X, Y) = (\infty, \infty) \Rightarrow (Bp1, -|G| * Bp1)$$

$$P2(X, Y) = (\infty, \infty) \Rightarrow (-Bp1, |G| * Bp1)$$



Zusammenfassung

Chaos in elektrischen Schaltungen ist möglich. Je komplexer eine Schaltung wird desto wahrscheinlicher ist es, daß es einen Fall gibt in dem sich die Schaltung chaotisch verhält. Erst die Entwicklung von schnellen Rechnern als auch des Simulationsprogrammes SPICE machten es überhaupt möglich dieses Verhalten zu untersuchen.

Es gibt zwei Gründe warum es notwendig ist den Entstehungsvorgang von Chaos aus der Ordnung zu kennen. Erstens ist man mit diesem Wissen in der Lage chaotisches Verhalten in Schaltungen zu vermeiden. Zweitens werden einem völlig neue Möglichkeiten zur Lösung eines Problems gegeben.

Da es aber bisher keine allgemeingültige chaotische Systemtheorie gibt ist man gezwungen jeweils nur Einzellösungen für Spezialfälle zu finden. Wenn allerdings jemals genug Einzellösungen zusammengetragen werden und man in der Lage ist eine allgemeingültige Lösungsbeschreibung zu formulieren, so wäre die vorhandene Systemtheorie überholt.

Realisierung der Schaltung

Der nichtlineare Widerstand wird mit Hilfe von zwei Operationsverstärkern erzeugt. Da ein Operationsverstärker keine ideale spannungsgesteuerte Spannungsquelle ist, gelten folgende zwei Einschränkungen für die v_i -Kennlinie:

- Nur im Bereich zwischen -15V und +15V
- Nur im Bereich zwischen -20mA und +20mA

Um diese Einschränkungen zu erfüllen müssen die experimentell ermittelten Werte für Chua's Circuit umdimensioniert werden.

Die experimentellen Werte für den Breakpoint bei 1V lauten:

$$C1 = \frac{1}{9} \text{ F (Farad)} \quad G = 0.7 \text{ S (Siemens)}$$

$$C2 = 1 \text{ F (Farad)} \quad G_a = -0.8 \text{ S (Siemens)}$$

$$L = \frac{1}{7} \text{ H (Henry)} \quad G_b = -0.5 \text{ S (Siemens)}$$

Die Einheit des Stromes in diesen Gleichungen ist A. Um die benötigten mA zu erzeugen, müssen die Werte um den Faktor 1000 umdimensioniert werden. Dies bedeutet für die Bauelemente:

- Widerstand $= R * 10^3$
- Leitwert $= G * 10^{-3}$
- Kapazität $= C * 10^{-3}$
- Induktivität $= L * 10^3$

Da es einfacher ist Kapazitäten im Bereich von nF zu verwenden, muß eine weitere Umdimensionierung vorgenommen werden. Diese Umdimensionierung betrifft nur die zeitabhängigen Energiespeicherelemente, d.h. die Widerstände bzw. Leitwerte bleiben unverändert. Es wird der Faktor $2 * 10^{-4}$ verwendet. Dies bedeutet für die Bauelemente:

- Kapazität $= C * 2 * 10^{-4}$
- Induktivität $= L * 2 * 10^{-4}$

Nach dieser Umdimensionierung sehen die Werte für Chua's Circuit wie folgt aus:

$$C1 = \frac{1}{18} * 10^{-7} \text{ F (Farad)} \quad G = 0.7 * 10^{-3} \text{ S (Siemens)}$$

$$C2 = \frac{1}{2} * 10^{-7} \text{ F (Farad)} \quad G_a = -0.8 * 10^{-3} \text{ S (Siemens)}$$

$$L = \frac{1}{14} * 10^{-1} \text{ H (Henry)} \quad G_b = -0.5 * 10^{-3} \text{ S (Siemens)}$$

Um aber eine Schaltung aufzubauen ist es einfacher handelsübliche Bauelemente zu verwenden. Es wurde gewählt:

$$C1 = 10\text{nF} \quad C2 = 100\text{nF} \quad L = 18\text{mH} \quad G = 1800\Omega$$

Der nichtlineare Widerstand wurde festgelegt mit den Werten:

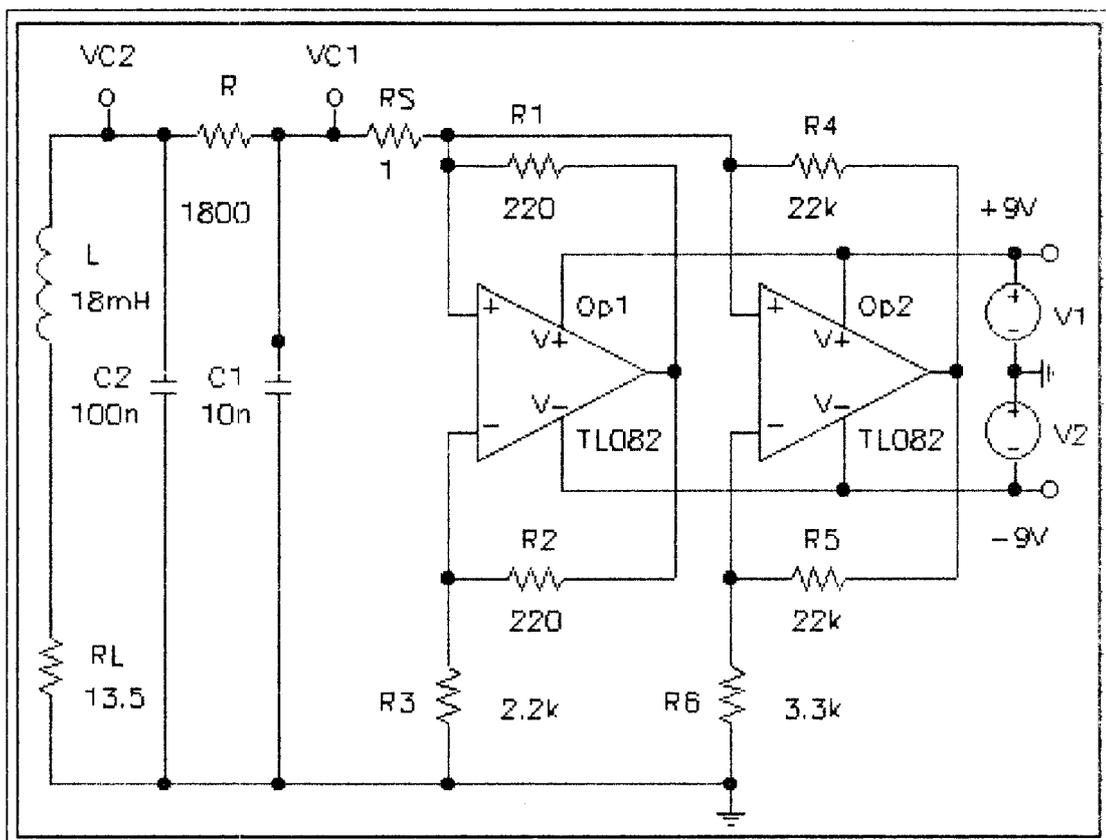
$$R1 = 220\Omega \quad R2 = 220\Omega \quad R3 = 2200\Omega$$

$$R4 = 22000\Omega \quad R5 = 22000\Omega \quad R6 = 3300\Omega$$

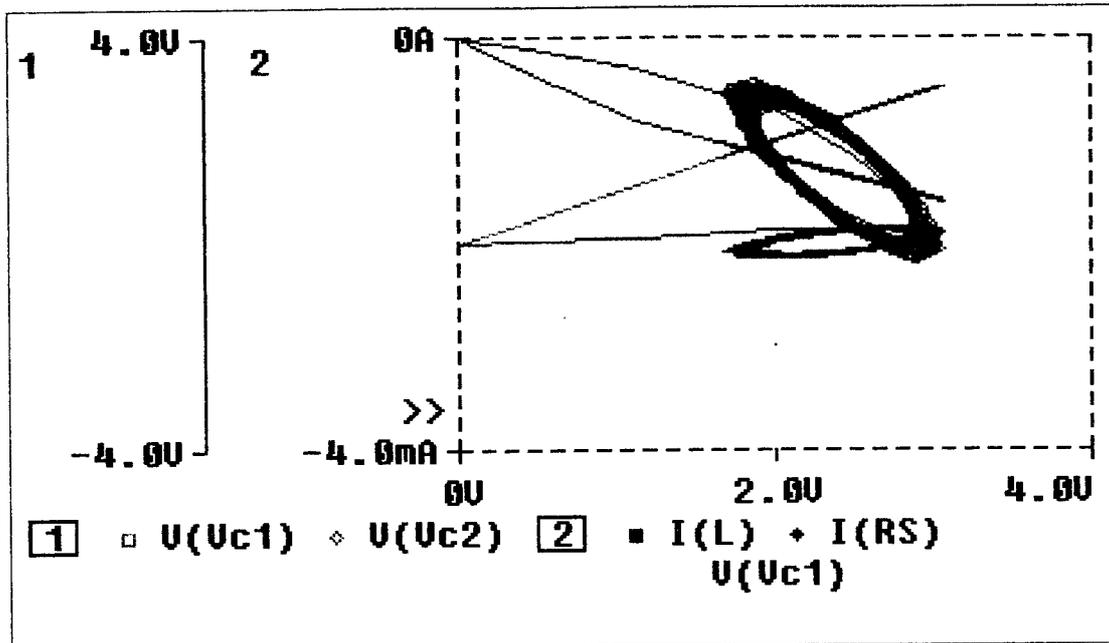
Dies entspricht den Werten:

$$G_a = -0.757 \text{ mS} \quad G_b = -0.409 \text{ mS} \quad B_p = \pm 0.97 \text{ V}$$

Dies führt zu folgendem Schaltbild:



Mit dieser Schaltung wurde nun folgendes Bild erzeugt:



VC1 auf die VC1 - Achse abgebildet ergibt eine Gerade die bei auftreten der Double Scroll als Spiegel - Achse wirkt.

VC2 auf die VC1 - Achse abgebildet zeigt die momentane Schwingung an.

$I(L)$ auf die VC1 - Achse abgebildet stellt den dazugehörigen Strom dar.

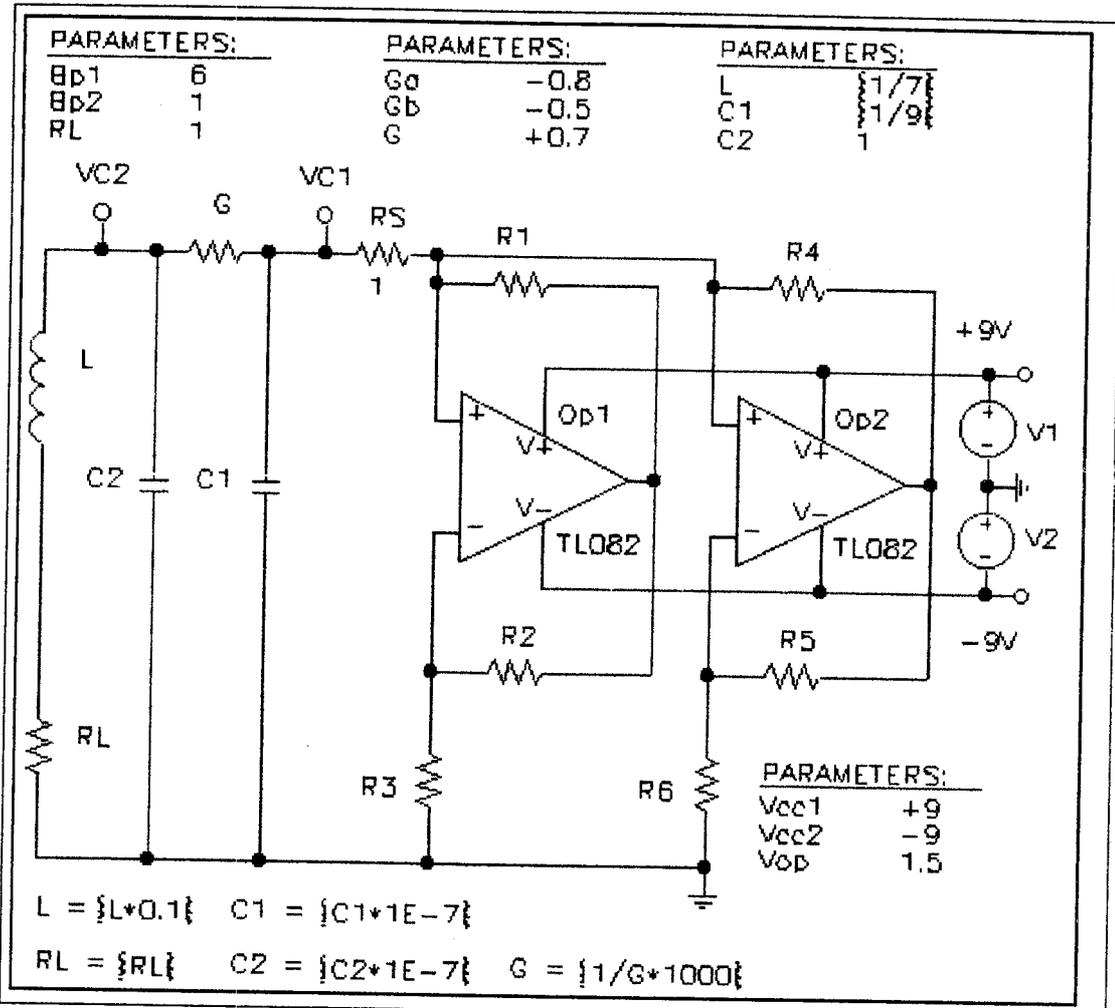
Der Strom $I(RS)$ des Shunts RS erzeugt auf die VC1 - Achse abgebildet das Strom - Spannungsverhalten des nichtlinearen Widerstandes.

Wichtiger Hinweis für Benutzer der PSPICE Demo Version 5.3 !!!

- Im Setup-Menü muß Use Initial Conditions angekreuzt werden.
- Wer UIC nicht verwenden will kann die Schaltung zum schwingen bringen, indem er den Einschaltimpuls der Spannungsquelle z.B. mit Element S (Voltage - Controlled Switch) simuliert. In diesem Fall sollten zusätzliche Abblockkondensatoren an den Eingängen der OP's verwendet werden.
- Die Verwendung des Elements Bubble für die Label VC1 bzw. VC2 ist zu empfehlen um den Absturz des Programms Probe zu vermeiden.

Simulation von Chua's Circuit

Um mit jeweils nur einem Parameter simulieren zu können wurde die folgende Schaltung entwickelt.



$$R1 =$$

$$\left\{ \left(1 - \frac{V_{cc1} - V_{op}}{B_{p1}} \right) / \left(\frac{G_b}{1000} + \frac{B_{p2}}{V_{cc1} - V_{op}} + \left(\frac{G_a}{1000} - \frac{G_b}{1000} \right) \right) \right\}$$

$$R2 = \left\{ \left(\frac{V_{cc1} - V_{op}}{B_{p1}} - 1 \right) + 1000 \right\}$$

$$R3 = 1000$$

$$R4 = \left\{ - \frac{V_{cc1} - V_{op}}{B_{p2}} / \left(\frac{G_a}{1000} - \frac{G_b}{1000} \right) \right\}$$

$$R5 = \left\{ \left(\frac{V_{cc1} - V_{op}}{B_{p2}} - 1 \right) + 1000 \right\}$$

$$R6 = 1000$$

ERSTELLEN VON HARDMAKROS UND AUFBAU EINER ZELLBIBLIOTHEK UNTER VERWENDUNG DES ES2 - LIBRARY - KITS

Detlef Pouw

ASIC Design Center, Fachhochschule Offenburg
Leitung: Prof.Dr.-Ing. Dirk Jansen

Es wird eine Anleitung zur Erstellung von Hardmakros mit der Mentor Graphics - Software gegeben. Die Hardmakros werden mit Standardzellen aus der ES2-Bibliothek der Firma EUROCHIP aufgebaut. Die Hardmakros werden in eine eigenständige Bibliothek abgelegt und können in neuen Chip-Designs verwendet werden.

1. EINLEITUNG

An der Fachhochschule Offenburg wurde im Jahr 1989 unter Leitung von Professor Jansen das ASIC - Design - Center mit 5 APOLLO Workstations und Software der Firma MENTOR GRAPHICS [MG] eingerichtet.

Es wurden (im Zusammenhang mit dem Wahlfach ASIC) ein elektronischer Würfel und als Diplomarbeit der Empfangsteil eines GPS-Empfängers (Global Position System) entwickelt und in Standardzellen - Layout [SZ] hergestellt.

Die ASIC-Entwürfe beschränkten sich bis dato auf zwei Techniken. Zum einen das normale SZ-Layout, in dem Zellen aus der ES2-Bibliothek plaziert und geroutet werden, zum anderen ein Layout, das sich aus Softmakros zusammensetzt.

Soft-Makros werden aus ES2-Zellen aufgebaut und in das Gesamt-Layout integriert. Der Nachteil dieser Entwurfstechnik liegt darin, daß Softmakros sich beim Integrieren in größere Designs, d.h. das Makro wird bis auf die Standardzellen hinunter geflattet und diese beliebig im Design plaziert und geroutet.

Somit fallen Softmakros in jedem Design unterschiedlich aus. Dies impliziert jedoch bei jedem

Design neue physikalische Eigenschaften (Timing) des verwendeten Softmakros.

Um diese negativen Eigenschaften zu vermeiden, besteht mit der MG-Software die Möglichkeit, hierarchisch aufgebaute Hardmakros zu entwerfen, die einmal richtig ausgetestet und in einer Bibliothek abgelegt werden.

Beim Routen des Chip-Design muß das Hardmakro nur an seinen Ein/Ausgangsport verdrahtet werden; das Hardmakro selbst ist bereits komplett plaziert und geroutet. Dies bringt neben bekanntem, definiertem Timing des Hardmakros den Vorteil kürzerer Routingzeiten mit sich.

Im folgenden wird ein Überblick über die MG-Software sowie eine kurze Anleitung zum Aufbau eines Makros gegeben.

2. DESIGN ÜBERBLICK

Um mit der **MG-Software V7.0** und dem **ES2-Kit** Hardmakros aufzubauen, müssen zuerst (einmalig) einige Konfigurationsdateien editiert werden, eigene Signal- und Powerports erstellt, und in eine Bibliothek integriert werden.

Dann ist es möglich, eigene Designs mit dem schematischen Editor zu erstellen und dieses automatisch in eine Hardmakrozelle verwandeln zu lassen. Eine Optimierung ist interaktiv während jedes Designs - Schritts möglich.

Schließlich können die neuen Zellen in eine Bibliothek integriert werden.

Alle Programme des ES2-Kits folgen der Syntax
es2_programm-name design.

Nach der Eingabe eines Schaltplanes mit dem schematischen Editor **NETED** muß zunächst mit dem Symboleditor **SYMED** ein Symbol für diesen Schaltplan entworfen werden.

Abbildung 1 zeigt die Ausgabedateien für das schematische Design.

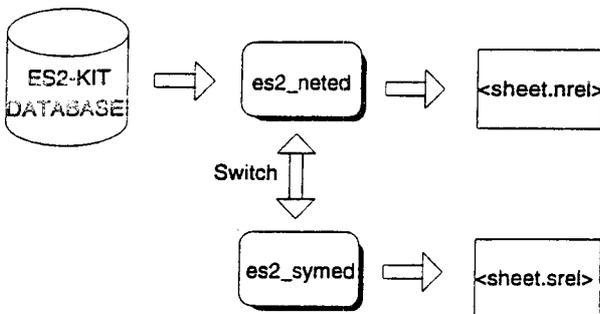


Abb. 1 NETED und SYMED schreiben ihre Daten in die Dateien <sheet.nrel> bzw. <sheet.srel>

Da NETED und SYMED die Möglichkeit besitzen, hierarchische Designs zu erstellen, müssen die schematischen Daten vor Erstellung einer Netzliste geflattet werden.



Abb. 2 ES2_COMP_EXPAND schreibt Daten nach <comp.ereI>

ES2_EXPAND_COMP sorgt dafür, daß alle Properties, die für die Netzliste benötigt werden, expandiert und in die Datenbasis <comp.ereI> integriert werden (Abb. 2).

ES2_LOGIC_ENTRY erstellt die physikalische Daten Basis aus dem expandierten logischen Design-File <comp.ereI>.

Die erstellten Dateien <design.xxx> werden von den Standardzellen-Layout-Programmen CELLGRAPH und CHIPGRAPH benötigt.



Abb. 3 ES2_LOGIC_ENTRY wandelt die logischen, expandierten Daten eines Design in physikalische Daten um.

In Abb. 4 ist das Zusammenspiel der drei Programme Floorplan-Generator, Platzierer und Router dargestellt.

ES2_CELLFLOOR generiert automatisch einen Floorplan für das angegebene Design. Daten werden nach <floor.chp> und <floor.dsn> geschrieben.

ES2_CELLPLACE platziert automatisch sämtliche Makros des Designs. Es benutzt und erneuert Daten des physikalischen Design Files <design.dsn>, generiert von LOGIC_ENTRY. Daten werden nach <place.chp> und <place.dsn> geschrieben.

ES2_CELLROUTE routet das Design. Alle Makros müssen zuvor gültig platziert worden sein. Daten werden nach <route.chp> und <route.dsn> geschrieben.

Diese drei Programme lassen sich über die Parameterdatei <cadi.prm> (editiert von <edit_parms>) steuern.

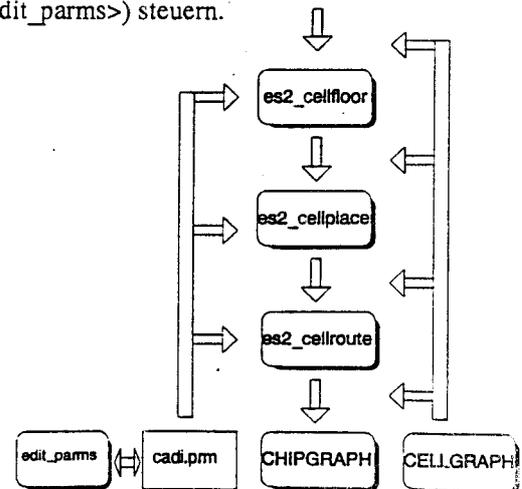


Abb. 4 CELLFLOOR, CELLPLACE und CELLROUTE beziehen ihre Parameter aus der Datei <cadi.prm>. CELLGRAPH kann in jedem Design-Schritt aufgerufen werden.

Um Zellen in eine Bibliothek aufnehmen zu können, muß zunächst entweder **ES2_CHIPGRAPH** oder **ES2_CHIPGRAPH_OUTPUT** aufgerufen werden, um die CHIPGRAPH Cell Data File-Base zu schreiben.

Ein Technologie-Verzeichnis (beim ES2-Kit ist eines `.../es2/ecpd15/sc_layout_data`) besteht mindestens aus drei Dateien:

- <cadi.blc> Binärdatei, die die physikalischen Makro- Daten enthält; erzeugt von <build_lib>
- <cadi.chp> Chip-Technologie-Datei; erzeugt von <tdf_chip_input>
- <cadi.prm> ASCII-Parameterdatei in Technology Definition Format (TDF); wird mit <edit_parms> editiert

TDF_CHIP_INPUT compiliert die Chip-Technologie-Datei <chip.tdf> unter Verwendung von Parametern aus <cadi.prm> und schreibt seine Daten nach <cadi.chp>.

BUILD_LIB nimmt die CHIPGRAPH-Daten unter Verwendung der Daten aus <cadi.chp> in die Bibliotheksdatei <cadi.blc> auf.

Abbildung 5 zeigt die Verbindung der Cell Data Base mit den Programmen TDF_CHIP_INPUT und BUILD_LIB.

Eine Zusammenfassung der wichtigsten Abläufe zur Generierung eines Hardmakros zeigt Abb. 6.

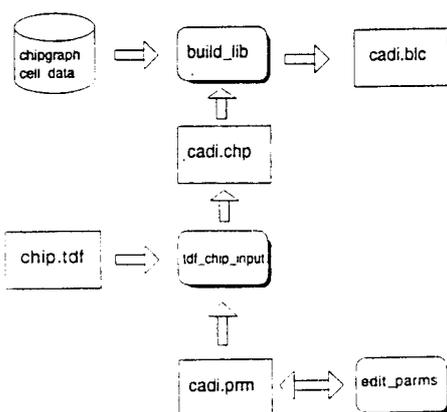


Abb. 5 Die Technologiedateien, die Bibliotheksdatei <cadi.blc> und ihre Erzeugung durch die Programme <build_lib> und <tdf_chip_input>

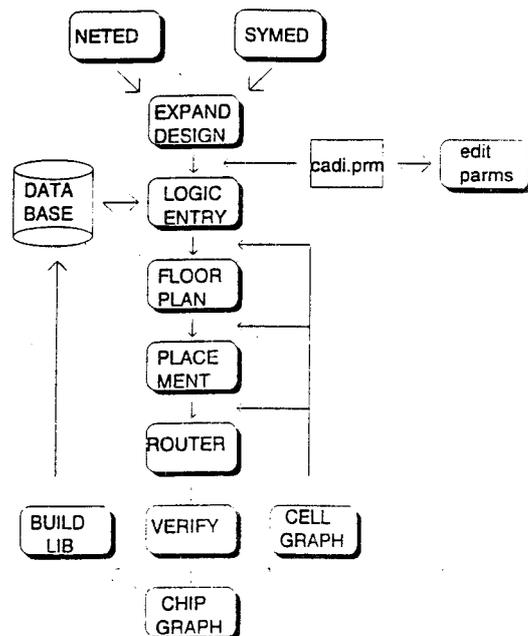


Abb. 6 Ablaufplan (mit interaktivem CELLGRAPH) der wichtigsten MG-Programme zur Erzeugung eines Hardmakros

3. POWER - UND SIGNAL -PORTS ERSTELLEN

Bevor eigene Zellen erstellt werden können, müssen Signal-I/O-Ports und Powerports definiert und in der Bibliothek aufgenommen werden.

Abbildung 7 und 8 zeigen die Ports (Metal2.i und Metal2.e) mit allen Properties.

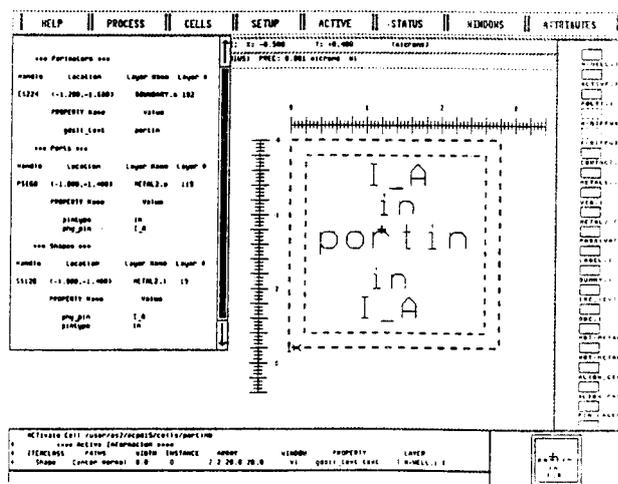


Abb. 7 Signal-Eingangsport in CHIPGRAPH mit Statusfenster für die Property-Anzeige

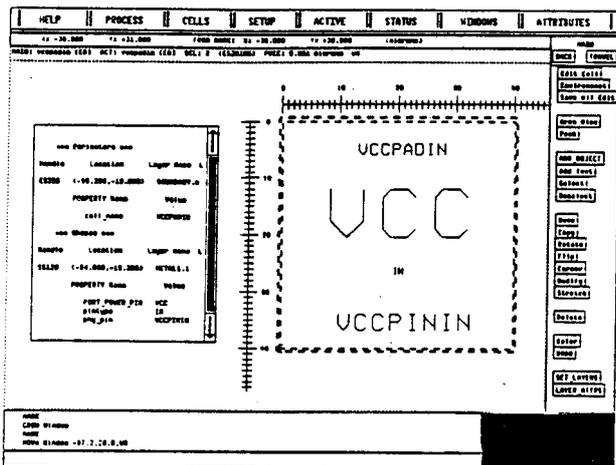


Abb. 8 VCC-Port in CHIPGRAPH mit Statusfenster für die Property-Anzeige

Die Zellen werden mit `<build_lib> sc_layout_data` in die Bibliotheksdatei `<cadi.blc>` aufgenommen.

In NETED werden Powerports aus der ES2-Zellbibliothek einfach modifiziert, d.h. es werden neue Properties vergeben (Abb. 9).

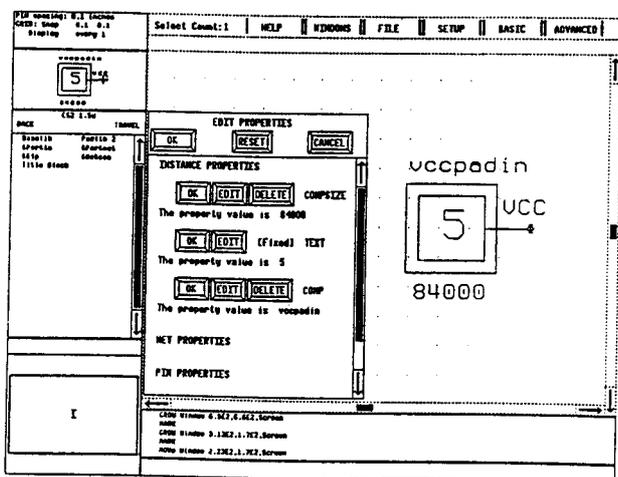


Abb. 9 NETED mit Spannungsport-Symbol und Properties

Die Sheet-Daten diesem Symbol müssen ebenfalls mit neuen Properties versehen werden (Abb. 10).

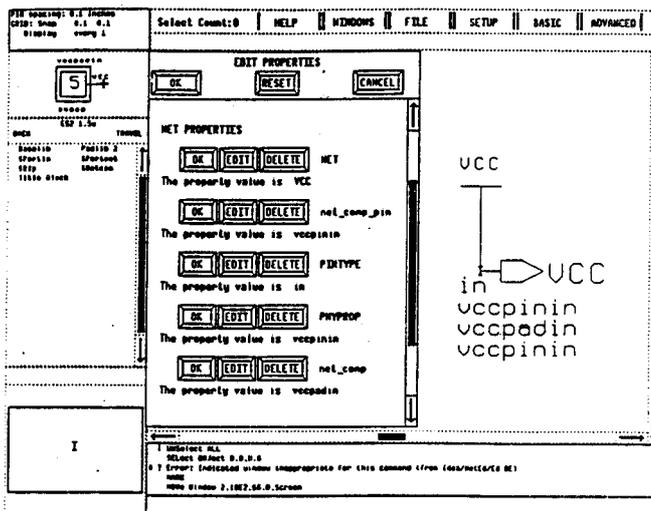


Abb. 10 NETED mit den NET-Properties für vccpadin
Auf gleiche Weise wird der VSS-Port erstellt.

Um die Verbindung vom Neted-Design zu den erstellten Ports zu schaffen, werden in NETED folgende Properties an das Design vergeben (Abb. 11):

PINTYPE IN IN für Eingangssignale

NET_COMP Portname Verbindung von NETED \$portin zum selbst erstellten Signalport

NET_COMP_PIN I_A Physikalische Pin Bezeichnung

Mit dem Programm `<es2_autocell>` können nun die Sheet-Daten in physikalische Daten gewandelt werden. Abb. 12 zeigt das optimierte Hardmakro der ALU.

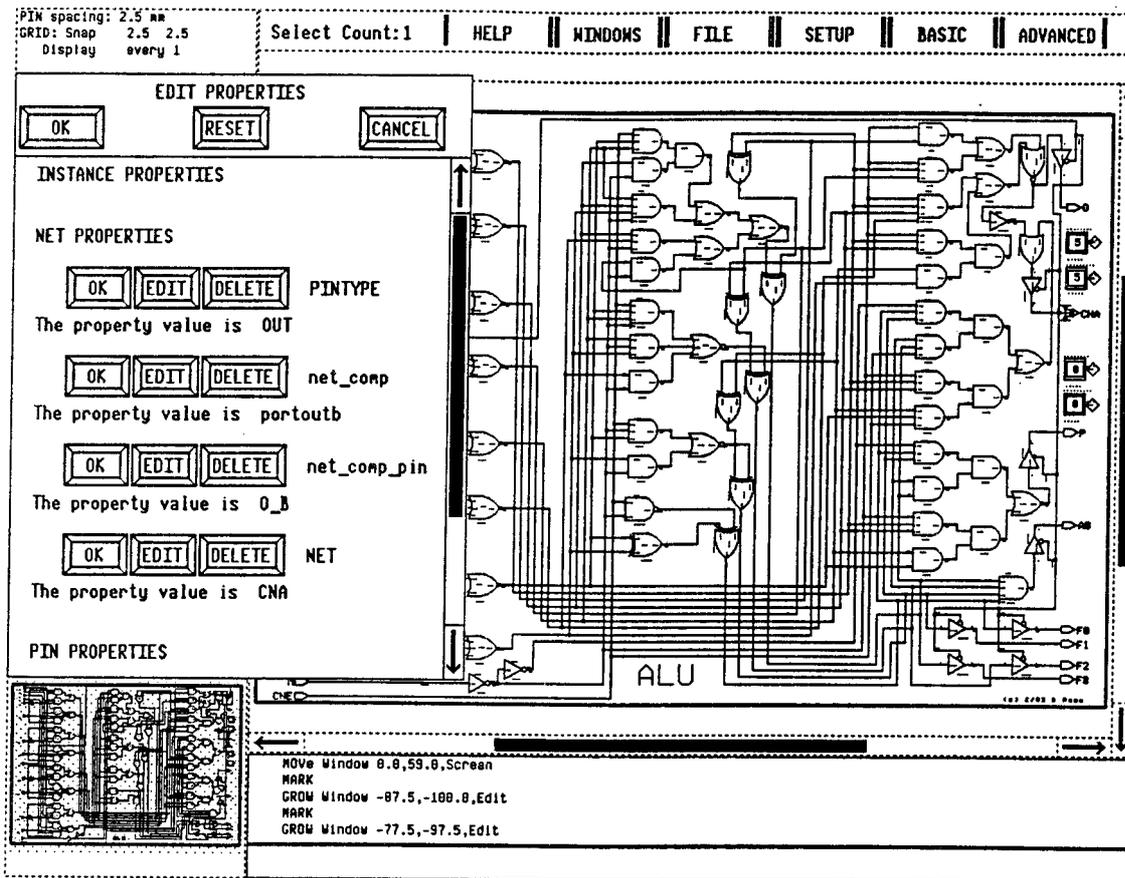


Abb. 11 NETED-SHEET einer ALU mit selektiertem Ausgangsport CNA und Fenster für Properties

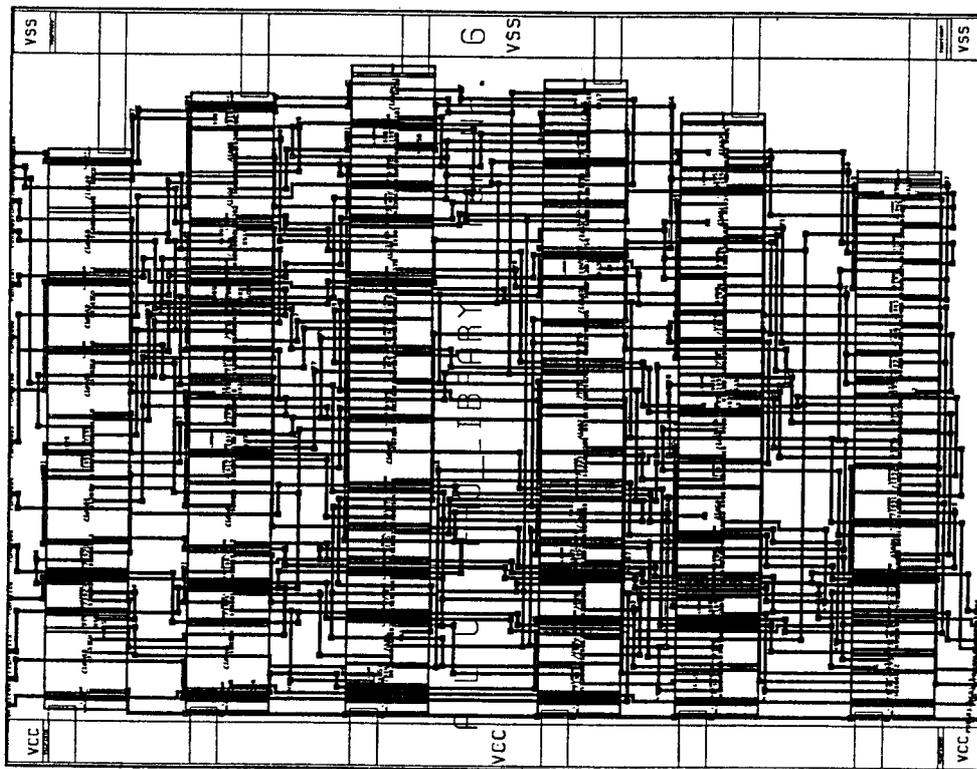


Abb. 12 HARDMAKRO der ALU in CHIPGRAPH

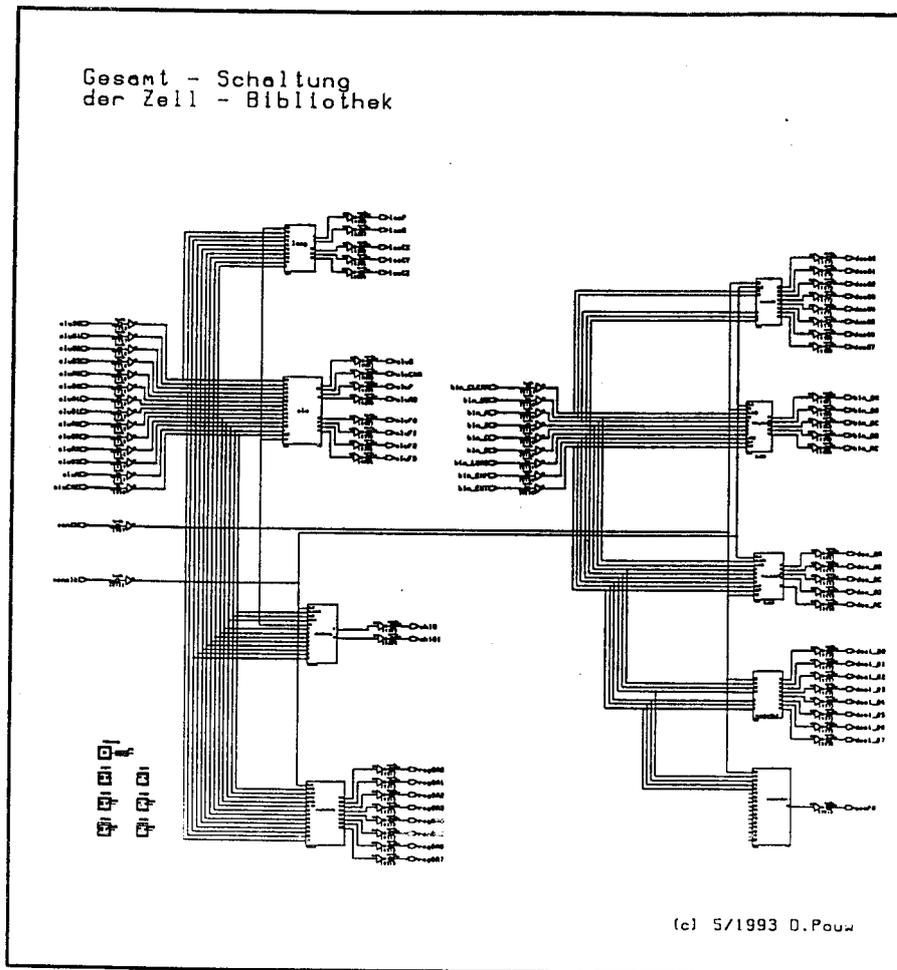


Abb. 13 Gesamtlayout des Testchips in NETED

Es wurden an der Fachhochschule Offenburg neun Hardmakros erzeugt und in eine eigene Bibliothek abgelegt. Abbildung 13 zeigt die Gesamtschaltung in NETED (um Eingangspins und damit Chipfläche zu sparen, wurden mehrer Eingänge zusammengeschaltet), Abbildung 14 zeigt den fertigen Chip. Dieser wurde gefertigt, um die Funktionalität und das Timing der Hardmakros zu testen.

4. ZUSAMMENFASSUNG

Nachdem alle Vorarbeiten (Ports erstellen und aufnehmen) in CHIPGRAPH und NETED durchgeführt sind, kann jeder neuer Schaltplan (mit den entsprechenden Properties) mittels `<es2_autocell>` in ein Hardmakro gewandelt und mit `<build_lib>` in eine Bibliothek aufgenommen werden.

Probleme treten in erster Linie beim Programm `<es2_cellplace>` auf, das die Hardmakros nicht plazieren kann. Die Plazierung muß in CELLGRAPH erfolgen mit anschließendem Routen durch das Programm `<es2_cellroute>`.

Die automatische Erstellung von Hardmakros mittels `<es2_autocell>` erbringt keine zufriedenstellenden Ergebnisse. Jedes Makro muß anschließend manuell in CHIPGRAPH optimiert werden (Flächenoptimierung).

Auch der Autorouter bringt öfter unbefriedigende Ergebnisse (Signalpfade werden als Schleifen ausgeführt, obwohl genug Platz für eine direkte Verbindung besteht usw.). So sollte man das gesamte Design auch bezüglich des Routens überprüfen und optimieren.

Das fertige Chip-Layout wird mit dem Programm `<translate>` write in das GDS2-Format übersetzt und kann via E-Mail an GMD gesendet werden. Hier wird mit Design-Rule-Checkern das Layout auf Fehler überprüft und der Chip gefertigt.

Die Version 8.0 der MENTOR GRAPHICS Software ist wesentlich anwenderfreundlich und zur Erstellung von Hardmakros besser geeignet.

Eine exakte Beschreibung der Vorgehensweise zum Erstellen von Hardmakros kann über die

Fachhochschule Ulm

bezogen werden.

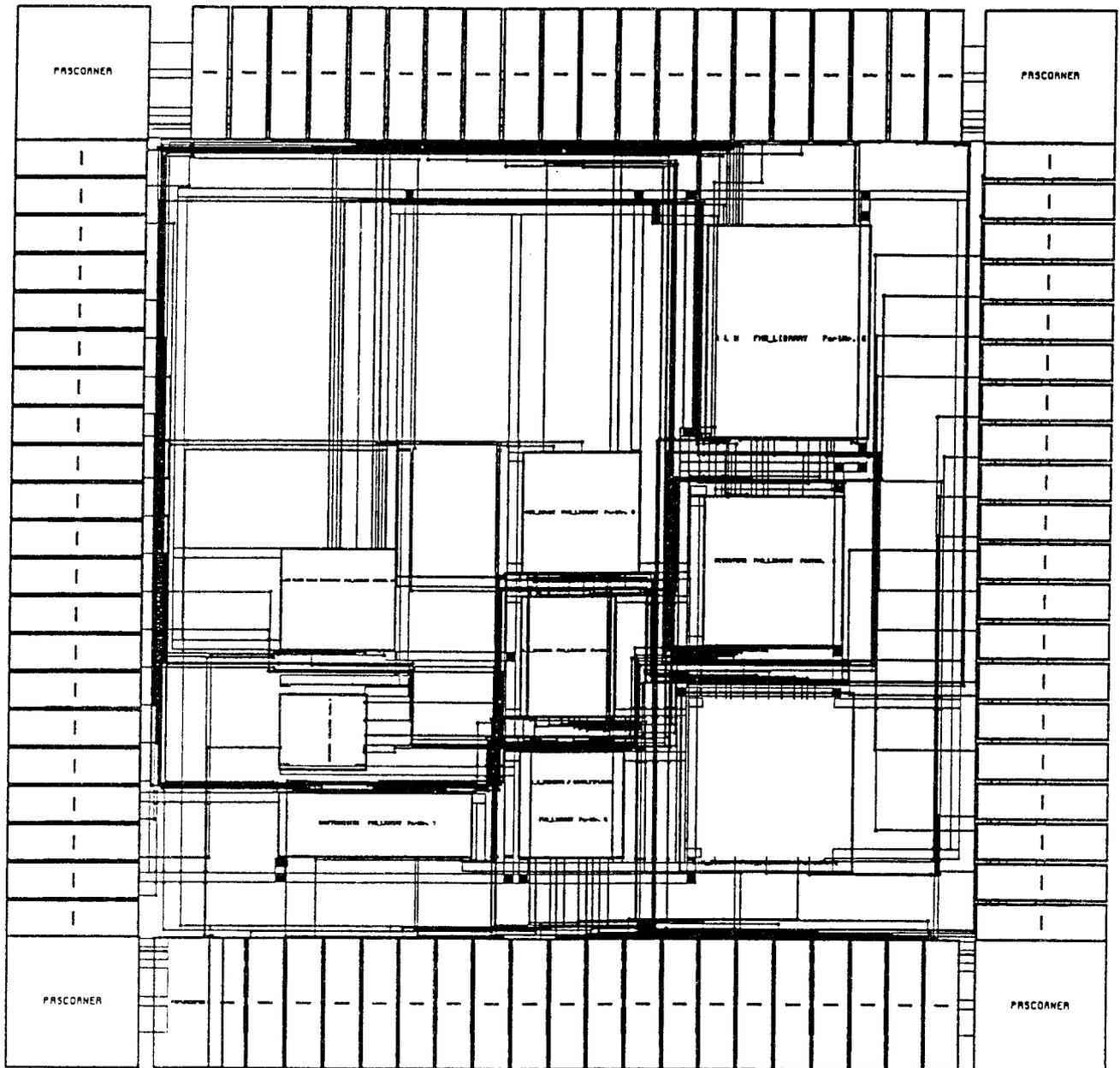


Abb. 14 Der fertige Chip in CHIPGRAPH mit allen Hardmakros

Ein Praktikum zur Schaltungsintegration

Faulhaber, Führer, Rathenow

Gefördert mit Mitteln des LARS-Programms
"Leistungsanreize für die Lehre"

Zusammenfassung:

Es wird ein Praktikum beschrieben, das Studenten der Nachrichtentechnik und Industrieelektronik mit der Arbeitsweise beim Entwurf einer integrierten Digitalschaltung vertraut macht. Ausgehend von der Schaltungsbeschreibung mit einer Beschreibungssprache wird die Integration auf mehreren programmierbaren Logikbauelementen und auf einem Gate Array durchgeführt. Die Integration auf einem Logic Cell Array wird demonstriert.

1. Zielsetzung

1.1 Allgemeines

Es sollte ein Praktikum entwickelt werden, das Studenten des Studiengangs Nachrichtentechnik/Industrieelektronik mit der Arbeitsweise beim Entwurf einer integrierten Digitalschaltung vertraut macht. Das Vorhaben ist mit Mitteln des LARS-Programms (Leistungsanreize für die Lehre) gefördert worden.

1.2 Randbedingungen

Das Praktikum findet im 7. Semester statt. Die Studenten habe im 5. Semester bereits eine vierstündige Vorlesung "Schaltungsintegration" gehört, in der Sie mit der CMOS-Schaltungstechnik, Herstellungsverfahren für Integrierte Schaltungen, mögliche Realisierungsformen bei Verwendung von Halbfabrikaten und mit Beschreibungsformen und Simulationsverfahren für Digitalschaltungen vertraut gemacht werden.

Für das Praktikum stehen 7 Termine zu vier Stunden zur Verfügung.

Das Labor ist mit 10 Mentor Graphics Workstation (HP APOLLO DN3500) mit Software für den Entwurf integrierter Schaltungen und mit ergänzender Software von anderen Herstellern ausgestattet. Ein PLD-Programmiergerät und Geräte für das Testen integrierter Schaltungen sind vorhanden.

Pro Studienrichtung und Semester müssen in der Regel 36 Studenten das Praktikum durchführen.

1.3 Lehrinhalte

Mit dem Praktikum sollen die Studenten folgende Arbeiten kennenlernen:

- Verhaltensbeschreibung mit Hilfe einer Hardwarebeschreibungssprache,
- Simulation des Schaltungsverhaltens zur Kontrolle der Verhaltensbeschreibung,
- Synthese der Schaltung für die Realisierung auf PLDs, Aufteilung der Gesamtschaltung auf mehrere Bausteine,
- Bibliotheksgestützte Synthese der Schaltung für die Integration auf einem Gate Array (GA) oder Logic Cell Array (LCA),
- Entwicklung von Teststimuli und Kontrolle der Testbarkeit durch Fehlersimulation,
- Entwurf des Schaltungslayouts mit Hilfe automatisch oder halbautomatisch arbeitender Entwurfswerkzeuge.

2. Ausfuehrung

2.1 Allgemeines

Das Praktikums kann mit der verfügbaren Stundenanzahl nur dann erfolgreich durchgeführt werden, wenn die Studenten durch anderweitige Probleme möglichst wenig von den Lehrzielen abgelenkt werden. Anderweitige Probleme resultieren aus der Bedienung der Workstation und der auf dieser verfügbaren Werkzeuge sowie aus der Funktion der zu integrierenden Schaltung. Daraus

resultiert die Forderung nach einer möglichst straffen Führung der Studenten bei der Arbeit an der Workstation und nach einer besonders einfachen Schaltung, deren Funktion jedem Studenten sofort einsichtig ist. Selbst bei einfachen Schaltungen ergibt sich ein Arbeitsaufwand, der von einem Studenten in der verfügbaren Zeit nicht erbracht werden kann. Die Arbeit muß auf ein Team von Studenten aufgeteilt werden. Dieses Team darf nicht so groß sein, daß die Koordinationsarbeit von den Lehrzielen ablenkt. Am zweckmäßigsten wird die Koordinationsarbeit von den Betreuern erbracht.

2.2 Organisation

Aus der Studentenzahl (36), der Zahl der verfügbaren Übungstermine (7), den Lehrzielen (6) und der Anzahl Workstation ergibt sich folgendes Organisationsschema:

- Ein Übungstermin wird für eine Vorbereitungsphase verwendet, an jedem der restlichen 6 Termine wird ein Lehrziel bearbeitet.
- Aufteilung der Studenten in Zweiergruppen (18).
- Zwei Zweiergruppen bilden ein Team, insgesamt ergeben sich 9 Teams.
- Aufteilung der 9 Teams in zwei Gruppen (4 und 5 Teams), die im wöchentlichen Wechsel im Labor arbeiten. Das Labor ist damit während 13 Wochen jeweils für 4 Stunden belegt. Zusätzlich muß den Studenten freies Üben ermöglicht werden. Dafür werden jede Woche zwei zusätzliche Stunden eingeplant. Von den vorhandenen Workstation bleibt mindestens eine frei. Von dieser aus können die Betreuer notfalls ohne die Arbeit der Studenten zu behindern in den Arbeitsprozeß eingreifen.

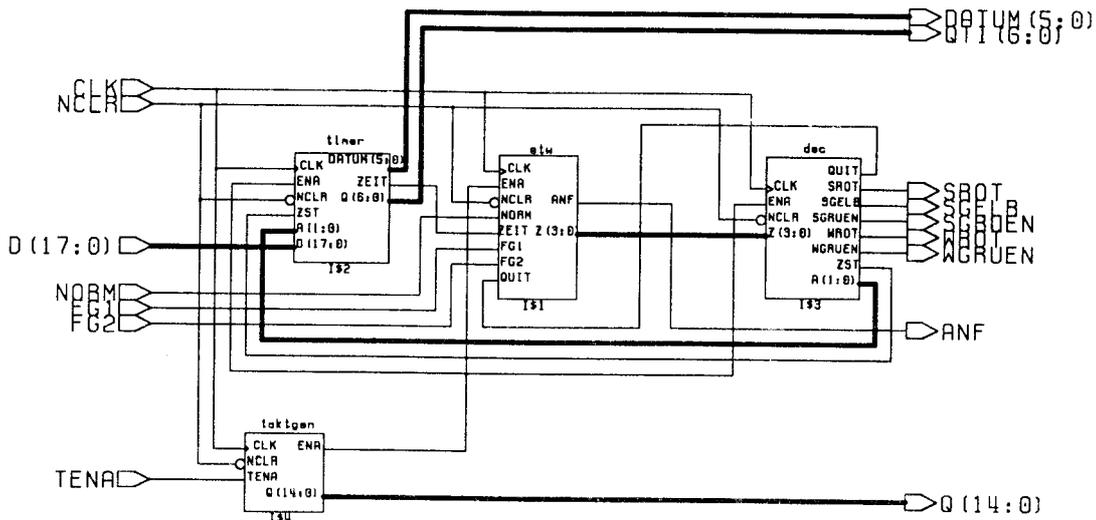
2.3 Die ausgewählte Schaltung

Als zu integrierende Schaltung ist die Steuerung für die Ampelanlage eines Fußgängerübergangs ausgewählt worden (Bild 1).

Die Schaltung ist hierarchisch aufgebaut und besteht aus vier Blöcken. Sie arbeitet voll synchron.

Der **Taktgenerator** (taktgen) erzeugt aus einem 32 kHz Takt ein Enablesignal mit der Dauer einer Taktperiode und der Frequenz von ca. 1 Hz. Das **Steuerwerk** (stw) besteht aus einem Mooreautomat für die Ablaufsteuerung und aus einem Flipflop, in dem Fußgängeranforderungen gespeichert werden. Der **Decoder** (dec) erzeugt aus den Zustandssignalen des Steuerwerks die Steuersignale für die Ampeln und speichert diese in einem Register. Ferner liefert er ein Startsignal und eine 2 bit Adresse für den Timer und ein Rücksetzsignal für den Flipflop, in dem die Fußgängeranforderungen gespeichert werden. Mit dem **Zeitgeber** (timer) wird die Dauer der Ampelphasen im Bereich 2..65 s festgelegt. Für jede Ampelphase kann mit Hilfe der 2 bit Adresse eine andere Zeit aus dem 18 bit Datenwort D(17:0) ausgewählt werden.

Die Aufteilung der Schaltung in vier Teilschaltung ist zweckmäßig, weil dadurch die Vorteile hierarchischer Schaltungsstrukturen gegenüber ungegliederten Strukturen deutlich gemacht werden können und weil dadurch die Möglichkeit besteht, daß jedem Studenten des vierköpfigen Teams die Verantwortung für einen Schaltungsteil übertragen werden kann.



1 Ampelsteuerung

2.4 Die Vorbesprechung

Bei der Vorbesprechung werden die Teams gebildet. Die Studenten erhalten das Blockschaltbild (Bild 1) und eine Funktionsbeschreibung der Gesamtschaltung, die teilweise verbale Form hat und teilweise aus Protokollen vorbereiteter Rahmendateien besteht. Ferner erhalten die Studenten die Arbeitsanleitungen für die sechs Teilaufgaben. Die Funktion der Gesamtschaltung und der vier Teilschaltungen wird erläutert. Die Lernziele der Teilaufgaben werden vorgestellt und die zu jedem Übungstermin auszuführenden Vorbereitungsarbeiten sowie der Inhalt der anzufertigenden Berichte wird durchgesprochen.

2.5 Die sechs Teilaufgaben

2.5.1 Schaltungsbeschreibung

Die Schaltungsbeschreibung erfolgt mit einer Beschreibungssprache, wobei die Sprache des Programmpakets LOG/IC oder VHDL verwendet werden kann. Für die Beschreibung stehen vorbereitete Rahmendateien in der Sprache von LOG/IC zur Verfügung, die entweder von den Studenten ergänzt werden können oder die als Basis für die Beschreibung mit VHDL dienen. Jeder Teilnehmer bereitet für die Beschreibung des Schaltungsteiles, für den er verantwortlich ist, zu Hause ein Konzept vor (z.B. das Ablaufdiagramm des endlichen Automaten).

Die beiden Teilaufgaben Schaltungsbeschreibung und Verhaltenssimulation werden zweckmäßig für jede Teilschaltung unmittelbar nacheinander ausgeführt, weil diese beiden Arbeiten miteinander eng verzahnt sind. Wird bei der Verhaltenssimulation ein fehlerhaftes Verhalten festgestellt muß häufig die Beschreibung korrigiert und erneut simuliert werden. Diese Arbeitsschritte sind u.U. mehrmals hintereinander zu durchlaufen.

Vom Studenten vorzubereiten: Konzept für die Schaltungsbeschreibung.
Inhalt des Berichts: Protokoll der Beschreibungsdateien.

2.5.2 Verhaltenssimulation

Die Verhaltenssimulation erfolgt mit dem Programm LOG/IC-VERIFY. Jeder Student bereitet für die Simulation des Schaltungsteiles, für den sein Partner in der Gruppe verantwortlich ist, vor dem Übungstermin ein Simulationskonzept vor. Dieses Konzept wird zweckmäßig mit dem Rechner erstellt (ASCII-Datei) und besteht aus einer Folge von Kommentaren die die durchzuführenden Simulationsschritte erläutern. Bei der Übungsdurchführung werden die Kommentare in die zu erstellende Stimuldatei eingefügt und sinngemäß in Stimuli für die Eingangssignale und in Kommandos zur Kontrolle der Ausgangssignale umgesetzt.

Die Verhaltenssimulation soll den Nachweis erbringen, daß die Schaltungsbeschreibung richtig durchgeführt worden ist und daß die Schaltungsfunktion vollständig verstanden worden ist. Dadurch, daß die Beschreibung und die Simulation einer Teilschaltung nicht von einem Gruppenmitglied vorbereitet werden sondern beide daran beteiligt sind, kontrollieren sich die Studenten gegenseitig und erreichen ein korrektes Ergebnis.

Vom Studenten vorzubereiten: Simulationskonzept.

Inhalt des Berichts: Protokoll der Stimuldateien

2.5.3 Realisierung mit PLDs

Die Teilschaltungen sollen mit PLDs im 20- oder 24-Pin-Gehäuse realisiert werden. Dazu ist eine Aufteilung des Zeitgebers und des Taktgenerators auf mehrere Bausteine erforderlich (d.h. jede Gruppe muß eine Teilschaltung aufteilen). Ein Konzept für diese Aufteilung ist von jeder Gruppe für den Übungstermin vorzubereiten.

Die Programmierdaten und Testvektoren für die Herstellung der Teilschaltungen werden mit dem Programm LOG/IC-PLD-COMPILER erzeugt. Die Gruppen haben zunächst freie Hand bei der Auswahl geeigneter PLDs und können verschiedene Varianten durchspielen. Das endgültige Ergebnis muß sich aber an der vorbereiteten Testschaltung orientieren, die bestimmte Gehäusegrößen und Pinbelegungen vorschreibt. Für jede Teilschaltung sind Dateien vorbereitet, in denen diese Pinbelegung beschrieben ist und die in die Schaltungsbeschreibung kopiert werden können. Außerdem muß Rücksicht auf die im Labor verfügbaren PLD-Typen genommen werden.

Am Ende des Übungstermins kann jedes Team die erforderlichen PLDs programmieren, diese in die vorbereitete Testschaltung einfügen und einen Funktionstest durchführen.

Vom Studenten vorzubereiten: Konzept für die Aufteilung einer Schaltung auf mehrere PLDs.

Inhalt des Berichts: Das vorbereitete Aufteilungskonzept, Protokoll von Teilen der Beschreibungsdateien der Teilschaltungen, Testvektoren einer Teilschaltung mit Analyse der Testvektoren für ein bestimmtes Ausgangssignal.

2.5.4 Synthese fuer die Realisierung auf einem Gate Array

Die Synthese erfolgt mit den Synthesewerkzeugen LOG/IC-GATES und LOG/IC-GATESPRO. Als Zieltechnologie dienen GateForest Gate Arrays, für Testzwecke wird ein Prüfpfad (Scanpath) eingefügt.

Eine Teilschaltung wird in mehreren Varianten synthetisiert. Mit GATES (es werden neben Flipflops lediglich NAND- und NOR-Gatter sowie Inverter und Buffer verwendet), mit GATESPRO (neben den vorgenannten Elementen können auch Komplexgatter, EXOR- und EXNOR-Gatter sowie Multiplexer verwendet werden) und mit den Optionen Flächenoptimierung sowie Laufzeitopti-

mierung. Die Ergebnisse werden hinsichtlich Flächenbedarf und Verzögerungszeit längs des kritischen Pfades miteinander verglichen. Das Ergebnis mit dem minimalen Flächenbedarf wird für die Weiterarbeit ausgewählt. Die zweite Teilschaltung, die von der Gruppe bearbeitet wird, wird lediglich mit der Option Flächenoptimierung mit GATES und GATESPRO synthetisiert. Auch bei dieser Schaltung wird die flächenoptimale Lösung weiter verwendet.

Die Stromlaufpläne der ausgewählten Lösungen werden mit einem Programm automatisch erzeugt und mit dem Mentor Stromlaufplaneditor weiter bearbeitet (es sind lediglich geringfügige Ergänzungen und Umbenennungen erforderlich). Die Symbole der Teilschaltungen sind bereits vorbereitet.

Abschließend werden die Stimulidateien der Teilschaltungen in das für QUICKSIM (den Mentor-Digitalsimulator) erforderliche Format übersetzt und die synthetisierten Schaltungen durch Simulation kontrolliert.

Vom Studenten vorzubereiten: die Beschreibung der Synthesewerkzeuge ist zu studieren.

Inhalt des Berichts: vergleichende Darstellung der Synthesergebnisse, Liniendiagramme und Protokolle der Kontrollsimulation.

2.5.5 Entwurf der Testmuster und Fehlersimulation

Für jede Gruppe eines Teams wird eine Gesamtschaltung vorbereitet, die aus den von diesem Team erzeugten Teilschaltungen aufgebaut ist. Der Stromlaufplan dieser Gesamtschaltung mit Prüfpfad ist in der Übungsanleitung enthalten.

Jede Gruppe bereitet für diese Teilaufgabe ein Testkonzept für einen Schaltungsteil vor, in dem der Prüfpfad benutzt wird. Auf der Basis dieses Testkonzepts werden bei der Durchführung dieser Teilaufgabe Teststimuli entworfen und mit Hilfe der Simulation auf Korrektheit überprüft. Abschließend wird mit diesen Teststimuli eine Fehlersimulation des ausgewählten Schaltungsteils durchgeführt und die Fehlerüberdeckung ermittelt.

Vom Studenten vorzubereiten: Testkonzept für eine Teilschaltung mit Verwendung des Prüfpfads der Gesamtschaltung.

Inhalt des Berichts: Testkonzept, Ergebnis der Fehlersimulation.

2.5.6 Erzeugung des Gate Array Layouts

Die Erzeugung des Layouts erfolgt weitgehend automatisch. Lediglich die Signal-Pad-Zuordnung muß von den Studenten durchgeführt werden. Der fertige Layout wird am Ende der Übung mit dem Mentor Layouteditor CHIPGRAPH angesehen. Als Vorbereitung auf diese Teilaufgabe müssen die Studenten aus der GateForest Gate Array Familie den passenden Typ aussuchen und sich mit den Arbeitsschritten bei der Erstellung des Layouts (die bei der Ausführung automatisch ablaufen) vertraut machen.

Die Studenten haben bei dieser Aufgabe nur wenige Arbeiten selbst zu erledigen. Es bleibt darum Zeit für die Demonstration des Funktionstests mit der vorbereiteten Gate Array Lösung und für die Demonstration der LCA Lösung. Die LCA Schaltung wird bibliotheksgestützt synthetisiert. Dadurch werden die Logikblöcke des LCAs nicht optimal genutzt. Auf diesen Mangel und auf Möglichkeiten diesen Mangel zu beseitigen werden die Studenten hingewiesen.

Vom Studenten vorzubereiten: Der passende Gate Array Typ, die Beschreibung der Arbeitsschritte bei der Erstellung des Layouts ist zu studieren, die Struktur der Logikblöcke eines LCAs ist zu studieren.

Inhalt des Berichts: Signal-Pad-Zuordnung, Protokoll der Layouterstellung, Beispiel für die Zusammenfassung von Funktionen in einen LCA-Logikblock.

3. Erfahrungen und Ausblick

3.1 Durchfuehrung des Praktikums

Das Praktikum ist im Sommersemester 1993 erstmals durchgefuehrt worden. Dabei konnten die Teilaufgaben 1..3 in der vorgesehenen Zeit bearbeitet werden. Die Teilaufgaben 4 und 5 haben statt der geplanten zwei Übungstermine die restlichen drei beansprucht. Die Teilaufgabe 6 konnte nicht mehr bearbeitet werden. Dafu"r sind wahrscheinlich folgende Gr"unde ma"ßgeblich:

- Die Bedienung der Mentorwerkzeuge QUICKSIM und QUICKFAULT ist in den Ubungsanleitungen noch nicht ausreichend gut beschrieben.
- Der Zeitaufwand fu"r die Nachbearbeitung des synthetisierten Stromlaufplans ist zu hoch.
- Der Zeitaufwand fu"r die Nachbearbeitung der in das QUICKSIM-Format ubersetzten Simulationsstimuli ist zu hoch.
- Die Studenten sind auf die Benutzung des Pr"ufpfads nicht ausreichend gut vorbereitet worden.
- Die Studenten haben sich ungen"ugend auf diese Teilaufgaben vorbereitet.

Diese M"angel sollen durch folgende Ma"nahmen bis zum Wintersemester 1993/94 behoben werden:

- Die Ubungsanleitungen werden ubearbeitet und verbessert.
- Fu"r die Nachbearbeitung des Stromlaufplans wird ein Macro vorbereitet, das diese Arbeit gro"ßteils automatisch durchfu"hrt.
- Das Programm fu"r die Ubersetzung der Simulationsstimuli wird verbessert, soda"ß eine Nachbearbeitung entfallen kann.
- Die Benutzung des Pr"ufpfads wird in einem ausfu"hrlicheren Beispiel als bisher, das in die Ubungsanleitung mit aufgenommen wird, gezeigt.
- Die Teilaufgaben 4 und 5 werden in der Vorbesprechung ausfu"hrlicher behandelt als bisher.
- Die Studenten erhalten fu"r die Vorbereitung der Ubungen Merkbla"tter in Form von Fragebogen, die zu Beginn der Ubung ausgefu"llt vorzulegen sind.

Durch diese Ma"nahmen wird es hoffentlich gelingen, die Teilaufgaben 4 und 5 von u"berfl"ussigem Ballast zu befreien und den Studenten die M"oglichkeit zu bieten sich auf die wesentlichen Lernziele zu konzentrieren.

Schlechte Erfahrungen sind auch mit der bisherigen Form der Berichte gemacht worden. Die Berichte waren in der Regel zu umfangreich und bestanden lediglich aus Rechnerprotokollen, die unreflektiert zusammengestellt und abgegeben worden waren. Dem soll k"unftig dadurch begegnet werden, da"ß noch genauer angegeben wird, was der Inhalt des Berichts sein soll, und da"ß die Forderung erhoben wird, da"ß der Inhalt der in den Bericht aufgenommenen Rechnerprotokolle jeweils mit einem Text erl"autert werden mu"ß.

3.1.1 Weiterentwicklung des Praktikums

Die gegenw"artige Form des Praktikums ist auf die Verwendung der Version 7.x der Mentor Graphics Software abgestimmt. Die Fachhochschulen Baden-W"urttembergs haben aber mittlerweile die Version 8.x der Software geliefert bekommen und sind dabei, ihre Lehrmaterialien auf diese neue Software umzustellen. Diese Umstellung mu"ß auch bei diesem Praktikum erfolgen. Da die

Version 8.x eine grafische Benutzeroberfläche besitzt und sich dadurch von der Version 7.x grundlegend unterscheidet, ist eine vollständige Neubearbeitung der Übungsanleitungen notwendig. Die Version 8.x macht darüberhinaus neue Programme verfügbar, die beim Praktikum künftig genutzt werden sollen.

Die Schaltungsbeschreibung soll künftig ausschließlich mit VHDL erfolgen. Auch für die Verhaltenssimulation soll VHDL als Stimuli-Beschreibungssprache verwendet werden. Dadurch ist künftig eine Übersetzung der Stimuli aus dem LOG/IC-Format in das QUICKSIM-Format nicht mehr erforderlich.

Die LOG/IC-Werkzeuge GATES und GATESPRO sollen durch das Mentor-Werkzeug AUTOLOGIC und durch SYNOPSIS ersetzt werden.

Die Teilaufgabe 6 muß künftig entfallen, weil die Version 8.x die Erzeugung von Gate Array Layouts bisher nicht unterstützt, oder es muß auf die Erzeugung einer Standardzellenschaltung übergegangen werden.

4. Laborausstattung und Hilfsmittel

4.1 Erforderliche Laborausstattung

Für die Durchführung des Praktikums ist folgende Ausstattung der Arbeitsplätze erforderlich:

- HP-APOLLO Workstation mit AEGIS Betriebssystem.
- Mentor Graphics Softwarepakete IDEA Station, GATE Station, AUTOGATE, QUICKFAULT und CHIPGRAPH.
- LOG/IC mit den Optionen VERIFY, PLD-Database, GATES, GATESPRO und HINT (bei Beschreibung mit VHDL, nur auf PC).
- GateForest Mentor Design-Kit Version 1.1, Release 2.0
- PLD-Programmiergerät

4.2 Verfügbare Hilfsmittel

Das Labor Schaltungsintegration der Fachhochschule Ulm stellt anderen Hochschulen auf Wunsch folgende Hilfsmittel für die Durchführung des Praktikums zur Verfügung:

- schriftliche Praktikumsanleitung für die Studenten,
- schriftliche Anleitung für die zur Vorbereitung der Teilaufgaben notwendigen Arbeiten für den Betreuer des Praktikums,
- vorbereitete Datensätze,
- Lösungen der Teilaufgaben und Programmierdaten für das LCA,
- das Programm VERI_TO_QUICK für die Übersetzung der Simulationsstimuli,
- das Programm TRANS für die Erzeugung der seriellen Prüfmuster für den Prüfpfad aus hexadezimal codierten parallelen Datenworten.
- eine Testschaltung für den Funktionstest mit Sockeln für PLDs, das Gate Array und das LCA,
- Schaltungsmuster auf Gate Array.

