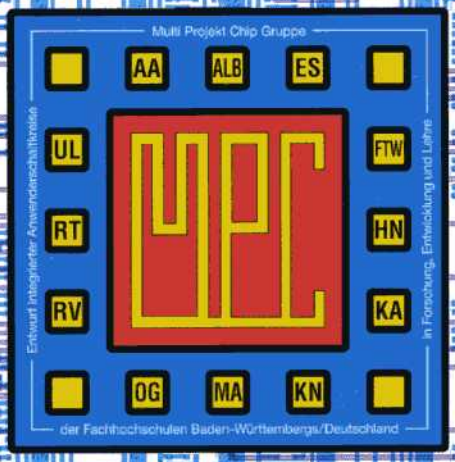
MULTIPROJEKT CHIP-GRUPPE

BADEN - WÜRTTEMBERG

Workshop Januar 2000

Mannheim



MULTIPROJEKT CHIP-GRUPPE

BADEN - WÜRTTEMBERG

Workshop Juli 2000

Ulm

Herausgeber: Fachhochschule Ulm

Adressen der

MULTIPROJEKT-CHIP-GRUPPE (MPC-Gruppe)

BADEN - WÜRTTEMBERG

http://www.mpc.belwue.de

Fachhochschule Aalen
Prof. Dr. Kohlhammer, Postfach 1728, 73428 Aalen
Tel.: 07361/576-296, Fax: -324, Email: bernd.kohlhammer@fh-aalen.de

Fachhochschule Albstadt-Sigmaringen
Prof. Dr. Rieger, Johannesstr. 3, 72458 Albstadt-Ebingen
Tel.: 07431/579-124, Fax: -149, Email: rieger@fh-albsig.de

Fachhochschule Esslingen
Prof. Dr. Kampe, Flandernstr. 101, 73732 Esslingen
Tel.: 0711/397-4221, Fax: -4212, Email: gerald.kampe@fht-esslingen.de

Fachhochschule Furtwangen
Prof. Dr. Rülling, Postfach 28, 78113 Furtwangen
Tel.: 07723/920-503, Fax: -610, Email: ruelling@fh-furtwangen.de

Fachhochschule Heilbronn
Prof. Dr. Clauss, Max-Planck-Str. 39, 74081 Heilbronn
Tel.: 07131/504-400, Fax: /252-470, Email: clauss@fh-heilbronn.de

Fachhochschule Karlsruhe
Prof. Ritzert, Postfach 2440, 76012 Karlsruhe
Tel.: 0721/925-1512, Fax: -1513, Email: ritzert@fh-karlsruhe.de

Fachhochschule Konstanz
Prof. Dr. Voland, Brauneggerstraße 55, 78462 Konstanz
Tel.: 07531/206-644, Fax: -559, Email: voland@fh-konstanz.de

Fachhochschule Mannheim
Prof. Dr. Albert, Speyerer Str. 4, 68136 Mannheim
Tel.: 0621/2926-351, Fax: -454, Email: g.albert@fh-mannheim.de

Fachhochschule Offenburg
Prof. Dr. Jansen, Badstr. 24, 77652 Offenburg
Tel.: 0781/205-267, Fax: -242, Email: d.jansen@fh-offenburg.de

Fachhochschule Pforzheim
Prof. Dr. Kesel, Tiefenbronner Str. 65, 75175 Pforzheim
Tel.: 07321/28-6567, Fax: -6060, Email: kesel@fh-pforzheim.de

Fachhochschule Ravensburg-Weingarten
Prof. Dr. Klotzbücher, Postfach 1261, 88241 Weingarten
Tel.: 0751/501-630, Fax: /49240, Email: klotzbuecher@fbe.fh-weingarten.de

Fachhochschule Reutlingen
Prof. Dr. Kreutzer, Federnseestr. 4, 72764 Reutlingen
Tel.: 07121/341-108, Fax: -100, Email: hans.kreutzer@fh-reutlingen.de

Fachhochschule Ulm
Prof. Führer, Postfach 3860, 89028 Ulm
Tel.: 0731/50-28338, Fax: -28363, Email: fuehrer@fh-ulm.de

Inhaltsverzeichnis

Workshop-Vorträge

Weiterer Beitrag

Low-Noise Amplifiers using SiGe HBT Technology

U. Erben
TEMIC Semiconductor GmbH,
Lise–Meitner–Strasse 15, 89081 Ulm/Germany
e–mail: uwe.erben@ulm.temic–semi.de

Abstract

A commercially available SiGe HBT MMIC technology is used to realize low noise amplifiers covering a wide frequency band. Minimum noise figures of 1 dB, 1.6 dB and 3.3 dB are observed in 1.9, 5.7 and 10 GHz LNAs, respectively. The gain of all amplifiers is in excess of 12 dB. The large signal performance is investigated by two–tone intercept point measurements, where the IIP3 was found to be better than -13 dBm in all amplifiers.

1 Introduction

The communication market is one of the fastest growing markets today. The introduction of new services into this market demands low–cost devices. Silicon–Germanium heterojunction bipolar transistor (SiGe HBT) technology can combine the maturity of a Silicon process technology with excellent performance in the high frequency range. Especially, the application of a hole–blocking base–emitter heterojunction concept to the SiGe transistor can offer small base resistances in conjunction with reasonably high current gain. Both parameters are necessary pre–requisites for low–noise operation of bipolar transistors [1, 2]. Looking at Friis's law [3], the noise figure F of a receiver front end is equal to

$$F = F_1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2}, \qquad (1)$$

where F and G are the noise figure and the gain of the low–noise amplifier (1), the image filter (2) and the mixer (3), respectively. As can be seen from equation 1, the noise figure of the RF system is mainly determined by the noise figure of the first amplifier. The SiGe HBT has been shown to offer a noise performance suitable for all but the most demanding system requirements, up to 10 GHz.

The topic of this paper is the application of monolothically integrated SiGe HBTs in this technology to a variety of low–noise amplifiers (LNAs) for several frequencies, 1.9, 5.7 and 10 GHz.

2 Transistor technology and performance

The SiGe HBT monolithic microwave integrated circuit (MMIC) process developed at Temic features a constant Germanium concentration of 20 % throughout the base providing for an effective hole–blocking barrier at the base–emitter interface. The process starts on a 20 Ωcm substrate with conventional buried layer and channel stop. The collector region is formed by a 700 nm CVD silicon deposition and is separated by a subsequent recessed local oxidation. After implantation of the collector and substrate contacts the final transistor layer structure is deposited with a CVD process. A constant Germanium content and a boron doping concentration of $4 \times 10^{19}\ cm^{-3}$ are used in the base. The growth is mono–crystalline in the oxide windows and poly–crystalline on the silicon oxide. In order to reduce the lead and contact resistance in emitter and base, titanium silicide is formed by a salicide

process. The fabrication process finishes with an oxide passivation and a two–level Al metallization. The process is described in more detail in [4].

There exist two types of SiGe HBTs , either with low

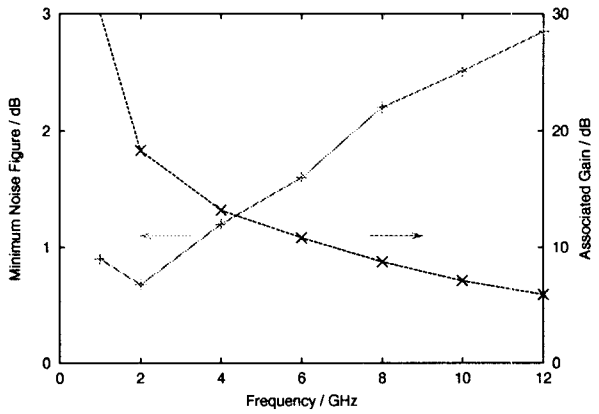


Figure 1: Noise performance of a 2x1.2x20 μm^2 SiGe HBT at a bias of 1.8 V and 2 mA

doped or selectively high doped collectors. Depending on the collector design, the peak transit frequency is 35 and 50 GHz, respectively. Both types of HBTs exhibit a maximum frequency of oscillation above 50 GHz. Furthermore, the technology comprises several resistive layers and a MIM capacitance with 1.1 fF/cm^{-3}. Inductors with quality factors up to 10 are also available. Figure 1 summarizes the typical noise performance of a two–finger SiGe HBT. At 2 GHz a minimum noise figure of 0.7 dB and an associated gain of 18 dB were observed. For a frequency of 10 GHz the minimum noise figure increases to 2.5 dB while the associated gain decreases to 7.3 dB.

3 Circuit design

Because passive matching networks are lossy and degrade the noise performance of the amplifier [3], active matching techniques are used here. Using inductive feedback in the emitter of the first–stage transistor and proper scaling, the input bondwire inductance is sufficient to achieve input match. For a frequency of 1.9 GHz a transistor with a 6 emitter finger with $1.2 \times 20\ \mu m^2$ each is used. The gain

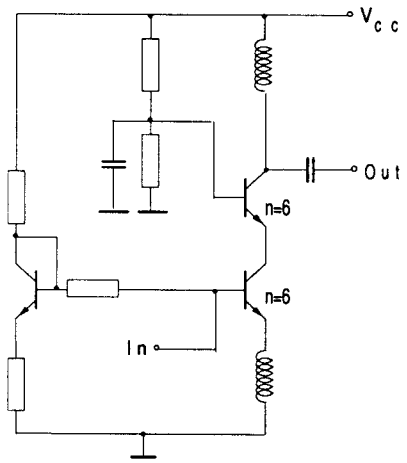


Figure 2: Circuit diagram of a 1.9 GHz SiGe HBT LNA

degradation due to the emitter feedback technique is compensated by using a cascode topology, as depicted in figure 2. The broadband perfomance of this concept enables the application of this type of amplifier in the frequency range of 1 to 2.5 GHz. The noise optimum input match will be achieved with varying input bond wire length.

For further gain enhancement of the 5.7 GHz amplifier, a second stage is added. The transistor size is reduced to

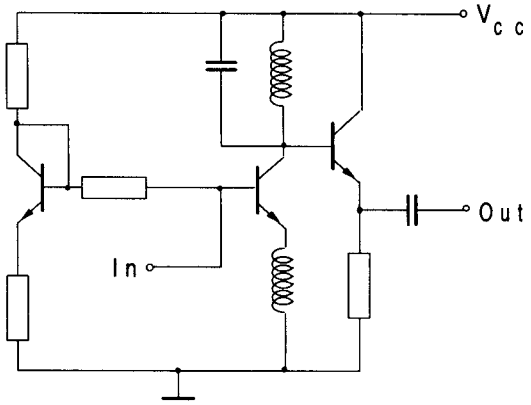


Figure 3: Circuit diagram of a 5.7 GHz SiGe HBT LNA

$2 \times 1.2 \times 20\ \mu m^2$. Inductors are used as reactive loads for the first stage. Due to the small series resistance they contribute only little thermal noise to the amplifier. The reactive load of the first stage is simultaneously used as a matching network to achieve a noise optimum source re-

flection coefficient for the second stage. The second stage is in common–collector configuration in order to omit the coupling capacitor between the stages. Furthermore, the bias current in the second stage is mainly determined by the large–signal performance of the circuits.

For the 10 GHz amplifier a two–finger transistor is used, which compromizes between noise optimum input match and minimum noise figure. For gain enhancement in the

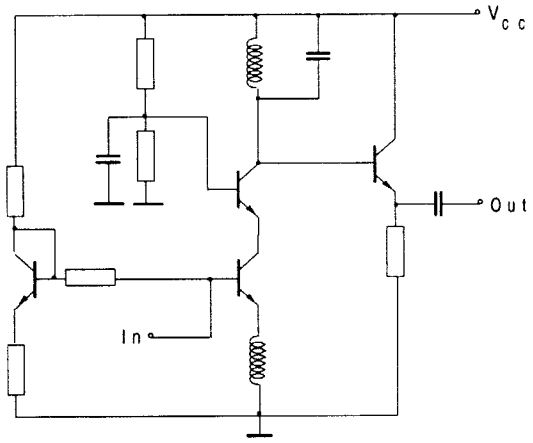
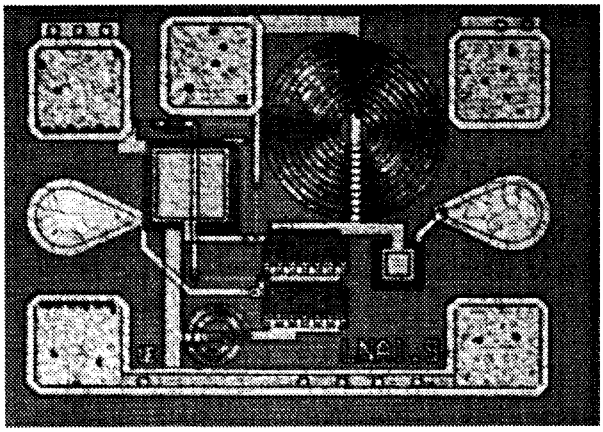


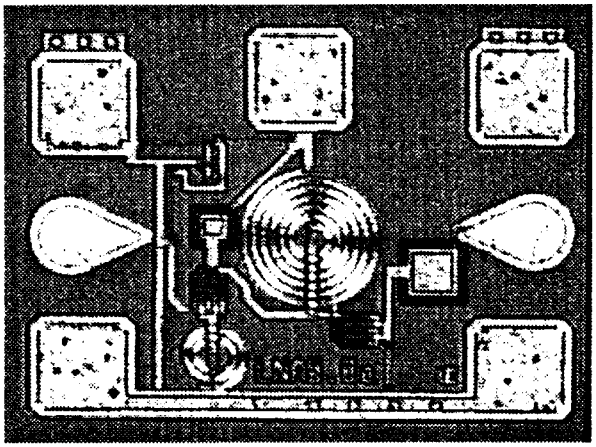
Figure 4: Circuit diagram of 10 GHz SiGe HBT LNA

first stage, the cascode topology is applied again. A parallel resonant circuit works as a reactive load with a resonance frequency of 10 GHz. Again, a common–collector stage increases the power gain further. The output capacitor serves for DC decoupling and enhances the output match of the amplifier.

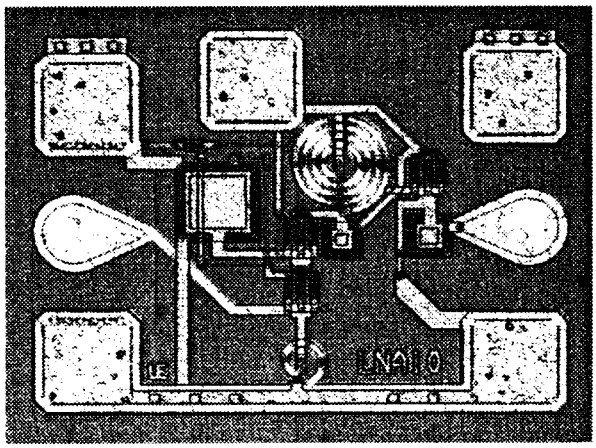
Each amplifier, described above, is realized on a chip with $450 \times 350\ \mu m^2$. Figure 5 shows the chip photograph of the realized 1.9, 5.7 and 10 GHz low–noise amplifier, respectively.



(a)



(b)



(c)

Figure 5: Chip photographs of the realized 1.9 GHz (a), 5.7 GHz (b) and 10 GHz (c) low–noise amplifiers

4 Measurement results and discussion

Measurements were done on–wafer using an ATN NP5B noise measurement setup. Due to a 20 percent higher sheet resistance of the bias resistors most of the circuits, originally designed for a single 3.6 V supply voltage, operate at 4.5 V. The measured supply current of all circuits agrees well with the corrected simulation. The 1.9 GHz LNA exhibit a minimum noise figure of approximately 1 dB and

an associated gain of 12.7 dB, as can be seen in figure 6 and 7, respectively. In the 5.7 GHz amplifier a minimum noise figure of 1.6 dB, comparable to the noise figure of a single transistor on the same wafer, is observed. The

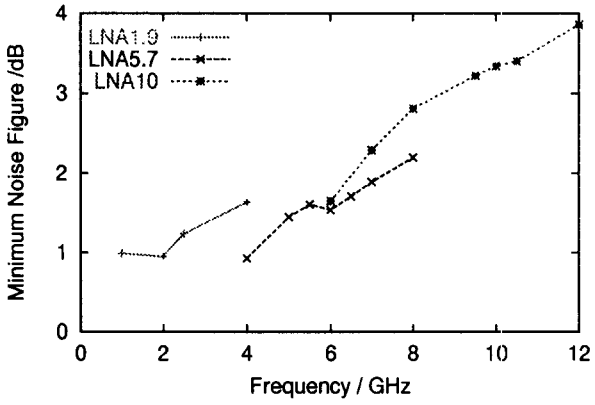


Figure 6: Noise performance of the SiGe LNAs under inverstigation; bias information: 1.9 GHz: 4.5 V/6 mA, 5.7 GHz: 4.5 V/7.4 mA and 10 GHz: 3.6 V/12 mA

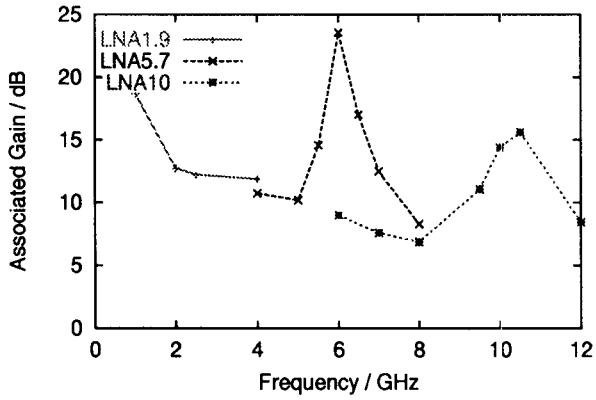


Figure 7: Associated gain vs. frequency for the investigated SiGe HBT LNAs, bias conditions: see figure 6

measured minimum noise figure of the 10 GHz LNA is 3.3 dB. This value is higher than the initial noise figure of a single transistor, depicted in figure 1. In fact, this is a result of the relative small gain of the first stage, which is not able to fully suppress the noise figure of the following stages, as expected by Friis's law [3].

Finally, two–tone intermodulation experiments were

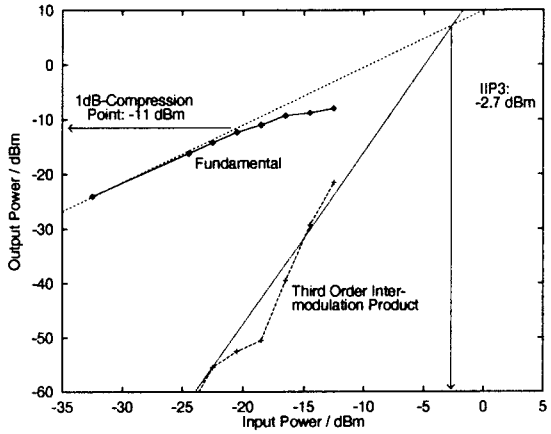


Figure 8: Two–tone intermodulation experiment for a 1.9 GHz LNA; $\Delta F = 50\ MHz$, 4.5 V/6 mA

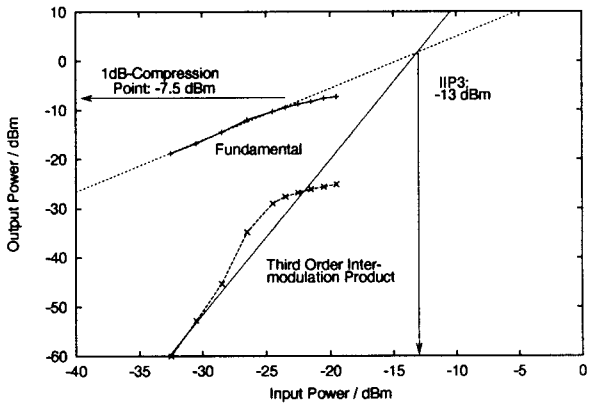


Figure 9: Two–tone intermodulation experiment for a 5.7 GHz LNA; $\Delta F = 50\ MHz$, 4.5 V/7.4 mA

performed on–wafer using a 50 Ω impedance environment. Except for the 5.7 GHz LNA, the amplifiers show a third order two–tone input intercept point of better than -10 dBm. The large signal performance is summarized in figures 8 to 10. The single–tone 1–dB–compression point of the 1.9, 5.7 and 10 GHz amplifiers are -8 dBm, -7 dBm and -5 dBm, respectively.

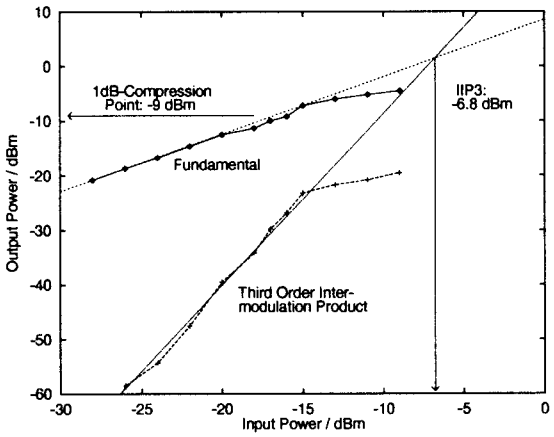


Figure 10: Two–tone intermodulation experiment for a 10 GHz LNA; $\Delta F = 50\ MHz$, 3.6 V/12 mA

5 Conclusion

It is shown that the noise performance of a single SiGe HBTs can be translated into comparable noise performance of LNA–MMICs for the frequency range of 1.9 up to 5.7 GHz. Minor degradation was observed for 10 GHz low–noise amplifiers. All amplifiers exhibit a power gain in excess of 12 dB, sufficiently high to suppress the noise contribution of other components in a RF receiver chain. These results demonstrate the suitability of SiGe HBTs for the application in low–noise amplifiers in a wide frequency band.

Acknowledgement

The author wold like to acknowledge the close collaboration with Professor H. Schumacher from the department on electron devices and circuits at the University of Ulm.

References

[1] R.J. Hawkins, "Limitation of Nielsen's and related noise equations applied to microwave bipolar transistors, and a new expression for the freuqency and current dependent noise figure", *Solid–State Electronics*, vol. 20, pp. 191–196, 1977.

[2] H. Schumacher and U. Erben, "Heterojunction bipolar transistors for noise–critical applications", in *Proceedings of XVIII SOTAPOCS, Vol. 93–27*, 1993, pp. 345–352.

[3] H.T. Friis, "Noise figure of radio receivers", in *Proceedings of IRE, Vol. 32, No. 7*, 1944, pp. 419–422.

[4] A. Schüppen, H. Dietrich, S. Gerlach, H. Köhnemann, J. Arndt, U. Seiler, R. Götzfried, U. Erben, and H. Schumacher, "SiGe–technology and components for mobile communication systems", in *Proceedings of 1996 Bipolar BiCMOS Circuit and Technology Meeting*, 1996, pp. 130–134.

Add-on-tools für den Schaltkreis- und Systementwurf

Dr. Peter Schwarz

Fraunhofer Institut für Integrierte Schaltungen, Außenstelle EAS Dresden
Zeunerstraße 38, 01069 Dresden
Tel. / Fax: (0351) 4640-730 / -703,
email: schwarz@eas.iis.fhg.de
URL: http:/www.eas.iis.fhg.de

Zusammenfassung

Trotz eines großen Angebots an leistungsfähiger CAD-Software für den Schaltkreis- und Systementwurf existieren Lücken, die nicht durch kommerziell vertriebene Software geschlossen werden können. Bei FhG IIS / EAS Dresden werden Tools zur Unterstützung von Modellierung, Simulation und Optimierung entwickelt, die helfen sollen, diese Lücken zu schließen. Einige dieser Tools könnten auch bei Partnern im Rahmen von Lehre und Forschung eingesetzt werden.

1. Softwareunterstützung beim Entwurf

Es gibt zahlreiche CAD-Systeme, die sowohl in der Industrie als auch in Lehre und Forschung erfolgreich beim Entwurf von integrierten Schaltkreisen, Leiterplatten und Systemen eingesetzt werden. Die Komplexität techni-

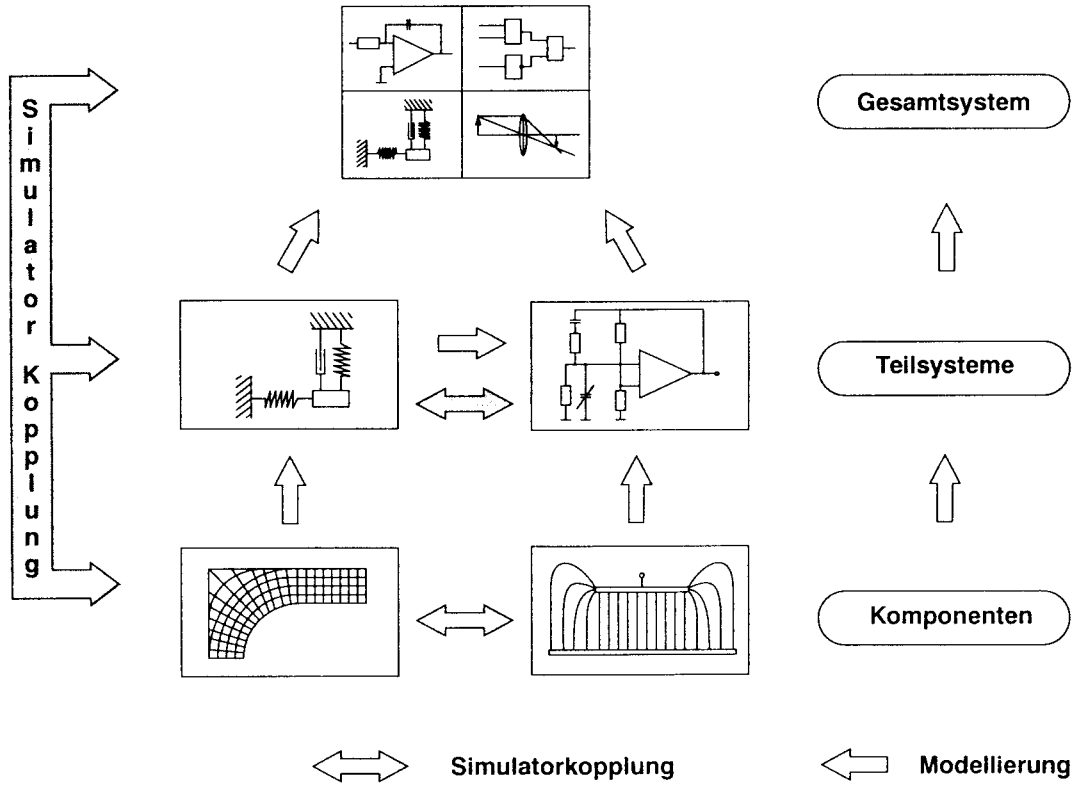


Bild 1 Abstraktionsebenen beim Entwurf heterogener Systeme

scher Systeme führt aber dazu, daß mit diesen kommerziell verfügbaren Tools zwar die meisten, aber nicht alle grundsätzlich für den CAD-Einsatz geeigneten Probleme rechnergestützt gelöst werden können. Nachrichtentechnik, Mechatronik und Mikrosystemtechnik sind dafür typische Beispiele [Jan00]. In **Bild 1** ist angedeutet, welche Abstraktionsebenen bei der Modellierung und Simulation komplexer Systeme betrachtet werden müssen. Einige der Arbeiten im FhG EAS Dresden sind darauf gerichtet, derartige „Nischen" in den CAD-Systemen durch eigene Toolentwicklungen zu füllen. Dazu gehören Modellierungswerkzeuge, Software zur Simulatorkopplung und ein Optimierungssystem. Soweit nicht durch Eigentumsrechte von Projektpartnern die Weitergabe dieser Tools erschwert ist, sind wir am Einsatz der Tools bei Partnern in Lehre und Forschung interessiert. Die Pflege und Weiterentwicklung der Tools kann zunehmend durch FhG EAS Dresden gesichert werden.

2. Kurzbeschreibung der Tools

2.1 Moscito

Das Optimierungssystem Moscito (Modular System for Constraint Nonlinear Microsystem Optimization) wurde zunächst für den Einsatz in der Mikrosystemtechnik entwickelt [SPS99], ist darüber hinaus aber in vielen Bereichen der Technik anwendbar. Es arbeitet simulationsbasiert, die Eigenschaften der zu optimierenden Systeme werden also in der Regel mittels Simulation bestimmt. Gegenwärtig sind die Simulatoren SPICE, Saber, ELDO, KOSIM sowie der FEM-Simulator ANSYS eingebunden (**Bild 2**), Erweiterungen durch andere Simulatoren und zusätzliche Optimierungsalgorithmen sind in Arbeit. Der Einsatz erfolgt vor allem für Schaltungs- und Systemoptimierung sowie für Modellparameterbestimmung. In Arbeit ist eine plattformunabhängige Implementierung, d.h. Moscito wird unter UNIX auf Workstations und unter WindowsNT auf PCs programmiert.

- **modularer Aufbau**
- **Integration vorhandener Werkzeuge**
- **Verteilung auf mehrere Rechner**
- **Internet-basierte Arbeitsweise möglich**
- **einheitliche, grafische Nutzerschnittstelle**

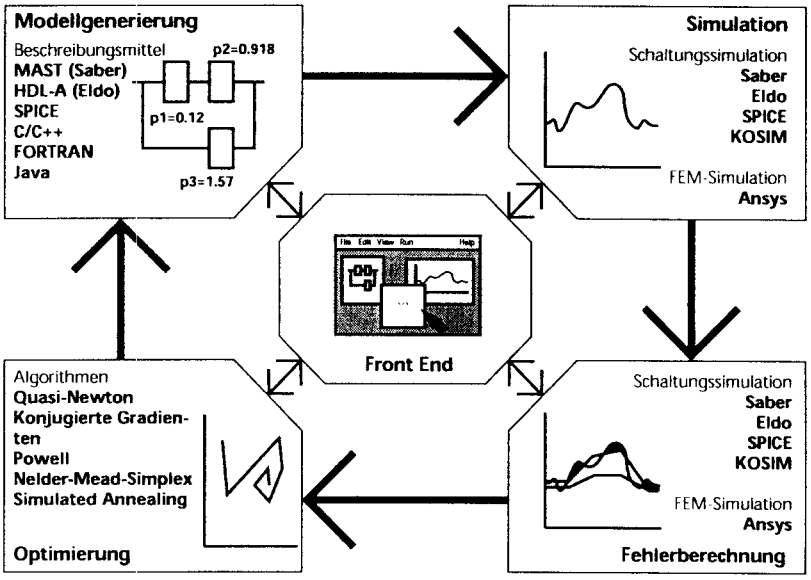


Bild 2 Optimierungssystem Moscito

2.2 KOSIM

KOSIM (Komplexe Simulation) ist ein Multi-Level-, Mixed-Mode-Simulator (**Bild 3**), der bereits in der DDR für die Mikroelektronikindustrie entwickelt wurde [SCD90]. Gegenwärtig erfolgt eine Neuimplementierung für Workstations und PC. Der Einsatz ist vor allem bei der Simulation heterogener Systeme, der Einbindung neuer Simulationsalgorithmen und Modellschnittstellen sowie für Simulatorkopplungen vorgesehen. Kopplungen wurden bisher z.B. mit VHDL-Simulatoren, mit ANSYS und mit einem Mechatronik-Simulator realisiert.

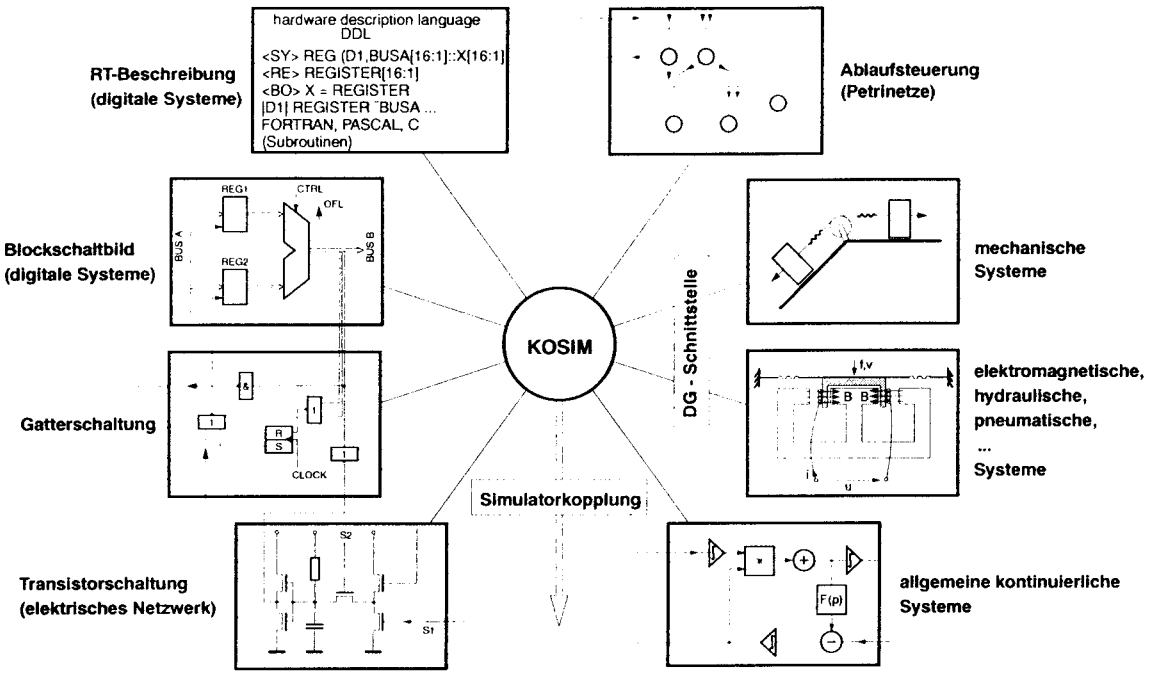


Bild 3 Multi-Level-, Mixed-Mode-Simulator KOSIM

2.3 ADDA

ADDA ist ein simulatorunabhängiges Ausgabetool für Simulationsergebnisse in unterschiedlichen Darstellungsformen (mixed-signal, waveforms, Frequenzgänge, Scatterdiagramme, ...). Es ist nur auf Unix-Workstations verfügbar. Als Schnittstelle wird ein verallgemeinertes CSDF-Format verwendet, das für die Anforderungen der Mixed-Signal-Simulation erweitert wurde.

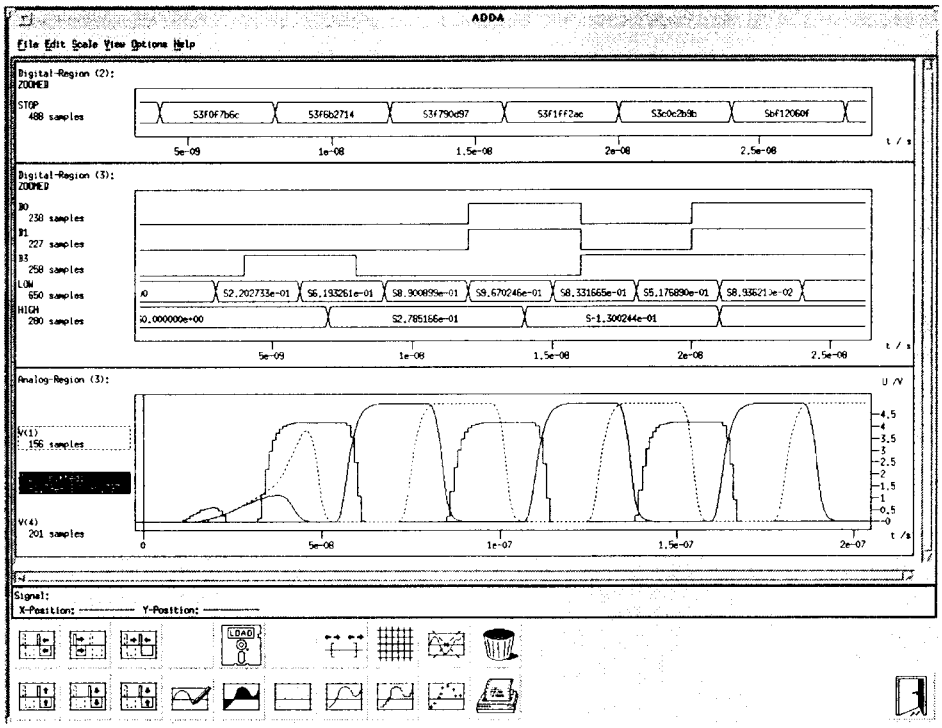


Bild 4 Mixed-Signal-Darstellung mit ADDA

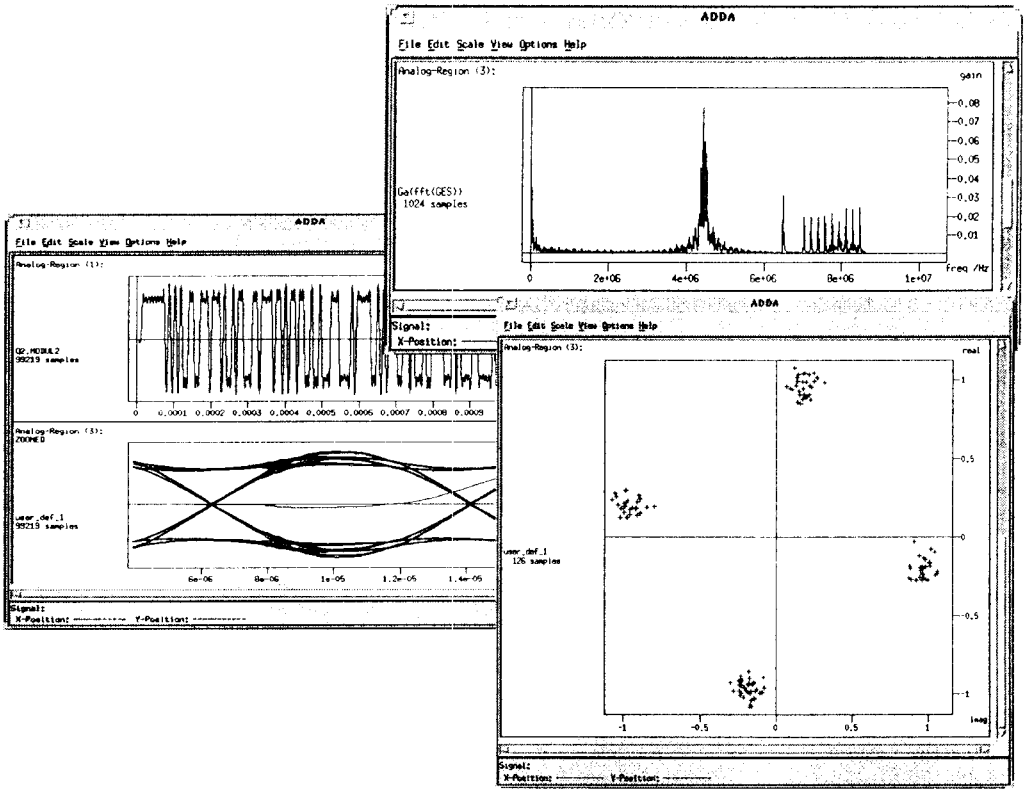


Bild 5 Ergebnisdarstellung von Simulationen in der Nachrichtentechnik mit ADDA

2.4 Modellbibliotheken

Im Rahmen öffentlicher Projekte wurden Modellbibliotheken für verschiedene Einsatzbereiche entwickelt.

NTS Nachrichtentechnische Systeme [EJS95]:
Modelle für Filter, Modulatoren und Demodulatoren, Koder/Dekoder, digitale Signalverarbeitung (Gleit- und Festkommaoperationen mit einstellbarem Zahlenformat)

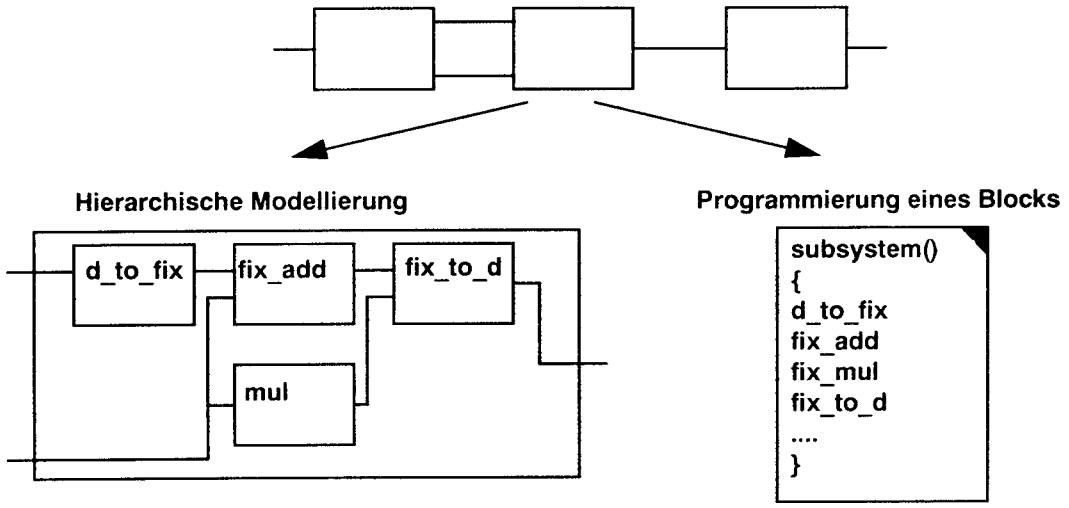
Im **Bild 6** ist angedeutet, wie das Modell für einen Block, der eine digitale Signalverarbeitungsaufgabe mit Festkommaarithmetik zu erfüllen hat, aussehen könnte. Es wird entweder aus Elementen, die elementare Operationen ausführen, zusammengesetzt, oder aber als „Verhaltensmodell" beschrieben, in dem Unterprogramme für die gleichen elementaren Operationen aufgerufen werden.

MIMOSYS Mikrosystemtechnische Komponenten, vor allem der Mikromechanik [NBL98]

Im **Bild 7** ist gezeigt, wie ein komplizierter mechanischer Sensor durch die Zusammenschaltung von Elementen modelliert wird. Ein solches Grundelement ist z.B. das Modell eines homogenen Biegebalkens. Die Gleichungen, die das Zusammenwirken der Kräfte und Verschiebungen sowie der Drehmomente und Drehwinkel beschreiben, sind als Matrixgleichung im Inneren des „Mehrpols" angegeben. Es handelt sich um eine Modellbildung mit „verallgemeinerten Netzwerken", mit deren Hilfe auch nichtelektrische Systeme mit Schaltungssimulatoren wie Saber und ELDO simuliert werden können.

CADWOK optoelektronische Komponenten (Laserdioden, PIN-Dioden, APD) [HKB99]

SIMKOS mechatronische Komponenten (einer Aufzugsteuerung);
FEM-Modelle und Netzwerkmodelle für Magnetkreisberechnungen [SSS99]



Elementare Blöcke (Module) für mathematische Grundoperationen (Gleitkommaarithmetik, parametrisierbare Festkommaarithmetik)

C-Funktionen für parametrisierbare Festkommaarithmetik

Bild 6 Modellierung eines Blockes für digitale Signalverarbeitung

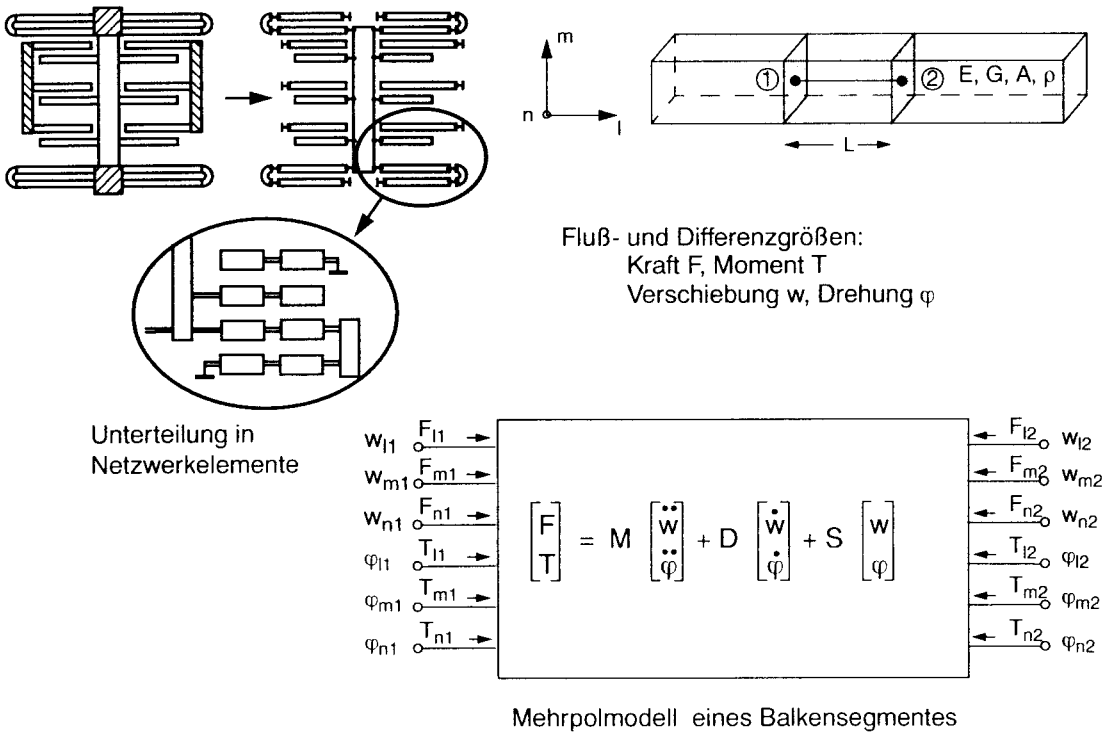


Bild 7 Modellierung eines mikromechanischen Beschleunigungssensors

2.5 Marabu

Die Approximation mehrdimensionaler nichtlinearer Funktionen erfolgt durch radiale Basisfunktionen (Multivariate Approximation with Radial Basis Functions). Ausgehend von einer Tabelle der zu approximierenden Kurvenverläufe werden Modelle in HDL-A, MAST und VHDL-AMS oder aber als C-Code generiert [Par97], **Bild 8**.

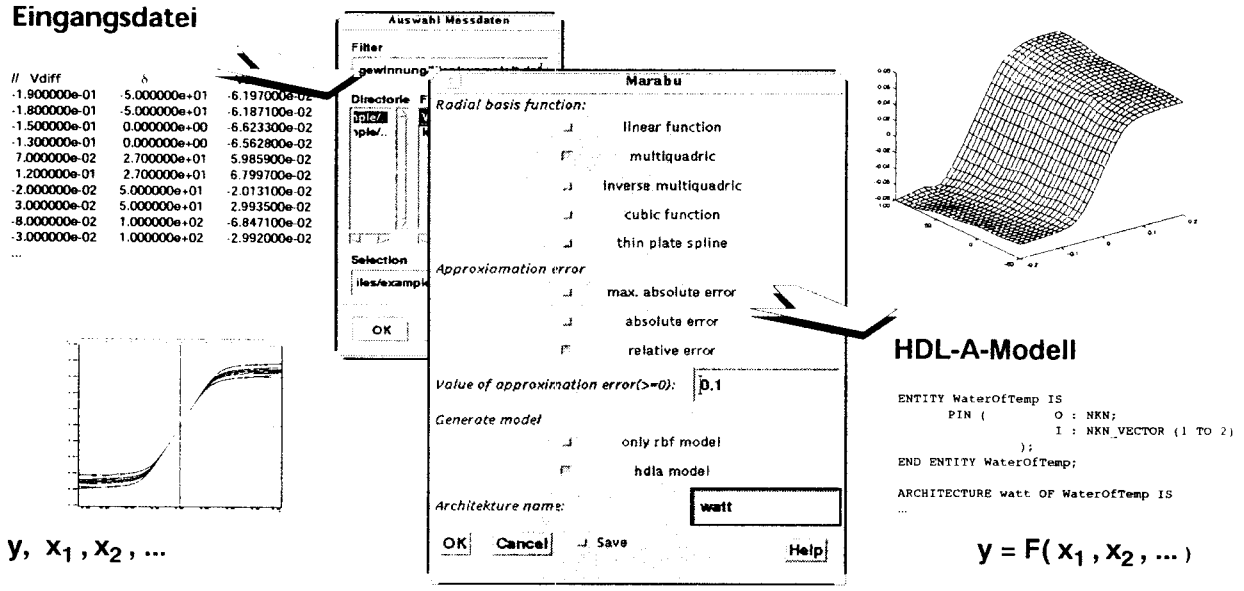


Bild 8 Approximation mehrdimensionaler nichtlinearer Kennlinienfelder mit MARABU

2.6 Thermischer Simulator und Modellgenerator TSMG

Die Berücksichtigung elektrisch-thermischer Wechselwirkungen (z.B. beim IC-Entwurf) kann durch geeignete Modellbildung für das thermische System mit einem Schaltungssimulator erfolgen (**Bild 9**a). Dafür kann mit dem Tool TSMG automatisch ein Modell generiert werden (**Bild 10**), das dann in Schaltungssimulatoren wie SPICE, Saber, ELDO,... eingebunden werden kann [ScW99]. Bei gegebener Verlustleistung der elektrischen Bauelemente wird die Temperaturverteilung berechnet und graphisch dargestellt. Eine Alternative ist die Kopplung eines Schaltungssimulators mit einem weiteren Simulator (**Bild 9**b) für das thermische System [WCS97].

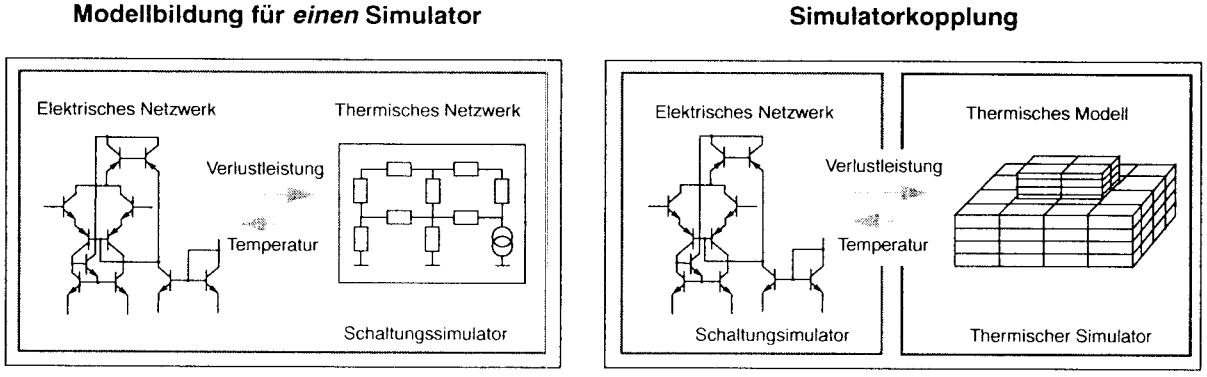


Bild 9 Zwei Ansätze zur Berechnung thermisch-elektrischer Wechselwirkungen

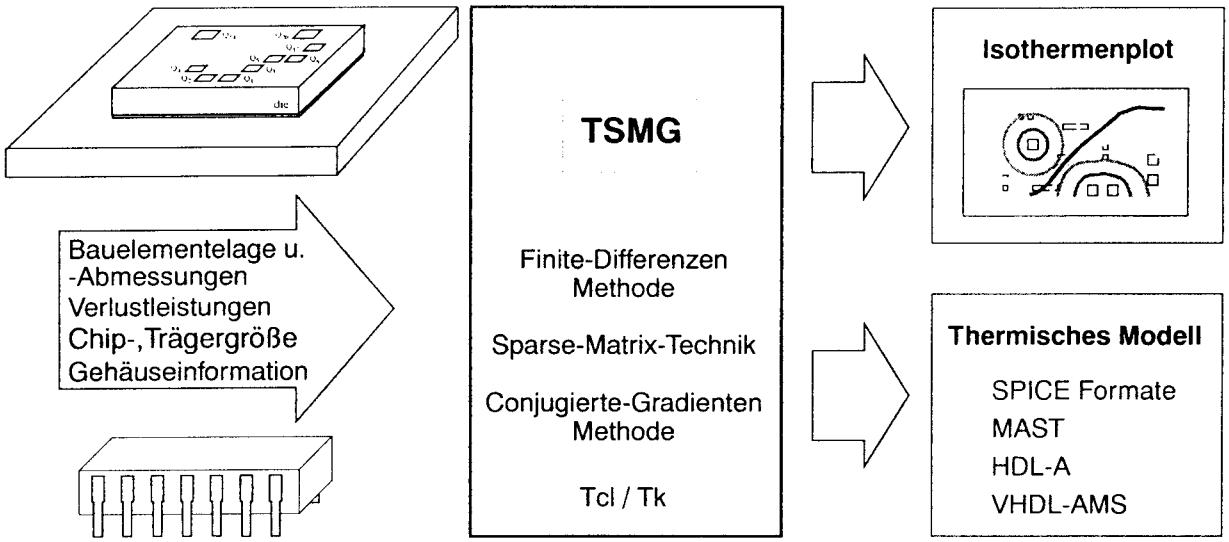


Bild 10 Thermisch-elektrischer Simulator und Modellgenerator TSMG

2.7 Analoger Fehlersimulator

In der Arbeitsgruppe „Test und Verifikation" von FhG EAS Dresden (Leiter: Dr. Bernd Straube) wurde ein System für die Modellierung und Simulation fehlerbehafteter Schaltungsbeschreibungen sowie ihrem Vergleich zum Sollverhalten (Modellierung und Simulation von Testvorgängen) entwickelt (**Bild 11**). Auch die Verteilung der Simulationsaufgaben im Rechnernetz wird unterstützt. Es wird in Verbindung mit dem Siemens-internen Simulator TITAN und mit dem Simulator Saber eingesetzt [SMV00].

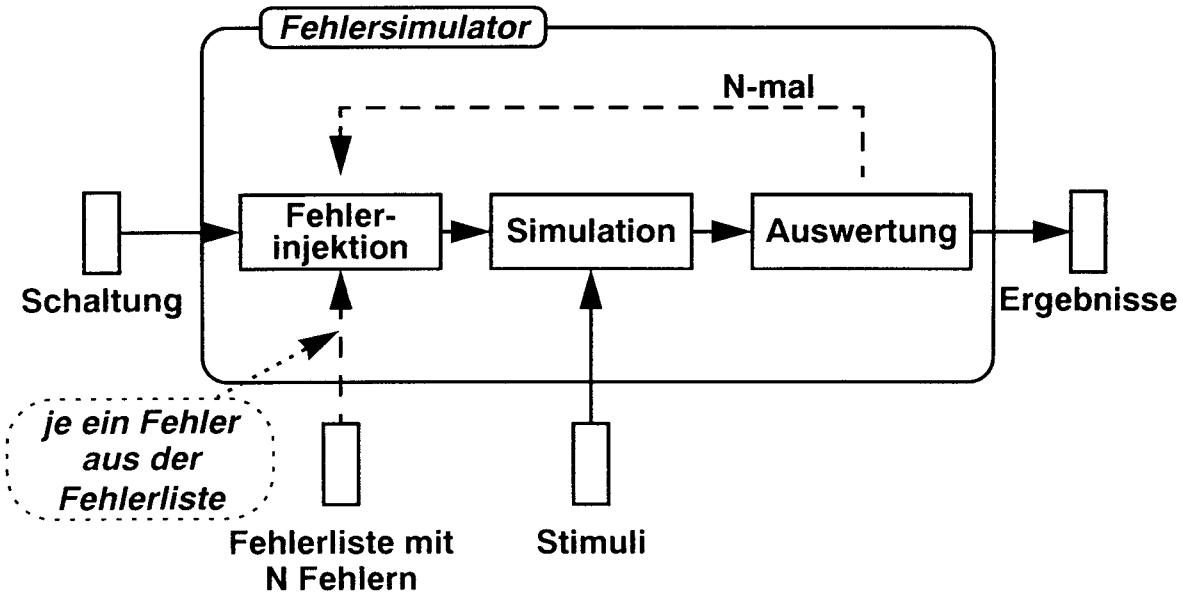


Bild 11 Serielle analoge Fehlersimulation

2.8 Simulatorkopplung

Für die Simulation heterogener Systeme ist die Leistungsfähigkeit eines Simulators oft nicht ausreichend. Daher wurden Kopplungen zwischen verschiedenen Simulatoren entwickelt, die vor allem in der Mikrosystemtechnik (z..B. Saber – ANSYS) [EKE96] und in der Nachrichtentechnik (z.B. COSSAP – Saber – Leapfrog) [EST96], [EKS99] eingesetzt werden. Im **Bild 12** ist gezeigt, wie beim Entwurf von Schaltkreisen der Nachrichtentechnik drei leistungsfähige Simulatoren so gekoppelt werden können, daß für die einzelnen Schaltungsteile der jeweils am besten geeignete Simulator verwendet werden kann. Gerade für den im **Bild 12** angedeuteten Modem-Schaltkreis ist es wichtig, daß auch das Zusammenwirken von analogen Schaltungsteilen mit den Blöcken für die digitale Signalverarbeitung simuliert werden kann.

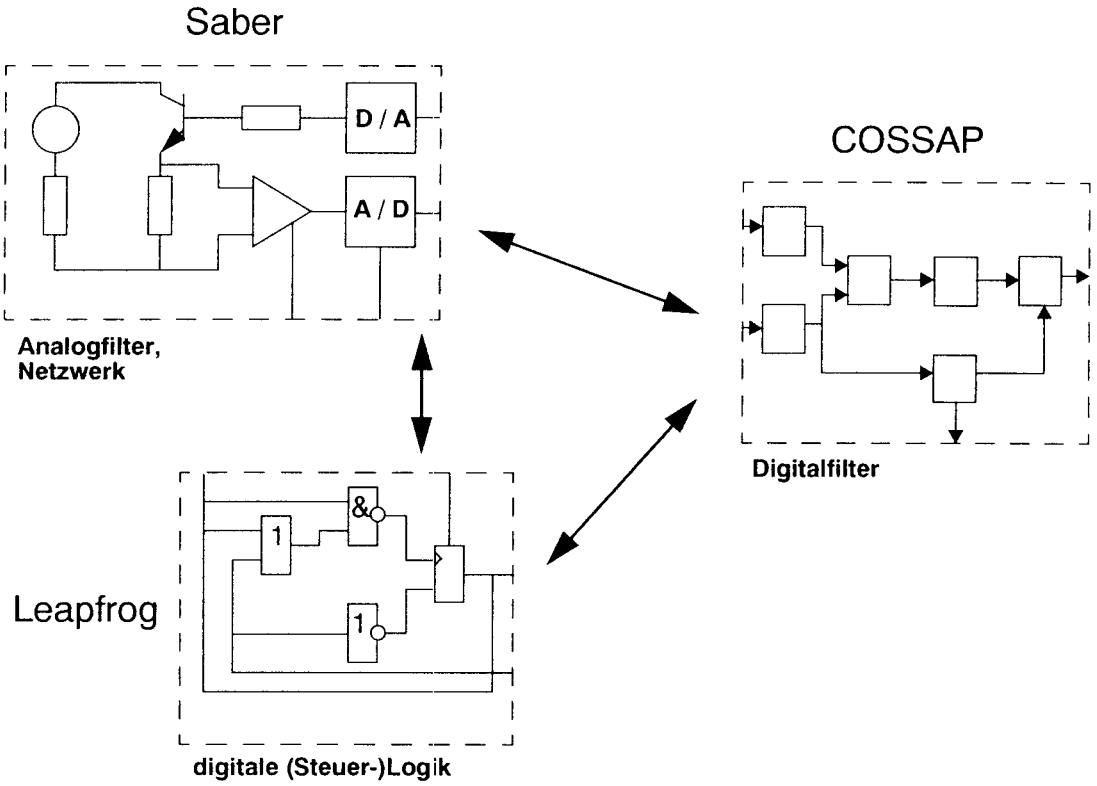


Bild 12 Simulatorkopplung für nachrichtentechnische Systeme

Der Einsatz gekoppelter Simulatoren für die Berechnung thermisch-elektrischer Wechselwirkungen ist schon im **Bild 9** gezeigt worden.

2.9 HW/SW-Cosimulation

Zur Kopplung von Logiksimulatoren und digitaler Hardware wurde eine Interface-Leiterkarte SimConnect und die zugehörige Treibersoftware entwickelt, **Bild 13**. Der Einsatz erfolgt vor allem beim Rapid Prototyping von FPGA-Boards und beim Debuggen dieser Boards. Außerdem konnte in einzelnen Anwendungsfällen eine erhebliche Beschleunigung der Logiksimulation erreicht werden [HSB98], [HFG00]. Auf dem Rechner läuft entweder ein C-Programm (beim HW/SW-Codesign) oder ein Logiksimulator. Die preiswerte Hardware macht einen Einsatz auch in der Ausbildung möglich. Im **Bild 14** sind die vielfältigen Einsatzmöglichkeiten zusammengefaßt.

Simulator- und C-Code Interface

Application Programming Interface (C-Library)

PROmetheus **SimConnect** Board

Portlogic (HDL-Library)

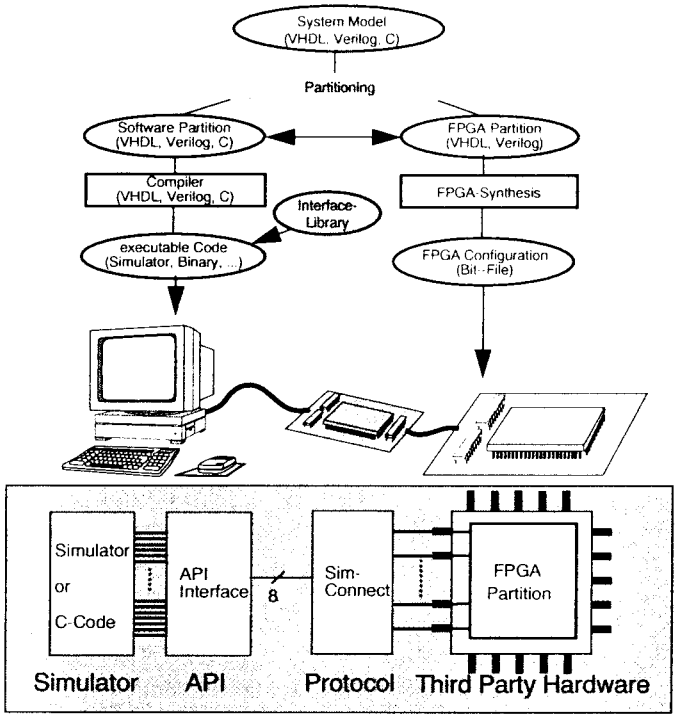
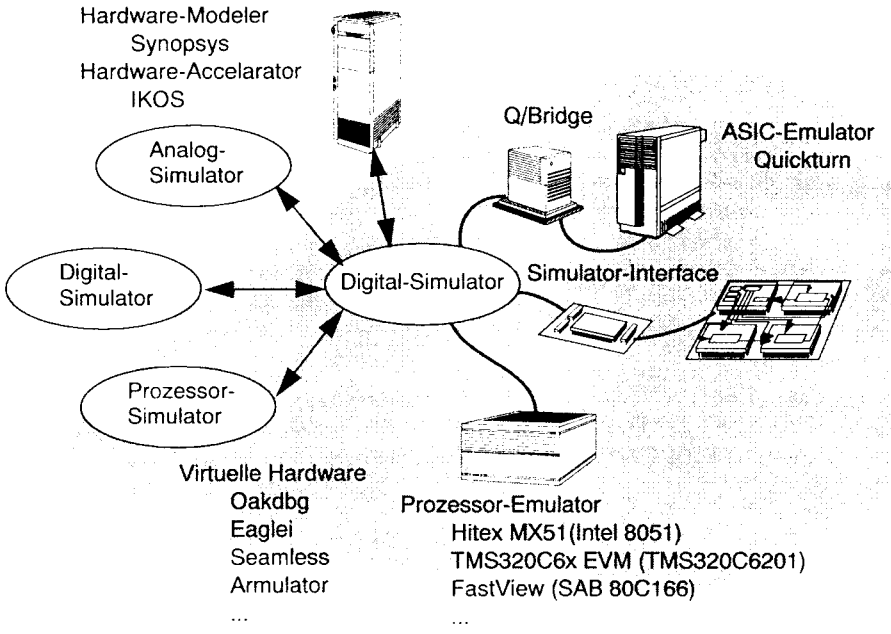


Bild 13 Designflow für eine Coverifikations-Umgebung



Simulations-beschleunigung

Re-Use von vorhandener Hardware

Inkrementeller FPGA-Entwurf

FPGA-Test

FPGA-Debugging

Data-Logging

HW/SW-Cosimulation

Echtzeitsimulation

Hardware-in-the-loop

Bild 14 Einsatzmöglichkeiten der HW/SW-Kopplung

3. Literatur

[EJS95] Engelmann, F.; Jentschel, H.-J.; Schwarz, P. (Hrg.): Tagungsband zum Workshop "Modellierung und Simulation in der Nachrichtentechnik", Dresden, 9. November 1995.

[EKE96] Eccardt, P.-C.; Knoth, M.; Ebest, G; Landes, H; Clauß, C.; Wünsche, S.: Coupled finite element and network simulation for microsystem components.Proc. MICRO SYSTEM Technologies '96, Potsdam, September 1996, pp. 145-150.

[EKS99] Einwich, K.; Knöchel, U.; Schwarz, P.: Modellierung und Simulation nachrichtentechnischer Schaltungen und Systeme. 13. ASIM-Tagung Simulationstechnik, Weimar, September 1999, Praxisforum, S. 15-20.

[EST96] Einwich, K.; Schwarz, P.; Trappe, P.; Zojer, H.: Simulatorkopplung für den Entwurf komplexer Schaltkreise der Nachrichtentechnik. 7. ITG-Fachtagung "Mikroelektronik für die Informationstechnik", Chemnitz, März 1996, S. 139-144.

[HFG00] Haufe,J.; Fritsch, Chr.; Gulbins, M.; Lück, V.; Schwarz, P.: Real-time debugging of digital integrated circuits. Proc. DATE'2000, User Forum, Paris, March 2000, pp. 235-241.

[HKB99] Hansel, G.; Kube, E.; Becker, J.; Haase, J.; Schwarz, P.: Entwurf von Systemen der Freiraum-Nachrichtenübertragung. 8.GMM-Workshop "Methoden und Werkzeuge zum Entwurf von Mikrosystemen", Berlin, Dezember 1999, S. 23-30.

[HSB98] Haufe, J.; Schwarz, P.; Berndt, T.; Große, J.: Accelerated logic simulation by using prototype boards. Proc. DATE'98 (Designer Track), Paris, February 1998, pp. 183-189.

[Jan00] Jansen, D. (Hrg.): Handbuch Electronic Design Automation. Hanser-Verlag, München 2000.

[NBL98] Neul, R.; Becker, U.; Lorenz, G.; Schwarz, P.; Haase, J.; Wünsche, S.: A modeling approach to include mechanical microsystem components into the system simulation. Proc. Conf. Design, Automation and Test in Europe DATE'98, Paris, Februar 1998, 510-517.

[Par97] Parodat, S.: MARABU - Ein Werkzeug zur Approximation nichtlinearer Kennlinien mit radialen Basisfunktionen. Proc. 6. Workshop "Methoden und Werkzeuge zum Entwurf von Mikrosystemen", Paderborn, Dezember 1997, S. 49-58.

[SCD90] Schwarz, P.; Clauß, C.; Donath, U.; Haufe, J.; Kurth, G.; Trappe. P.: KOSIM - ein Mixed-Mode, Multi-Level-Simulator. Informatik-Fachbericht 255 (Hrg.: B. Reusch), Springer, Berlin 1990, S. 207-220.

[ScW99] Schwarz, P.; Wünsche, S.: Modellierung und Simulation thermisch-elektrischer Wechselwirkungen in integrierten Schaltkreisen. 13. ASIM-Tagung Simulationstechnik, Weimar, Sept.1999, S. 265-272.

[SMV00] Straube, B.; Müller, B.; Vermeiren, W.; Hoffmann, C.; Sattler, S.: Analogue fault simulation by aFSIM. Proc. DATE'2000 (User Forum), Paris, March 27-30, 2000, pp. 205-210.

[SPS99] Schwarz, P.; Parodat, S.; Schneider, A.: Ein modulares Optimierungssystem für den Mikrosystem- und Schaltungsentwurf. 7. GMM-Workshop "Methoden und Werkzeuge zum Entwurf von Mikrosystemen", Paderborn, Januar 1999, S. 195-204.

[SSS99] Schneider, P. u.a.: Simulation Neuronaler Netze in komplexen Systemen. 8.GMM-Workshop "Methoden und Werkzeuge zum Entwurf von Mikrosystemen", Berlin, März 1999, S. 73-82.

[WCS97] Wünsche, S.; Clauß, C.; Schwarz. P.; Winkler, F.: Electro-thermal circuit simulation using simulator coupling. IEEE Trans. VLSI-5(1997)3 , pp. 277-282.

Beschreibungssprache für das Layout analoger integrierter Schaltkreise

Thomas Krüger
FH-Mannheim, Windeckstr. 110, 68163 Mannheim
0179/5137643, tkrueger@rumms.uni-mannheim.de

Zusammenfassung
Der Beitrag beschreibt eine im Rahmen einer Diplomarbeit entwickelte Beschreibungssprache für das Layout analoger integrierter Schaltkreise, die unabhängig von einem Betriebssystem oder einer bestimmten Entwicklungsumgebung ist. Sie ermöglicht das Beschreiben eines Layouts unabhängig von der verwendeten Technologie und gibt es nach Angabe der gewünschten Technologie als CIF-File aus.

1. Motivation

In der Vergangenheit wurde das Layout analoger Schaltkreise meist von Hand erstellt und so konnte man bestmögliche Symmetrie und ein kompaktes Design erzielen. Der einzige Nachteil dieser Methode ist der extrem hohe Zeitaufwand, vor allem bei nachträglichen Änderungen der Schaltung oder der Technologie. Eine andere Möglichkeit sind Programme, die das Layout automatisch erstellen. Dies ist die schnellste Methode ein Layout zu erstellen, allerdings sind die Programme sehr unflexibel, oft technologieabhängig und schwer zu erlernen.
Die hier vorgestellte Beschreibungssprache soll die Vorteile der anderen Methoden vereinen und deren Nachteile möglichst vermeiden. Die Beschreibungssprache orientiert sich vom groben Aufbau her an BALLISTIC[1], einer Layoutbeschreibungssprache die an der Universität in Toronto enwickelt wurde. Im Gegensatz zu BALLISTC ist sie jedoch völlig unabhängig von einer bestimmten Entwicklungsumgebung oder einem Betriebssystem, da sie das generierte Layout im CIF-Format (Caltec-Intermeadiade-Format) ausgibt, das von jeder Entwicklungsumgebung eingelesen werden kann und da sie als C++ Programm unter jedem Betriebssystem, für das es einen C++ Compiler gibt, lauffähig ist. BALLISTIC ist auch der eigentliche Auslöser für die Entwicklung einer eigenen Beschreibungssprache, da sich der Autor im Rahmen seiner Diplomarbeit eigentlich mit den Möglichkeiten von BALLISTIC beschäftigen sollte, aber keine Lizenzen mehr für Mentor Graphics GDT zu bekommen waren, und BALLISTIC nur mit diesem zusammen lauffähig ist. Somit war BALLISTIC nicht zu benutzen und man kam auf den Gedanken eine neue Beschreibungssprache ohne diese Nachteile zu entwickeln.

Layoutmethode	Zeitaufwand	Schwierigkeit des Designs	Flexibilität
manuell	sehr hoch	schwierig	sehr hoch
automatisch erstelltes Layout	sehr gering	mittel	niedrig
Layoutbeschreib-ungssprache	gering	leicht	hoch

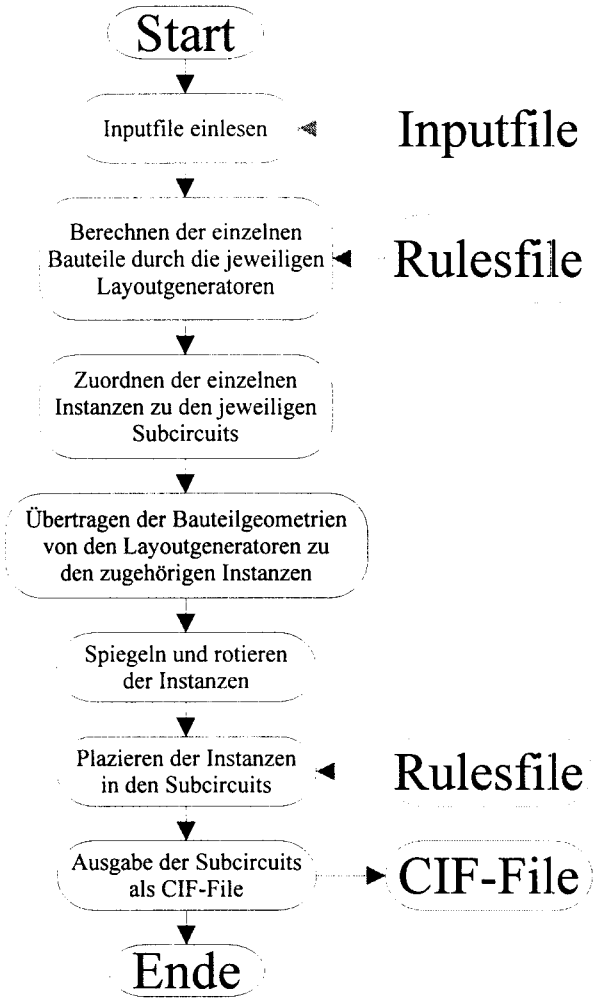
Tabelle 1: Vergleich verschiedener Design-möglichkeiten

Ziel der Layoutbeschreibungssprache ist es, dem erfahrenen Layoutdesigner ein einfaches und doch mächtiges Werkzeug zur Verfügung zu stellen, mit dem der Layoutprozess erheblich beschleunigt werden kann. Durch einfach parametrierbare Blöcke, wie Widerstände, Transistoren usw., einfaches Plazieren, absoluter Technologieunabhängigkeit und später auch mit einem automatischen Routing, können so Layouts deutlich schneller als bisher erstellt werden, bei gleichzeitig weniger Fehlern und einer maximalen Flexibilität in Bezug auf das Design. Durch das Trennen der Layoutbeschreibung, die im sog. Inputfile enthalten ist, von den Technologie- und Designregeln, die im sog. Rulesfile enthalten sind, ist es möglich, z.B. für eine Schaltung ohne zusätzlichen Aufwand das Layout für einen 0,3 µm CMOS-Prozess und einen 0,8 um BICMOS-Prozess zu errechnen.
Es werden allerdings immer noch intelligente Eingaben des Designers benötigt, da die Software nicht die Arbeit des Layouten selbst übernimmt, sondern nur unterstützt und bei unsinnigen Eingaben Fehler produzieren kann.

2. Der Programmablauf

Das Programm liest zu Beginn das sog. Inputfile, in dem sich alle technologieunabhängigen Angaben befinden, die es zum Erstellen des Layouts benötigt. Während des Einlesens der Daten wird für jeden Block, d. h. für jedes Bauteil, eine Klasse des zugehörigen Layoutgenerators angelegt und mit den Parametern des Blocks initialisiert. Weiter wird auch für jede Instanz und jeden Subcircuit eine Klasse angelegt und mit den im Inputfile enthaltenen Daten initialisiert.

Nach dem Einlesen des Inputfiles werden die verschiedenen Layoutgeneratoren gestartet. Diese lesen zuerst das Rulesfile ein, in dem sich die gesammten prozess- und technologieabhängigen Werte und Regeln befinden. Mit den Daten des Input- und des Rulesfile werden dann die Endabmessungen errechnet. Daraus wird die Anzahl der jeweiligen Polygone, z.B. Rechtecke, Vierecke, Hexagone, usw. errechnet und die Arrays, die die Koordinaten der Polygone beinhalten, initialisiert. Jetzt werden die gesamten Koordinaten der Polygone errechnet und in den Arrays abgelegt.



Zeichnung 1: Programmablauf

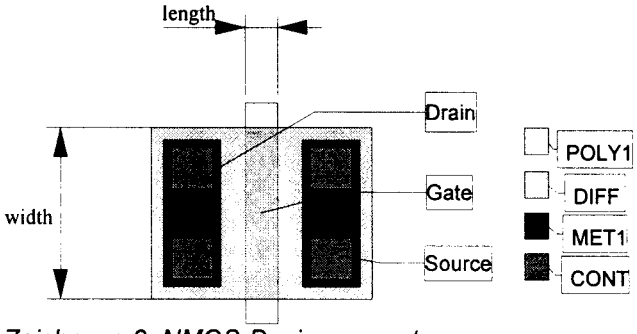
Wenn alle Blöcke berechnet sind, werden die Instanzen den jeweiligen Subcircuits zugeordnet und anschließend die Koordinaten der Polygone des zugehörigen Blocks in die Instanz übertragen. Im Gegensatz zu den bauteilspezifischen Blöcken sind die Instanzen alle identisch, d. h. sie sind vom Programmcode und ihren Funkionen her identisch, aber natürlich nicht von den in ihnen enthaltenen Geometriedaten her. Dadurch können sie bei der Plazierung alle gleich behandelt werden, was den Ablauf deutlich vereinfacht.
Nach dem Kopieren der Geometrien werden die Instanzen je nach Vorgabe gedreht und gespiegelt.

Jetzt liest das Hauptprogramm die Abstandsregel für die in den Instanzen benutzten Layer ein und beginnt mit dem Plazieren der Instanzen in den Subcircuits. Zuerst errechnet es mit den Koordinaten der Polygone den minimalen erlaubten Abstand. Dann wird mit den Optionen des move-Befehls die gewünschte Position der Instanz errechnet und der Instanz zugewiesen, sofern der minimal zulässige Abstand nicht unterschritten wird. Wird der minimal zulässige Abstand nicht eingehalten, wird die Instanz mit dem minimal zulässigen Abstand plaziert.
Zum Schluß wird das Ergebnis als CIF-File auf die Festplatte geschrieben, das von jeder beliebigen Entwicklungsumgebung eingelesen werden kann.

3. Beispiele für Devicegeneratoren

3.1 Der NMOS-Devicegenerator

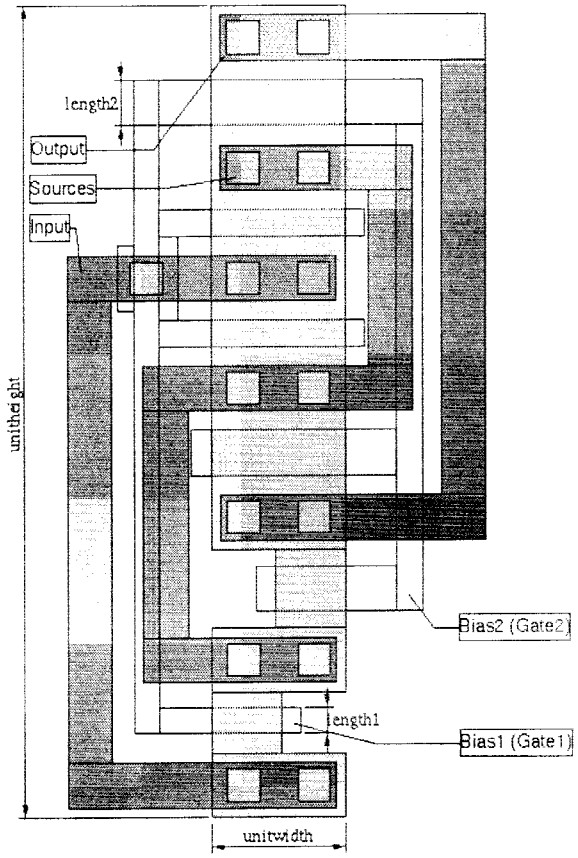
Zunächst als einfachstes Beispiel den Generator für die NMOS-Transistoren. Er ist in der Gatelänge und -breite parametrierbar. Aus diesen zwei Angaben wird der Rest der Abmessungen und die Anzahl der enthaltenen Polygone errechnet. In diesem einfachen Beispiel ist allerdings nur die Anzahl der Kontaktlöcher im Source- bzw. Drain-Anschluß variabel, d.h. von der Gatebreite abhängig.



Zeichnung 2: NMOS-Devicegenerator

3.2 Der Mirror-Devicegenerator

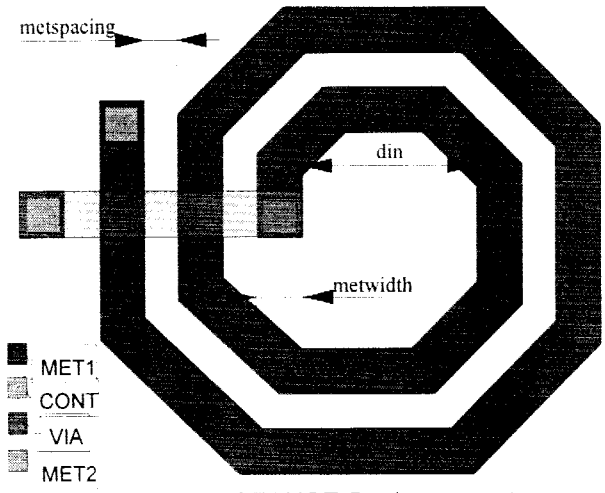
Etwas mehr Möglichkeiten zur Parametrierung bietet der Generator für NMOS-Stromspiegel: Hier sind die Gateflächen nach dem Matchingprinzip aufgeteilt. Die Gatelänge ist für beide Transistoren getrennt über die Parameter „length1" und „length2" einstellbar. Die Gatebreite wird über den Parameter width bestimmt und ist für beide Transistoren gleich. Zusätzlich wird einer der 3 folgenden Parameter benötigt: Entweder die Devicehöhe „unitheight", die Breite der Gateteilflächen „unitwidth" oder die Anzahl der Gateteilflächen, der Parameter „multiplicity". Aus dem Angegebenen Parameter ergeben sich jeweils die anderen Parameter und damit die Gesamt-abmessungen des Bauteils.



Zeichnung 3: MIRROR-Devicegenerator

3.3 Inductance-Devicegenerator

Als letztes Beispiel der Generator zur Erstellung von Induktivitäten. Es wurde eine achteckige Spirale gewählt, da sie der kreisrunden Spule am nächsten kommt und 45 Gradwinkel in jedem aktuellen Prozess erlaubt sind. Hier lassen sich 4 Parameter einstellen und zwar die Breite des Metalls, der Abstand der Metallbahnen von einander, der innere Durchmesser der Spirale und die gewünschte Induktivität in nH. Aus diesen Angaben wird dann die Anzahl der Windungen errechnet.



Zeichnung 4: INDUCTANCE-Devicegenerator

4. Plazieren der Bauelemente

4.1 Spiegeln und Rotieren

Es sollen an ein paar einfachen Beispielen die Möglichkeiten des Plazierens demonstriert werden. Dazu sind jeweils die Befehle und der zugehörige Screenshot darunter dargestellt. Zunächst können die Bauteile, die immer erst mit dem instance-Befehl in eine Instanz kopiert werden müssen, durch den angehängten Parameter flip in x- oder y-Richtung spiegeln. Hier wird die zweite Instanz in x-Richtung gespiegelt. Die move-Befehle dienen dem Zuweisen einer absoluten bzw. einer relativern Position. Mindestens einem Bauteil muß eine absolute Position zugewiesen werden. Die anderen Bauteile können dann relativ dazu angeordnet werden. Hier wird T2 rechts und mittig zu T1 angeordnet.

Entsprechend dem flip-Parameter gibt es auch den rotate-Parameter zum Rotieren eines Bauteils in 90 Grad-Schritten.

```
• instance T1 of cas1;
• instance T2 of cas1 flip=x;
• move T1 to 0,0;
• move T2 direction=right T1
  option=center;
```

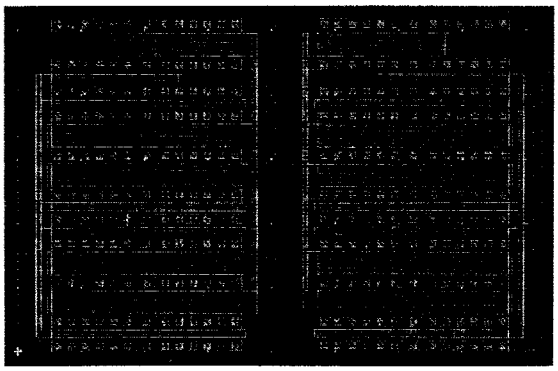


Abbildung 1: Spiegeln

```
• instance T1 of cas1;
• instance T2 of cas1 rotate=90;
• move T1 to 0,0;
• move T2 direction=right T1
  option=center;
```

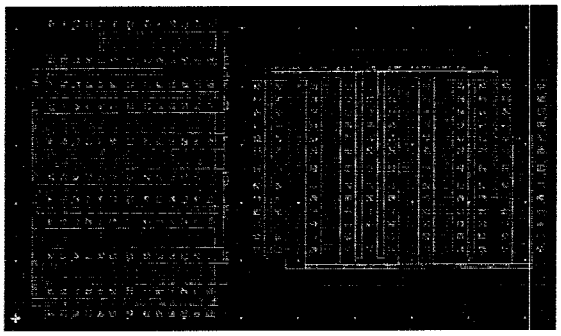


Abbildung 2: Rotieren

4.2 Relatives Plazieren

Weitere mögliche Parameter zum Plazieren eines Bauteils sind die Parameter xoffset, yoffset und sep, die auf den ersten Blick fast gleiche Auswirkungen haben. Der x- und yoffset wird zu einer errechneten Position hinzugefügt, sep gibt im Gegensatz dazu immer einen Gesamtabstand, also inklusive des minimal zulässigen Abstands an.

- `move R1 to 0,0;`
- `move R2 direction=above R1 option=center xoffset=10;`

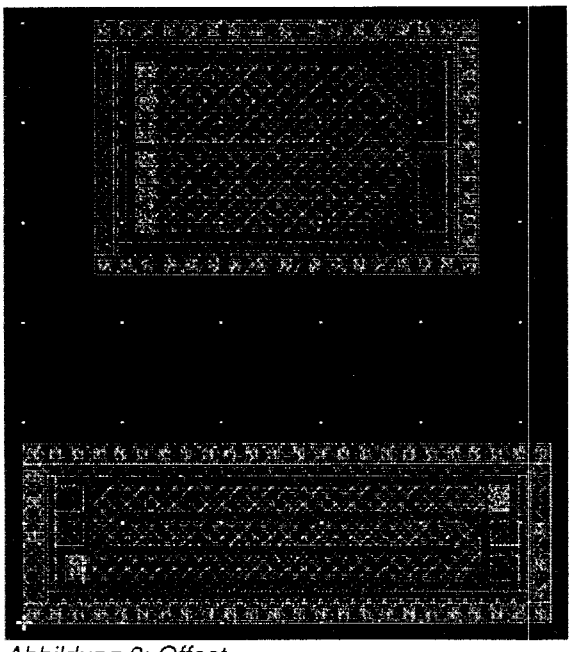


Abbildung 3: Offset

- `move R1 to 0,0;`
- `move R2 direction=above R1 option=center sep=15;`

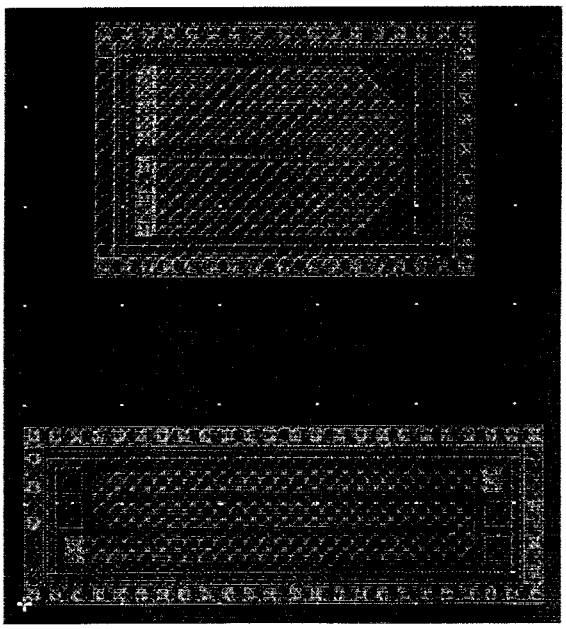


Abbildung 4: Separation

Zusätzlich zum zentrierten Ausrichten, kann man die Bauteile auch so ausrichten, daß die linken bzw. rechten Kanten übereinstimmen, oder bei seitlicher Anordnung die oberen bzw. unteren Kanten. Hier im Beispiel sind die Bauteile mit den rechten Kanten zueinander angeordnet. Weiter kann man auch bestimmte Kanten auf eine absolute Koordinate legen, hier wird die untere Kante auf 3,5 µm vom Nullpunkt gelegt.

- `move R1 to 0,0;`
- `move R2 direction=above R1 option=redges;`

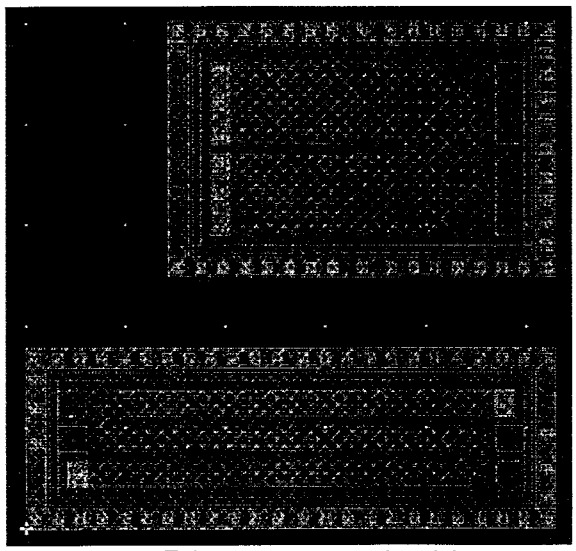


Abbildung 5: Ecken zueinander Ausrichten

- `move R1 to 0,0;`

- ```
  move R2 direction=right R1
  abs_coordinate_direction=bottom
  abs_coordinate_value=3.5;
  ```

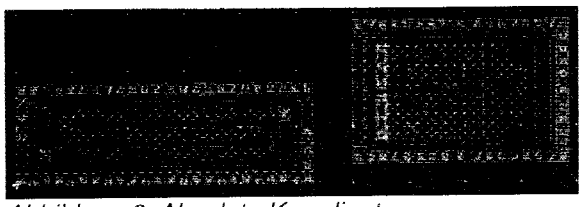


Abbildung 6: Absolute Koordinaten

5. Beispiel

Zum Abschluß noch ein Beispiel mit mehreren Bauteilen. Hier ist das komplette Inputfile angegeben und man kann gut erkennen, daß es sich in 3 Abschnitte gliedert: 1. Die Deklarationen, in denen alle verwendeten Bauteile, Instanzen und Zellen angegeben werden müssen. 2. Der sog. Blocksabschnitt, in dem alle Bauteile parametiert werden. Und 3. der sog. Subcircuitsabschnitt, in dem die berechneten Bauteile in die Instanzen kopiert und dann plaziert werden.

```
declarations
{
logical_layers = yes;
nmos nmos1;
pmos pmos1;
diffpair diff1;
inductance 45nH;
respoly1n 100k;
instance L1,R1,T1,T2,T3;
subcircuit circuitA;
}
blocks
{
nmos1 width=2.0 length=1.0;
pmos1 width=2.4 length=6.7;
diff1          width=35          length=0.8
multiplicity=4;
45nH        inductance=8        metwidth=2
metspacing=1.6 din=8;
100k res=7000 width=2.5 multiplicity=10;
}
subcircuits
{
circuitA
{
instance L1 of 45nH;
instance R1 of 100k;
instance T1 of nmos1;
instance T2 of pmos1;
instance T3 of diff1;
move R1 to 0,0;
move L1 direction=above R1 option=center
sep=4;
move T3 direction=right L1 option=center
xoffset=10;
move      T2      direction=above      T3
option=ledges;
move      T1      direction=below      T3
option=ledges;
}
}
```

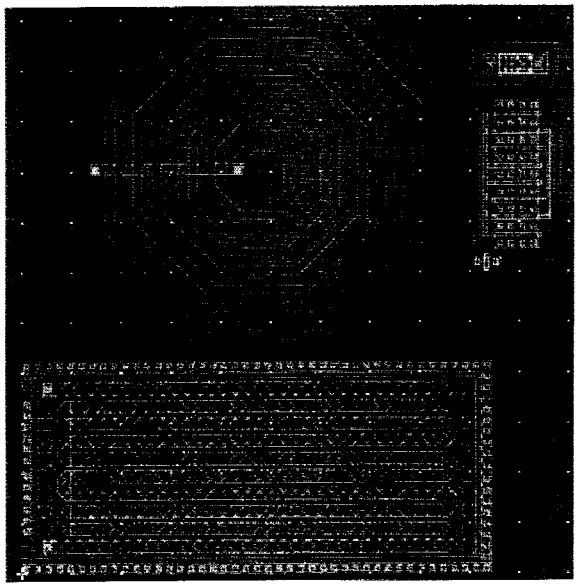


Abbildung 7: Beispiel

6. Ausblick

Das Programm steht mit ca. 3 Monaten Entwicklungszeit erst am Anfang seiner Entwicklung. Es werden weitere Devicegeneratoren für Bipolartransistoren, gematchte Widerstände, gematchte Kapazitäten, usw. folgen. Eine der größten Beschränkungen der jetztigen Version ist, daß das Routing komplett fehlt. In einer späteren Version sollen Punkt zu Punkt Verbindungen geroutet, der dafür nötige Platz zwischen den Bauteilen errechnet und der benötigte Abstand automatisch durch entsprechendes verschieben der Bauteile geschaffen werden. Auch kann man sich einen speziellen Editor für Input- bzw. Rulesfile vorstellen, da die Eingabe mit einem Standardeditor nicht besonders komfortabel ist und vor allen keine Syntaxprüfung durchgeführt werden kann. Weiter könnten noch weitere Ausgabeformate zusätzlich zum CIF-Format hinzugefügt werden.
```

7. Quellen

1: Bryn R. Owen, An Analog VLSI Layout Language, 1996, www.eecg.toronto.edu/~gdt/

Entwurfsautomatisierung mit ACSYN, dargestellt am Beispiel einer Mischerschaltung für DCS-Anwendungen

Meik Widmer, Frank Gruson, Gerhard Forster
TEMIC Semiconductor GmbH, Lise-Meitner-Str. 15, 89081 Ulm
Fachhochschule Ulm, Prittwitzstraße 10, 89075 Ulm

Das Entwurfswerkzeug ACSYN (Analog Circuit SYNthesis) wurde an der Fachhochschule Ulm in Zusammenarbeit mit der TEMIC Semiconductor GmbH entwickelt. Ziel ist es, die Entwicklungszyklen von analogen Hochfrequenzschaltungen zu verkürzen. ACSYN integriert dabei die drei Gebiete Wissenkodierung, automatische Dimensionierung und Charakterisierung. Im Rahmen einer Diplomarbeit wurde ein Modul zur automatischen Dimensionierung einer Mischerschaltung für Mobilfunkanwendungen erstellt und in ACSYN eingebaut. Dieser Artikel beschreibt zum einen die Funktionalität und den Aufbau von ACSYN und zum anderen wird der Algorithmus zur Mischer-Dimensionierung grob umrissen.

1 Einleitung

Im Kommunikationssektor gibt es zur Zeit extreme Wachstumspotentiale. Speziell im Mobilfunkbereich wird der Markt so stark forciert, daß etwa im Jahresrhythmus neue, verbesserte Endgeräte verlangt werden. Dadurch verkürzen sich natürlich auch die Entwicklungszyklen für integrierte Hochfrequenzschaltungen. Bei der Analyse analoger Frontends zeigt sich, daß bestimmte Kernkomponenten wie LNA, Mischer und VCO auf Konzepten mit einem hohen Wiederverwendungsgrad basieren [1, 2, 3].
Wird ein neuer Chipsatz entwickelt, so kann die Schaltungs-Architektur oft unverändert übernommen werden und es ist ausreichend, die Dimensionierung anzupassen. Ein erfahrener Entwickler wird dabei für verschiedene Spezifikationen immer wieder die prinzipiell gleichen Verfahren und Arbeitsschritte anwenden. Dabei wird das Wissen in Form der Schaltungsarchitektur, der veränderbaren Größen und des Entwurfsalgorithmus eingebracht. Ziel ist die spezifikationsgetriebene Dimensionierung.
Die Halbleiterhersteller sind daher bestrebt diese Dimensionierung zu automatisieren. Hier setzt ACSYN an. Ziel ist es, das Entwurfs-Wissen eines erfahrenen Entwicklers zu konservieren und quasi auf Knopfdruck automatisch anzuwenden. Zusätzlich stellt ACSYN Methoden zur umfangreichen, professionellen Dokumentation zur Verfügung.

2 Das EDA-Tool ACSYN

Das Entwurfswerkzeug ACSYN ist im Rahmen des europäischen MEDEA-Projekts an der Fachhochschule Ulm in Zusammenarbeit mit der TEMIC Semiconductor GmbH entstanden [4]. Die Einsatzgebiete von ACSYN sind: • Konservierung von Schaltungsentwurfs-Wissen (Wissenskodierung),
• Anwendung des Wissens (automatische Dimensionierung) und • Auswertung sowie Dokumentation (Charakterisierung).

2.1 Wissenskodierung

Bei der Wissenskodierung soll das Know-How eines erfahrenen Entwicklers möglichst umfassend in das Entwurfswerkzeug integriert werden.
Die Wissenskodierung erfolgt bei ACSYN nach dem in Abb.1 dargestellten Schema: Spezifikation und Schaltplan werden einem Code-Generator übergeben, welcher daraus einen Rahmencode zur Dimensionierung erzeugt. Die Spezifikation läßt sich dabei in Randbedingungen und Design-Ziele aufteilen. Unter Randbedingungen werden hier Vorgaben aus dem Umfeld der Schaltung verstanden. Bei einem Mischer sind das beispielsweise Versorgungsspannung, maximale Stromaufnahme, Umgebungstemperatur und Amplitude des externen LO-Signals. Die Design-Ziele sind hier: Kom-

pressionspunkt, Konversionsverstärkung, Rauschmaß und Ausgangsimpedanz. Der Schaltplan liefert Topologie, Technologie, Transistortypen, die zur Dimensionierung freigegebenen Bauteile und die notwendige Testbeschaltung. Der Code-Generator stellt dann automatisch alle zur Dimensionierung notwendigen Variablen aus Schaltplan und Spezifikation im C-Code bereit. Dies bedeutet konkret, daß beispielsweise für die Arbeitswiderstände des Mischers automatisch Variablen im C-Code bereitgestellt werden und somit der Zugriff auf diese Widerstände ermöglicht wird. Ziel des Code-Generators ist die einfache Implementierung des Entwurfs-Wissens.

In den Rahmencode fließen dann der eigentliche Entwurfsplan des Entwicklers in Form eines C-Codes, sowie Daumenregeln, symbolische Gleichungen und Charakterisierungsmodule [5] ein. Die Charakterisierungsmodule werden dabei benötigt, um komplexere elektrische Größen zu bestimmen, die nicht direkt vom Simulator geliefert werden. Beispiele hierfür sind der 1dB-Kompressionspunkt sowie die Interceptpunkte.

Aus diesen Daten wird dann durch ACSYN ein lauffähiges Objekt erzeugt, wobei der gesamte C-Code automatisch compiliert und gelinkt wird. Mit dem lauffähigen Objekt kann dann die Dimensionierung durchgeführt werden.

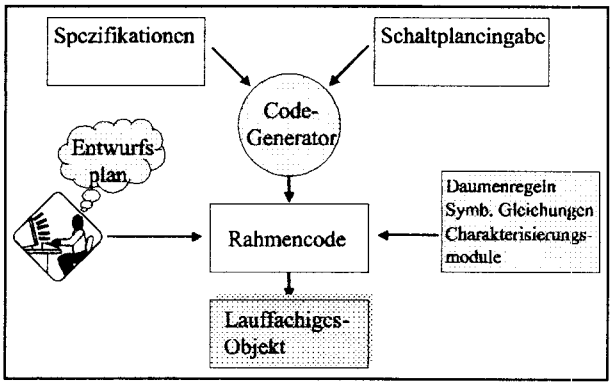


Abbildung 1: Wissenskodierung

2.2 Schaltungs-Dimensionierung

Die Kernkomponente der automatischen Dimensionierung mit ACSYN bildet das lauffähige Objekt. Die notwendigen Spezifikationsdaten werden vom Anwender in die entsprechende Eingabemaske eingetragen. Wie in Abb.2 gezeigt, greift das Objekt während der Dimensionierung sowohl direkt, als auch über Charakterisierungsmodule, auf den Simulator zu und ermittelt so die notwendigen elektrischen Parameter der Schaltung. Um die geforderte Spezifikation zu erreichen, werden die entsprechenden Bauteile der Schaltung gemäß Entwurfsplan innerhalb der Netzliste verändert.
Am Ende der Dimensionierung ergibt sich ein kompletter Satz von Bauteilparametern, mit dem die Spezifikation im Idealfall vollständig erreicht wird. Die so ermittelten Werte können dann automatisch an das Schematic übergeben werden (Backannotation).

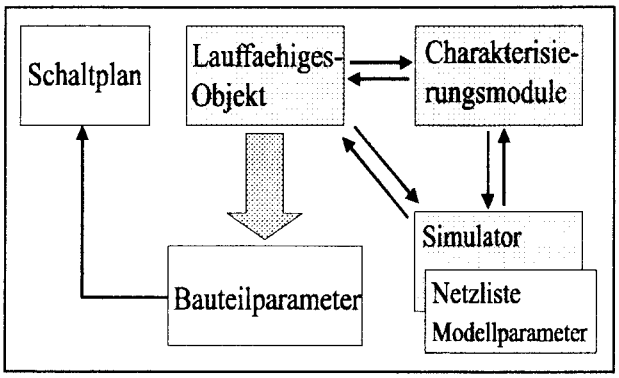


Abbildung 2: Schaltungs-Dimensionierung

2.3 Schaltungs-Charakterisierung

Die Charakterisierung dient der professionellen, umfassenden Dokumentation und Auswertung einer Schaltung. Zur Charakterisierung stellt der Anwender Schritt für Schritt ein interaktives Datenblatt zusammen, das alle für die Schaltung relevanten Parameter enthält (Abb.3). Hierbei werden durch ACSYN unter anderem weit verbreitete Tools wie der Design-Architect oder Eldo [6] von Mentor Graphics verwendet, um die Daten zu verwalten. Die betreffenden Größen können daraufhin automatisch ermittelt und sowohl graphisch als auch skalar dargestellt werden. Bei der Dokumentation einer Schaltung ist es notwendig, jederzeit die Bedingungen nachvollziehen zu können, unter denen ein bestimmter Parameter ermittelt wurde. Durch das interaktive Datenblatt können stets alle relevanten Daten angezeigt werden. Beispielsweise wird für jeden Parameter (jede Zeile) die komplette Testumgebung gespeichert; diese kann auf Knopfdruck mit dem Design-Architect angezeigt werden.

Somit wird eine umfassende Dokumentation der Schaltung, der Testbenches und der elektrischen Parameter erreicht. Als Beispiel ist in Abb.3 die Charakterisierung einer Schaltung, bestehend aus Mischer und LNA dargestellt. Mit jeder Zeile des

Datenblatts wird genau ein elektrischer Parameter der Schaltung bearbeitet. Alle zugehörigen Optionen und Aktionen können ebenfalls in der jeweiligen Zeile aufgerufen werden.

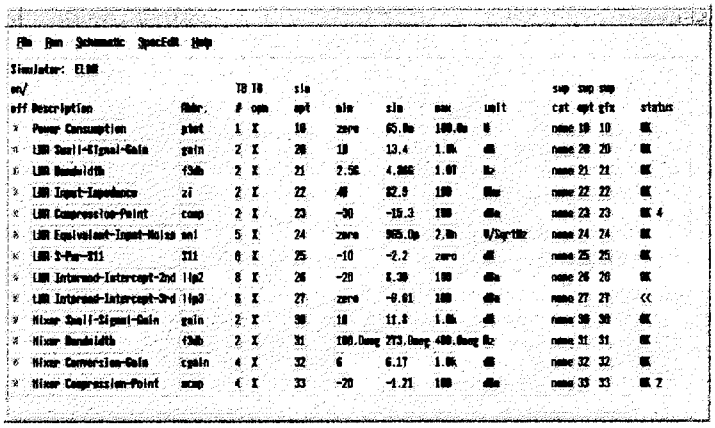


Abbildung 3: Charakterisierungs-Oberfläche

3 Wissenskodierung am Beispiel eines Mischers

Die Codierung des Wissens erfolgt in Form der Schaltpläne und des Entwurfsplans. Hier soll nun zunächst die Schaltung, dargestellt in Abb. 4, erläutert werden. Der Entwurfsplan wird im Anschluß dargestellt.

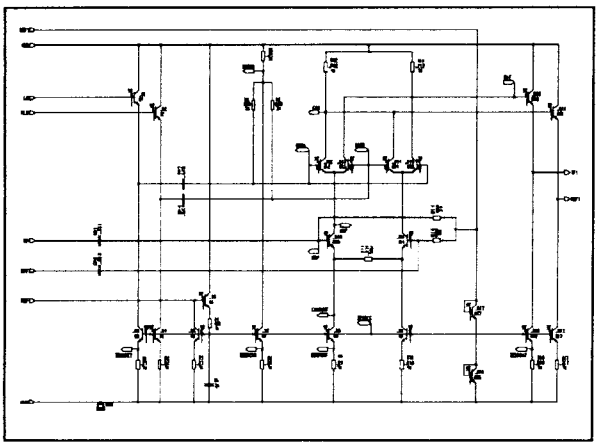


Abbildung 4: Mischer-Architektur

Der Kern des zur automatischen Dimensionierung gewählten Mischers wird durch eine leicht modifizierte Gilbert-Zelle gebildet.
Um die Sättigungsproblematik bei einer Versorgungsspannung von minimal 2.7 V etwas zu entschärfen, wird hier mit nur einem Gegenkopplungswiderstand gearbeitet. Somit entsteht kein Gleichspannungsabfall über dem Widerstand, allerdings erfordert diese Schaltungstechnik eine zweite Stromquelle. Des Weiteren beinhaltet die Schaltung Ein- und Ausgangstreiber, sowie alle benötigten Stromquellen und den zugehörigen Referenzstromspiegel. Eine weitere Besonderheit ist die AC-Kopplung zwischen den Eingangstreibern und der Schaltstufe. Dies ist ebenfalls aufgrund der geringen Versorgungsspannung notwendig und ermöglicht die gesteuerte Verteilung der Sättigungsspannungen zwischen RF- und Schaltstufe.

3.1 Dimensionierungs-Strategie

Zur automatischen Dimensionierung sind bis auf wenige Ausnahmen alle Bauteile freigegeben. Nach Abb. 5 wird folgende Strategie zugrunde gelegt: Im ersten Schritt wird eine Grobdimensionierung mit Faustformeln und symbolischen Gleichungen durchgeführt. Hier wird hauptsächlich mit numerischen Berechnungen gearbeitet, die Ergebnisse werden dann den entsprechenden Bauteilen zugewiesen. Um die Randbedingung „Maximale Stromaufnahme" einzuhalten, muß hier allerdings bereits eine Simulation durchgeführt werden.
Im zweiten Schritt, der Verifikation, werden dann die im ersten Schritt eingestellten Werte mit Simulationen überprüft und falls nötig die entsprechenden Parameter justiert.
Im dritten Schritt, der Hauptdimensionierung, erfolgt dann die Einstellung der Bauteilparameter, um die vorgegebenen Design-Ziele zu erreichen. Dabei kommen vorwiegend iterative Methoden zum Einsatz. Da hier unter anderem auch komplexere elektrische Größen zu ermitteln sind, muß auf Charakterisierungsmodule zurückgegriffen werden.

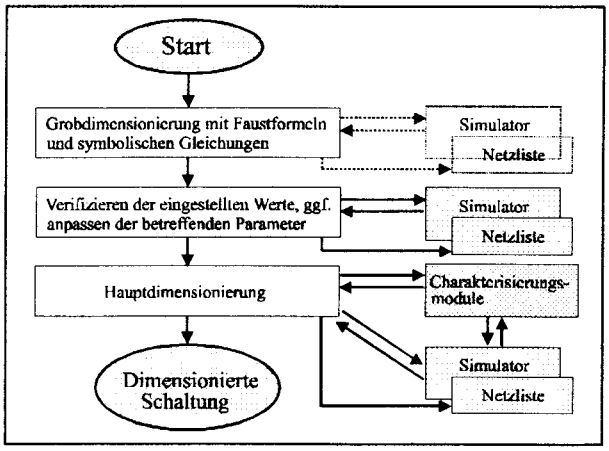


Abbildung 5: Dimensionierungs-Strategie

3.2 Grobdimensionierung

Die Reihenfolge der Berechnungen innerhalb der Grobdimensionierung orientiert sich an der zielführenden Vorgehensweise eines erfahrenen Entwicklers. Wie in Abb.6 dargestellt, wird zuerst der verfügbare Gesamtstrom möglichst effektiv auf die einzelnen Zweige der Schaltung verteilt. Der Referenzstrom des Stromspiegels wird durch den insgesamt über alle Stromquellen geführten Strom bestimmt; ein Spiegelfaktor von 20 sollte nicht überschritten werden.
Der Strom für die beiden Ausgangstreiber richtet sich nach der geforderten maximalen Ausgangsimpedanz. Dabei muß die minimale Stromreserve der Emitterfolger mindestens 10% der gesamten Stromamplitude betragen, um das Ausgangssignal nicht zu verfälschen. Dies muß bereits zu diesem Zeitpunkt der Dimensionierung mit einer Transientenanalyse überprüft werden und stellt einen Sonderfall innerhalb der Grobdimensionierung dar.

Für die Eingangstreiber gilt ebenfalls, daß die minimale Stromreserve von 10% nicht unterschritten werden darf. Dazu müssen die Emitterfolger in der Lage sein, die Kapazität zu treiben die sich an den Eingangsknoten der Schaltstufe ergibt. Diese Kapazität ergibt sich jeweils aus der Reihenschaltung eines Koppelkondensators und der Eingangskapazität zweier Transistoren der Schaltstufe. Die Eingangskapazität der Transistoren wird wiederum maßgeblich von deren Kollektorstrom bestimmt. Für die exakte Verteilung des Reststroms auf Eingangstreiber und Gilbert-Zelle wurde ein gemischtes Verfahren gewählt, welches auf einem Algorithmus und einer quadratischen Gleichung basiert. Mit der Gleichung kann für einen gegebenen Reststrom der optimale Strom für Eingangstreiber und Gilbert-Zelle bestimmt werden, wenn die Eingangskapazität der Schaltstufen-Transistoren bekannt ist. Da es sich dabei um eine stromabhängige Größe handelt, müßte der Zweigstrom bereits vor der Verteilung bekannt sein. Dies ist kausal natürlich nicht möglich. Aus diesem Grund wurde zusätzlich ein iteratives Verfahren eingesetzt.
Der Algorithmus legt dabei einen Startwert für den Strom der Gilbert-Zelle fest und berechnet mit diesem die Eingangskapazität der Transistoren. Daraufhin wird mit der quadratischen Gleichung die Stromverteilung berechnet und mit dem vorgegebenen Strom verglichen. Mit jeder Iteration wird dann der vorgegebene Strom korrigiert, bis dieser mit dem berechneten Wert übereinstimmt, dann ist die optimale Verteilung gefunden.
Abhängig von den so ermittelten Strömen für Eingangstreiber und Gilbert-Zelle werden dann die Ströme zur Erzeugung der Basisspannungen bestimmt. Nachdem alle Zweigströme bekannt sind werden die Flächenfaktoren aller Transistoren für optimale Stromdichte im Arbeitspunkt bestimmt. Die Transistoren, die als Stromquellen eingesetzt werden, erhalten die doppelte Fläche.
Die Widerstände zur Einstellung der Quellenströme werden unter Berücksichtigung der jeweiligen Basis-Emitter-Spannung des betreffenden Transistors bestimmt. Der Gegenkopplungswiderstand der RF-Stufe wird aufgrund des geforderten 1dB-Kompressionspunktes berechnet. Die Arbeitswiderstände ergeben sich dann aus Gegenkopplungswiderstand und angestrebter Konversionsverstärkung.
Anschließend wird die Basisspannung der Schaltstufe mit dem dafür vorgesehenen Widerstand so eingestellt, daß sich für RF- und Schaltstufe identische Sättigungsspannungen ergeben. Für die weitere Dimensionierung muß nun noch die Spannung an den Basen aller Stromquellen bestimmt werden.
Am Ende der Grobdimensionierung wird die Kapazität der Koppelkondensatoren so berechnet, daß sich jeweils eine Impedanz von 20 Ohm ergibt.

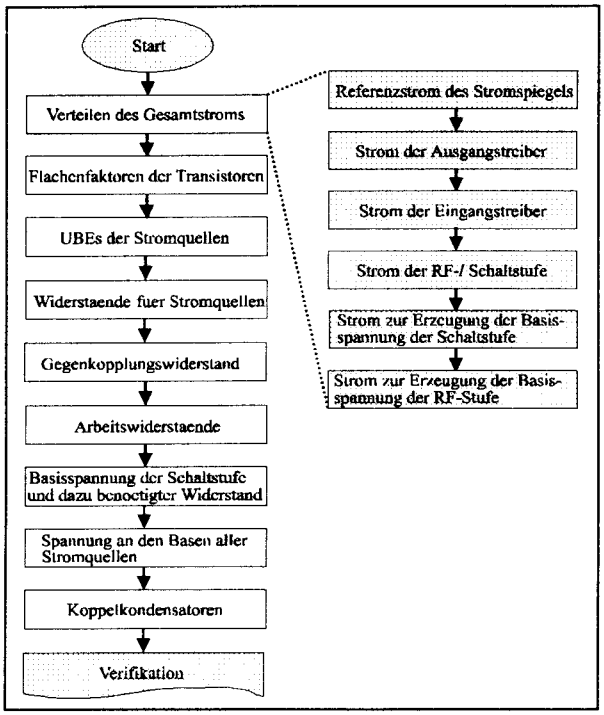


Abbildung 6: Grobdimensionierung

3.3 Verifikation

In der Verifikation werden die in der Grobdimensionierung eingestellten Werte durch Simulation überprüft und falls nötig die entsprechenden Parameter eingestellt (s. Abb.7).

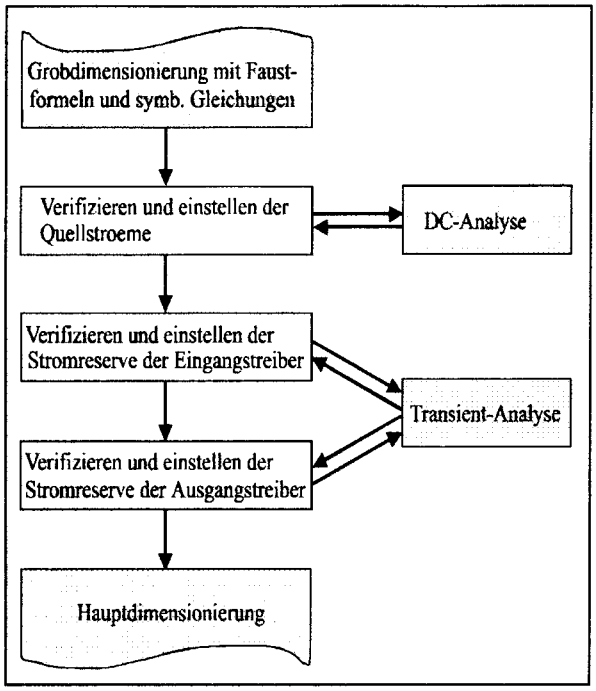


Abbildung 7: Verifikation

3.4 Hauptdimensionierung

Zur Hauptdimensionierung des Mischers wird eine Schleife abgearbeitet. Sie ist in Abb.8 dargestellt. Zu Beginn der Schleife werden jeweils alle Design-Ziele ermittelt und überprüft, ob diese bereits erreicht sind oder eines der Abbruchkriterien greift. Ein Abbruch erfolgt, wenn die maximale Anzahl an Iterationen überschritten wird oder keine weitere Verbesserung zu erwarten ist. Erfolgt kein Abbruch, so werden nacheinander die einzelnen Design-Ziele bearbeitet. Der Algorithmus für den Kompressionspunkt verändert dabei den Gegenkopplungswiderstand der RF-Stufe, bis die Vorgabe erreicht wird.
Zur Einstellung der Konversionsverstärkung werden anschließend die Arbeitswiderstände angepaßt. Das Rauschmaß wird jeweils durch die Variation der Flächenfaktoren der Transistoren der RF-Stufe minimiert.
Die einzelnen Algorithmen der Hauptdimensionierung sind so ausgelegt, daß sie ihr jeweiliges Design-Ziel entweder bereits beim ersten Aufruf erreichen oder aber überhaupt nicht. Durch die Bearbeitung eines Design-Ziels werden jedoch die beiden anderen Größen beeinflußt. Aus diesem Grund muß die Hauptschleife solange durchlaufen werden bis alle Design-Ziele direkt nacheinander erfolgreich überprüft werden können oder eines der Abbruchkriterien auftritt. Nachdem die Hauptschleife abgearbeitet ist, wird nach einer optimalen Verteilung der Sättigungsspannungen von RF- und Schaltstufe gesucht. Bis zu diesem Zeitpunkt wurde die Sättigung symmetrisch auf die beiden Stufen aufgeteilt, dies muß jedoch nicht unbedingt die beste Möglichkeit darstellen.

Abschließend werden alle Ergebnisse und Parameter in der Oberfläche ausgegeben und die Dimensionierung beendet.

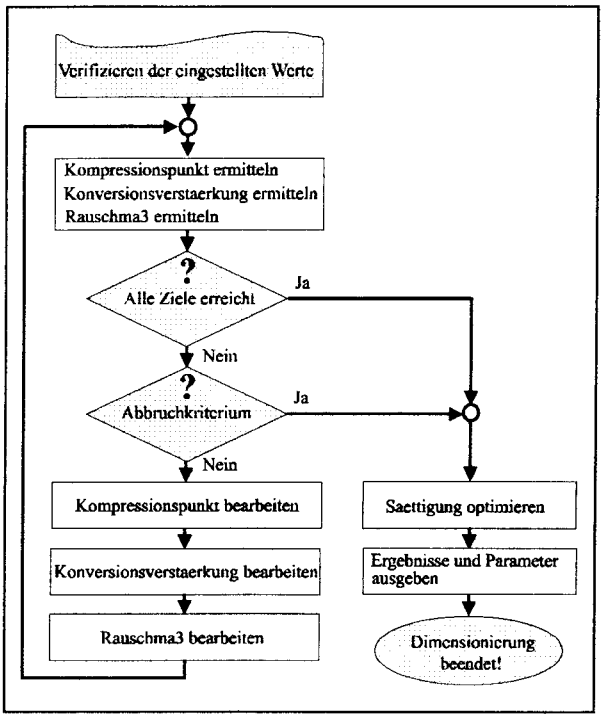


Abbildung 8: Hauptdimensionierung

4 Dimensionierung am Beispiel eines Mischers

Zur automatischen Dimensionierung des Mischers, muß der Anwender nach dem Start von ACSYN das entsprechende Modul auswählen und die zugehörige Oberfläche nach Abb.9 ausfüllen. Hier werden sowohl die Randbedingungen und Design-Ziele angegeben, als auch die notwendigen Genauigkeitseinstellungen vorgenommen. Nach erfolgter Dimensionierung werden die erreichten elektrischen Parameter der Schaltung innerhalb der Oberfläche dargestellt. Die aktuellen Werte aller freigegebenen Bauteile, sowie die wichtigsten Zweigströme der Schaltung sind ebenfalls aufgeführt.
In diesem Beispiel soll ein Down-Conversion-Mischer dimensioniert werden. Die Gesamtstromaufnahme soll 3,72 mA nicht überschreiten, die Versorgungsspannung liegt bei 2.7 V, die Amplitu-

de des externen LO-Signals beträgt 200 mV. Es soll ein RF-Signal bei 1900 MHz auf eine Zwischenfrequenz bei 400 MHz umgesetzt werden. Die Design-Ziele stellen sich wie folgt dar:

- Ausgangsimpedanz: 130 Ohm
- 1dB-Kompressionspunkt: 80 mV
- Konversionsverstärkung: 7 V/V
- Rauschmaß: 7 dB.

Das Rauschmaß berücksichtigt alle Rauschmechanismen einschließlich des Schaltrauschens [2, 6, 7]. Die erforderliche Genauigkeit ist für jedes Design-Ziel mit 2% angegeben.
Nach der Dimensionierung ergeben die folgenden elektrischen Parameter: 1dB-Kompressionspunkt: 78,9 mV, Konversionsverstärkung: 6,8 V/V, Rauschmaß: 6,8 dB.
Resultat: Alle Parameter erfüllen die Vorgaben innerhalb der angegebenen Genauigkeit von 2%, die Dimensionierung konnte also erfolgreich durchgeführt werden.

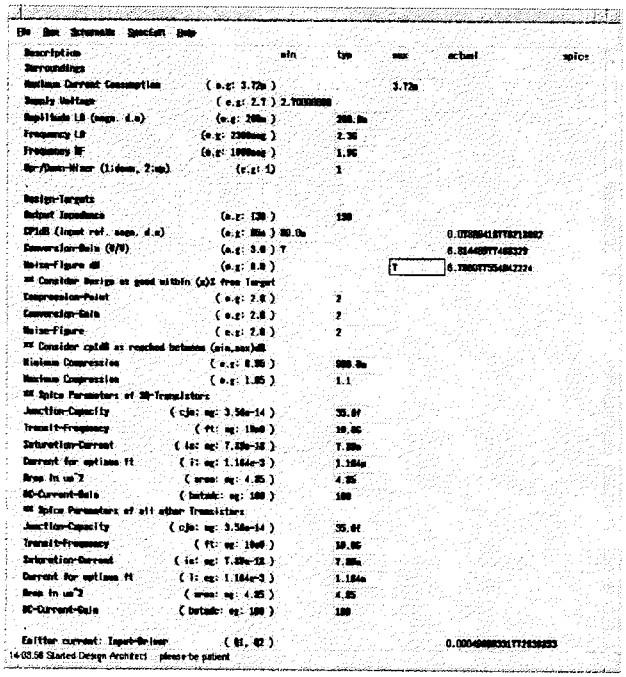


Abbildung 9: Dimensionierungs-Oberfläche

5 Charakterisierung am Beispiel eines Mischers

Die innerhalb der Dimensionierung ausgegebenen elektrischen Parameter des Mischers sind nicht ausreichend, um die entstandene Schaltung komplett zu beschreiben. Dies macht eine Charakterisierung notwendig, wie in Abb.10 gezeigt. Jede Zeile innerhalb der Charakterisierungs-Oberfläche beschreibt dabei genau eine elektrische Größe.
Für die zuvor dimensionierte Schaltung ergeben sich dabei folgende Ergebnisse:

- 1dB-Kompressionspunkt: -12,2 dBm
- IIP2: 64,6 dBm
- IIP3: -2,28 dBm
- Konversionsverstärkung: 7,0 V/V
- Kleinsignalverstärkung: 12,4 V/V
- Bandbreite: 5,0 GHz
- Rauschmaß: 6,56 dB
- Eingangsimpedanz: 247 Ohm
- Ausgangsimpedanz: 131 Ohm
- Leistungsaufnahme: 9,92 mW

Diese Werte befinden sich wiederum innerhalb der Vorgaben und entsprechen weitestgehend den bereits in der Dimensionierung ermittelten Größen.

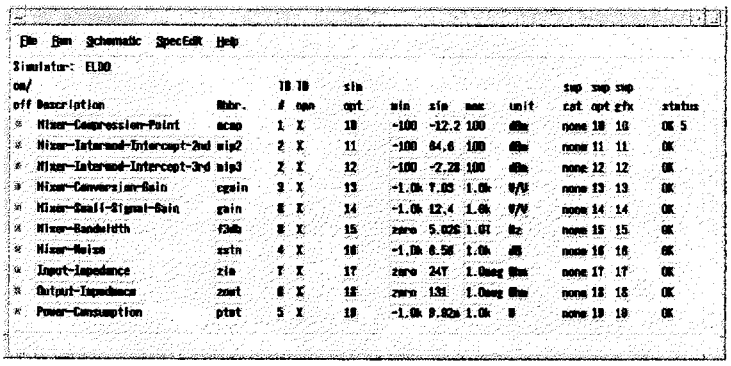


Abbildung 10: Mischer-Charakterisierung

6 Zusammenfassung

Mit ACSYN steht dem Entwickler ein leistungsfähiges und komfortables Werkzeug zur Verfügung. Dabei eignet sich ACSYN hervorragend zur automatischen Dimensionierung und zur Charakterisierung. Die Wissenskodierung wird ebenfalls gut unterstützt.
Durch die Anbindung an weit verbreitete kommerzielle Tools und die Anpassungsmöglichkeiten an neue Werkzeuge kann ACSYN universell eingesetzt werden.
Der im Rahmen einer Diplomarbeit entwickelte Algorithmus zur Mischer-Dimensionierung wurde unter verschiedensten Bedingungen ausgiebig getestet [8]. Es konnte nachgewiesen werden, daß der Algorithmus für unterschiedlichste Spezifikationen eine sehr gute Konvergenz aufweist. Die benötigte Zeit beträgt etwa drei Stunden auf einer Workstation mit 75 MHz Taktfrequenz. Dies kann im Vergleich mit einer manuellen Dimensionierung als sehr schnell eingestuft werden.

7 Referenzen

[1] Barrie Gilbert,
RF-IC-Design for wireless communication Systems, Part 2

[2] Barrie Gilbert,
Fundamentals of ACTIVE MIXERS

[3] Paul R. Gray / Robert G. Meyer,
Analysis and Design of ANALOG INTEGRATED CIRCUITS
Third Edition, ISBN: 0-471-57495-3

[4] R.Quell, G.Forster, A.Stürmer, M.Gerbershagen
ACSYN REFERENCE Manual

[5] Günter Müller,
Großsignalcharakterisierung auf der Basis impliziter numerischer Modelle, Diplomarbeit
FH-Ulm, TEMIC Semiconductor GmbH, Ulm

[6] ELDO RF Users´s Manual
Document Number 314979, April 2000

[7] Manolis T. Terrovitis,
Noise in Current-Commutating CMOS Mixers
IEEE JOURNAL OF SOLID STATE CIRCUITS
Vol. 43, No.6, June 1999

[8] Meik W. Widmer,
Einbau einer Mischer-Topologie in das Entwurfswerkzeug ACSYN, Diplomarbeit
FH-Ulm, TEMIC Semiconductor GmbH, Ulm

Virtual-Reality-Darstellung elektromagnetischer Felder in dreidimensionalen Mikrowellenstrukturen

Dipl.-Ing. (FH) Markus Feißt,
Prof. Dr.-Ing. Andreas Christ
Fachhochschule Offenburg, Badstr. 24, 77652 Offenburg
Tel. 0781/205 - 130 , Fax 0781/205 - 110

Untersuchungen haben gezeigt, daß der Mensch ein Vielfaches an Informationen in Form von visuellen Eindrücken, im Gegensatz zur textuellen Darstellung, verarbeiten kann. Mit Hilfe des numerischen Feld-Simulationsprogramms F3D können Mikrowellenstrukturen auf die Wechselwirkung mit elektromagnetischen Feldern untersucht werden. Das Programm F3D2VRML stellt die Ergebnisse in einer dreidimensionalen Virtual-Reality-Darstellung (VR) dar.
Damit ist es dem Betrachter möglich, mehr Informationen aufzunehmen, da die Informationen mit Formen und Farben im dreidimensionalen Raum visualisiert werden.

1 Einführung

1.1 Das Programm F3D

Das Programm F3D ist ein Simulationsprogramm, das mit Hilfe der Finite-Differenzen Methode das elektromagnetische Verhalten von Mikrowellenstrukturen simuliert [1]. Dabei wird die zu simulierende Struktur in kleine, nicht äquidistante Quader zerlegt (Abb. 1.1).

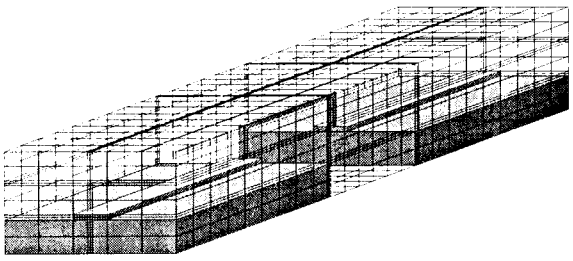


Abb. 1.1 : Unterteilung einer Mikrowellenstruktur in Quader

Das Ergebnis der Simulation beschreibt die Wechselwirkung der elektromagnetischen Felder mit der physikalischen Struktur. Es gibt zwei Möglichkeiten das Ergebnis zu „betrachten". Zum Einen gibt es die Möglichkeit, das Ergebnis als Streumatrix darzustellen. Zum Anderen kann das elektrische Feld (für eine Frequenz) betrachtet werden. Dabei wird als Resultat der Simulation für jeden Quader ein elektrischer Feldvektor berechnet und in einer Datei gespeichert.

1.2 Was ist VR (vrml,*.wrl)

VR ist die Abkürzung für Virtual Reality (Virtuelle Realität). Darunter versteht man im Allgemeinen, daß ein Welt aus Objekten auf eine Art dargestellt wird, die dem Betrachter einen dreidimensionalen Eindruck ermöglicht. Ein wesentliches Merkmal ist der vom Betrachter frei wählbare Blickwinkel. Der Betrachter ist in der Lage sich frei durch die „VR-Welt" zu bewegen und die Objekte von jeder gewünschten Seite zu betrachten.
Es gibt eine Vielzahl von Programmen zur 3D-Darstellung. Die meisten benutzen allerdings proprietäre Darstellungs- bzw. Speicherformate.
Das, in diesem Beitrag vorgestellte Programm F3D2VRML, verwendet als Dateiformat VRML (*.wrl). Die Gründe sind, daß das VRML-Format ein im Internet sehr häufig verwendetes Format ist, daß Plugins für Browser erhältlich sind und fast jedes Programm zum Bearbeiten von 3D-Darstellungen einen VRML-Importfilter bereitstellt.

1.3 Verbindung zwischen F3D und VR

Nach erfolgter feldnumerischen Simulation mit F3D liegen die Werte der Feldverteilung vor. Das Resultat wird in einer ASCII-Datei abgelegt. Eine typische Simulation mit 200000 Quadern erzeugt eine circa zehn MByte große ASCII-Datei, die sich einer direkten Betrachtung entzieht. Mit Hilfe der Software F3D2VRML kann nun eine VR-Welt generiert werden, die durchwandert und aus allen Winkeln betrachtet werden kann.

2 Kurzbeschreibung des Programms F3D

Das Programm F3D ist in der Programmiersprache FORTRAN geschrieben und läuft auf einer Apollo 700 HP Workstation.
Das Programm löst dynamische elektromagnetische Feldprobleme in dreidimensionalen, berandeten Strukturen. Über angeschlossene Wellenleiter (Hohlleiter, Streifenleitungen, Koaxialleiter etc.) wird die Struktur im Innern elektrisch erregt. Die Lösung wird durch die Quantisierung der Maxwellschen Gleichungen im Frequenzbereich (FDFD-Methode) erreicht [1].

2.1 Beschreibung der generierten Daten und deren Aufbau

Das Programm F3D generiert mehrere Dateien, von denen allerdings hier nur zwei Typen von Bedeutung sind. Wichtig ist zum Einen die Datei, welche die Mikrowellenstruktur-Daten enthält, zum Anderen die Datei, welche die Felddaten enthält. Da die Felddaten jeweils für eine Frequenz in eine separate Datei geschrieben werden, kann das Ergebnis einer Simulation mehrere Felddaten-Dateien umfassen. Von diesen Dateien ist für die Visualisierung allerdings immer nur eine relevant. Denn die dreidimensionale Darstellung der Mikrowellenstruktur und des elektrischen Feldes kann zur Zeit nur für eine Frequenz dargestellt werden.
Die Datei für die Mikrowellenstruktur beinhaltet die folgenden Informationen :

- Allgemeine Schlüsselwörter
- Die Anzahl der Quader in X-, Y- und Z-Richtung.
- Die realen Abmessungen der Quader in X-, Y- und Z-Richtung, da die Quader nicht äquidistant sind (Höhe, Breite, Tiefe).
- Eine Liste, die den Materialnamen die Materialkoeffizienten zuweist (ε, μ).
- Die Informationen über die einzelnen Quader. Es wird dabei der Name des entsprechenden Materials gespeichert.

3 Beschreibung des Programms F3D2VRML

3.1 Vor-/Nachteile von Java

Die Entscheidung, das Programm in Java zu erstellen, hat mehrere Gründe. Java ist Plattform unabhängig. Dies ist für den Einsatz in heterogener Rechnerumgebung (sowohl Windows NT als auch HP UX als Betriebsysteme)vorteilhaft. Ein weiterer Vorteil ist, daß aus Java-Programmen Applets generiert werden können, bzw. Teile des Programmcodes (in Form von Objekten) ohne Probleme in Applets übernommen werden können.
Ein Nachteil entsteht dadurch, daß ein Java Programm voraus setzt, daß eine Virtuelle Maschine (VM) korrekt installiert wurde. Außerdem ist das Einlesen von Dateien in Java, im Vergleich zu C/C^{++}, relativ langsam. Diese Tatsache fällt beim F3D2VRML-Programm besonders auf, da sehr große Dateien eingelesen werden müssen.

3.2 Entscheidung für ein GUI

Eine ebenfalls wichtige Entscheidung war die einer Konsole oder einer grafischen Benutzeroberfläche (GUI).
Der Vorteil einer Konsolenanwendung (ähnlich einer BATCH-Datei) ist, daß bei mehrfacher Ausführung die Eingabe über eine Datei gesteuert und somit die Generierung der Daten automatisiert werden kann.
Ein großer Nachteil ist allerdings, daß bei einer falschen Eingabe die gesamte Prozedur wiederholt werden muß.
Ein großer Vorteil der grafischen Oberfläche ist, daß dem Benutzer alle möglichen Optionen angezeigt werden, und er bei einer falschen Eingabe diese verbessern kann. Um aber bei gleichen Aufgaben nicht immer wieder alle Parameter und Optionen neu einstellen zu müssen, wurde ein Batchmodus (keine grafische Ausgabe) implementiert, der über eine Makro-Datei gesteuert wird.

3.3 Datenstruktur innerhalb des Programms F3D2VRML

Die Daten werden aus der Datei eingelesen und in einem internen-Format gespeichert. Alle Daten werden in entsprechenden Objekten gespeichert. Die Daten welche die Mikrowellenstruktur betreffen werden in einem „Struktur"-Objekt und alle Daten, die das elektrische Feld betreffen werden in einem „Feld"-Objekt gespeichert.
Das „Struktur"-Objekt beinhaltet einen 3-dimensinalen Array, wobei jeder Wert des Arrays das Material eines Quaders repräsentiert. Auf diese Weise läßt sich problemlos jeder Quader ansprechen. Außerdem beinhaltet das Objekt noch eine Liste der Materialien und der dazugehörigen Materialkoeffizienten (ε, μ). Des weiteren enthält das Objekt noch 3 Arrays, welche die realen Dimensionen speichern.
Das „Feld"-Objekt beinhaltet drei dreidimensionale Arrays, jeweils ein dreidimensionaler Array für die X-, Y- bzw. die Z-Komponente des elektrischen Feldes.
Die Ergebnisdaten der feldnumerischen Simulation mit F3D, die in einem anderen Format vorliegen, müssen eingelesen und dementsprechend umgewandelt werden.
Eine Änderung des Datenformates in F3D war nicht sinnvoll, da F3D die Daten in dieser Form benötigt.

3.4 Mikrowellenstruktur in Polygone aufteilen

In VRML können Flächen nur in einer Ebene dargestellt werden. Dabei muß die Ebene in der sich die Fläche befindet nicht parallel zur XY-, XZ- oder YZ-Ebene sein.
Da die simulierte Mikrowellenstruktur aus Quadern zusammengesetzt ist kommen darin nur Flächen die parallel zu einer Koordinatensystemebene sind vor.
Jeden Quader in der VR-Welt darzustellen würde bedeuten, daß die 3D-Struktur sehr unübersichtlich würde.
Einen besseren Eindruck gewinnt der Betrachter, wenn nur die Grenzflächen zwischen zwei Quadern mit unterschiedlichen Materialien dargestellt werden. Eine Grenzfläche zwischen zwei Objekten würde in diesem Fall aus vielen kleinen Rechtecken zusammengesetzt sein, was der Betrachter nicht bemerkt. Da sehr viele Flächen eine sehr große Rechenbelastung darstellen (Rendering wird sehr zeitaufwendig), reagiert die Darstellung allerdings damit sehr träge auf Veränderungen. Deshalb wurde ein Verfahren implementiert, welches die Eckpunkte der Grenzflächen liefert. Auf diese Weise kann eine Fläche in einer Ebene als Polygon dargestellt werden.

3.4.1 Problematik der Strukturfindung und der Lösungsansatz

Die erste Problematik ist, einen geeigneten Algorithmus zur Erkennung der Eckpunkte zu finden.
Hierbei wurde auf Verfahren der digitalen Bildverarbeitung zurückgegriffen [2]. Als Erstes wird die Struktur auf Grenzflächen hin untersucht. Dabei werden die Ebenen, die parallel zu den Koordinatensystem-Ebenen sind untersucht. Das Problem wird zuerst einmal vom dreidimensionalen in den zweidimensionalen Raum verlegt (Objekterkennung → Flächenerkennung). Das Ergebnis wird in einem Layer (zweidimensionale Matrix) gespeichert (Abb. 3.1).

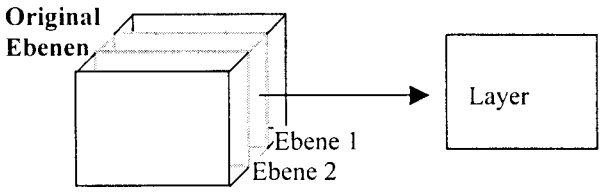


Abb. 3.1 : Erstellen eines Layers aus zwei Ebenen der Struktur

Wird ein Materialunterschied zwischen Quadern in benachbarten Ebenen (Ebene1 und Ebene2) festgestellt, wird im Layer der Wert eins abgespeichert. Sind beide Materialien gleich, wird im Layer eine Null abgelegt.

Ein Beispiel für einen erstellten Layer zeigt Bild 3.2.

0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0
0	0	0	0	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Abb. 3.2 : Beispiel eines Layers

Dieser Layer wird nun mit einer drei mal drei Matrix (Abb. 3.3) gefiltert, um die Eckpunkte zu bestimmen.

1	3	5
7	11	13
17	19	23

Abb. 3.3 : Filtermatrix

Das Ergebnis zeigt Abbildung 3.4

Abb. 3.4 : Layer, nach erfolgter Filterung

0	0	0	23	42	59	59	59	36	17
0	0	0	41	66	90	90	90	54	24
0	23	42	77	74	99	99	99	58	25
0	36	66	90	98	99	99	99	58	25
0	41	74	99	99	99	99	99	58	25
0	41	74	99	99	99	99	99	58	25
0	41	74	99	99	99	99	99	58	25
0	18	32	40	40	40	40	40	22	8
0	5	8	9	9	9	9	9	4	1
0	0	0	0	0	0	0	0	0	0

Der Filteralgorithmus setzt voraus, daß das Material mindestens zwei Reihen und zwei Spalten umfaßt. Diese Einschränkung ist im praktischen Einsatz unerheblich, da ein Material mit nur einer Reihe bzw. Spalte, aufgrund der numerischen Berechnung zu ungenauen Ergebnissen der Simulation führt und deshalb nicht sinnvoll ist.
Nachdem die Eckpunkte gefunden wurden, müssen diese der Reihe nach gespeichert werden. Hierzu wird die linke untere Ecke (Zahlenwert 32) gesucht. Danach bewegt man sich gegen den Uhrzeigersinn um das Polygon, bis man wieder am Anfangspunkt angekommen ist. Hierbei helfen die für die Ecken charakteristischen Zahlenwerte. Der Startpunkt sowie die Umlaufrichtung wurden dabei willkürlich gewählt. Wichtig ist allerdings, daß die Reihenfolge der Punkte im Umlauf stimmt. Auf diese Weise ist klar, daß der erste Punkt mit dem zweiten, der zweite mit dem Dritten... und der letzte mit dem Ersten verbunden ist. Die Speicherung, welcher Punkt mit welchem Punkt verbunden ist, entfällt.

Damit ist das Problem der Strukturfindung noch nicht gelöst. Betrachtet man Bild 3.5, stellt man fest, daß gemäß der VRML-Standards diese Fläche nicht dargestellt werden kann.

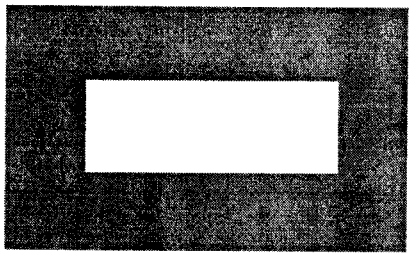


Abb. 3.5 : Nicht VRML-Standard konforme Darstellung

Das Problem ist, daß diese Fläche mehrfach zusammenhängend ist. Solche Figuren sind dem VRML-Standard unbekannt und somit nicht darstellbar. Wird trotzdem solch eine Figur erzeugt, ist die Darstellung nicht mehr vorhersagbar, die Punkte werden zu einer willkürlichen Figur verbunden.
Der hier verwendete Lösungsansatz zerteilt diese Fläche in folgende zwei Flächen (Abb. 3.6).

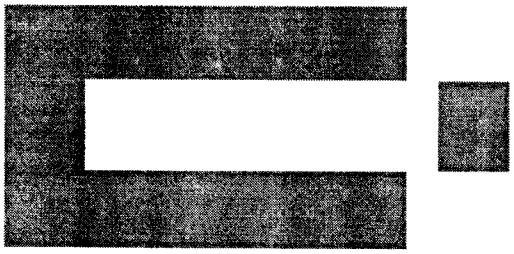


Abb. 3.6 : Zerlegung der Fläche in zwei einfach zusammenhängende Flächen

Dies bedeutet aber, daß der Layer bevor er gefiltert werden kann, so aufbereitet werden muß, daß nur einfach zusammenhängende Flächen vorkommen.
Der Originallayer der nach der Grenzflächenuntersuchung kreiert wurde, wird Zeile für Zeile in einen neuen übertragen (Layer A). Die übertragenen Einsen (Materialunterschied zwischen benachbarten Quaderebenen) werden aus dem Originallayer gelöscht (d.h. zu Null gesetzt). Wird ein Übergang von Eins auf Null innerhalb des Originallayers festgestellt, wird der restliche Teil der Zeile des Layers A automatisch mit Nullen aufgefüllt. Die restliche Zeile des Originallayers bleibt daraufhin unverändert.
Auf diese Weise ist sichergestellt, daß nur einfach zusammenhängende Flächen im Layer A vorkommen, der gefiltert und wie zuvor besprochen weiterverarbeitet werden kann.
Der Originallayer muß wiederholt untersuchen werden, da in diesem gegebenenfalls noch weitere Flächen vorhanden sind. Ist kein Materialunterschied mehr im Layer vorhanden (nur Nullen im Layer), kann die Untersuchung für die nächsten Grenzflächen gestartet werden.

3.5 Darstellung des E-Feldes

Das elektrische Feld wird in Form von Pfeilen dargestellt. Dabei spiegelt die Größe des Pfeils die Intensität des elektrischen Feldes wieder. Die Richtung, in der das Feld wirkt wird durch die Orientierung im dreidimensionalen Raum und die Pfeilspitze dargestellt.
Weil die Quader nicht äquidistant sind, wurde ein eigenes „Raster" für das elektrische Feld eingeführt. Obwohl die Einführung eines eigenen Rasters für das elektrische Feld eine Interpolation des elektrischen Feldes nötig macht, hat dies doch Vorteile.
Der Betrachter ist eine Darstellung mit äquidistanten Abständen der Pfeile gewohnt. Ist dies nicht der Fall, interpretiert der Betrachter dies als Feldstärkenunterschiede. Dieses verfälscht die Aussage über das elektrische Feld.
Ein weiterer Vorteil ist, daß der Betrachter für einen „ersten Eindruck" ein gröberes Raster wählen kann, was die Anzahl der Pfeile verringert. Damit gestaltet sich die Darstellung für den ersten Eindruck übersichtlicher.

Nach der Einführung des Rasters werden die „Rastermittelpunkte" berechnet, denn an diesen Punkten wird das elektrische Feld in der VR-Welt dargestellt.
Die Werte des elektrischen Feldes sind nur an den Mittelpunkten der Quaderkanten bekannt [1]. Da Raster- und Quaderkantenmittelpunkt im Allgemeinen nicht identisch sind, muß eine Interpolation durchgeführt werden.
Hierzu muß der Quader, in welchem der Rastermittelpunkt liegt, ermittelt werden. Mit Hilfe der benachbarten Quader wird der Wert des elektrischen Feldes linear interpoliert.
Abbildung 3.7 zeigt die prinzipielle Funktionsweise der Interpolation.

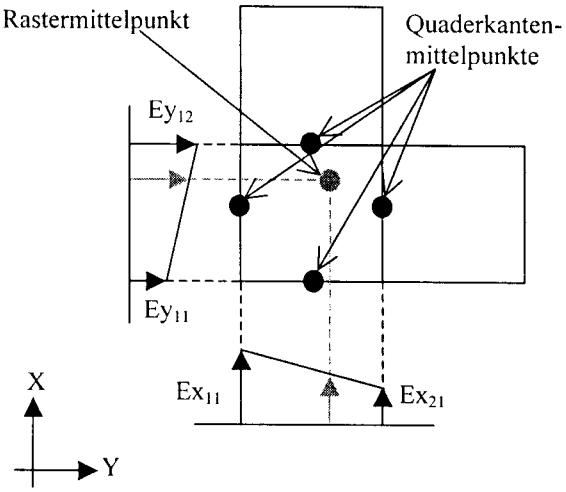


Abb. 3.7 : Prinzipielle Funktionsweise der Interpolation

Die Interpolation wird in allen drei Richtungen (X,Y und Z) durchgeführt. Die Quader, die dabei zur Interpolation benötigt werden, hängen von der Position des Rastermittelpunktes ab. Es sind insgesamt acht Fälle zu unterscheiden.
Es ist keinesfalls gewährleistet, daß die Materialien der benachbarten Quader identisch sind. Wird die Interpolation über zwei Quadern unterschiedlichen Materials durchgeführt, müssen die Materialkoeffizienten berücksichtigt werden.
Ein Sonderfall stellt ein Quader aus Metall dar. Die zur Grenzfläche parallelen E-Feld-Komponenten sind Null und die senkrechten E-Feld-Komponenten müssen gemäß dem Prinzip der Spiegelladungen berechnet werden.
Abbildung 3.8 zeigt die Interpolation für die E-Feld-Komponente in Y-Richtung im dreidimensionalen Raum.

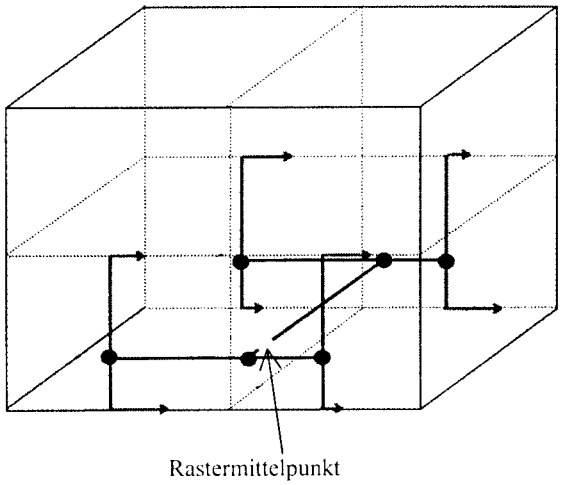


Abb. 3.8 : Interpolation der E-Feld-Komponente in X-Richtung im 3D Raum

Die Interpolation erfolgt in drei Stufen (schwarze Linien). Die schwarzen Punkte markieren hierbei Zwischenergebnisse, der graue Punkt das Ergebnis. Nach Interpolation auch der X- und Z-Komponente ist der E-Feld-Vektor im Rastermittelpunkt bekannt.

4 Zusammenfassung

Mit Hilfe des Programmes F3D können Mikrowellenstrukturen in bezug auf die Wechselwirkung zwischen Material und elektromagnetischen Wellen untersucht werden. Die Zusammenhänge lassen sich, durch die große Datenmenge aber vom Benutzer schwer erkennen.
Vom Menschen können durch die visuelle Wahrnehmung mehr Informationen aufgenommen werden (Farbe, Form, räumliche Anordnung, Entfernungen, usw.). Deshalb werden die Simulationsergebnisse mit Hilfe des Programme F3D2VRML in eine räumliche Darstellung umgesetzt.

5 Ausblicke

Es ist vorgesehen, das Programm zu erweitern, um auch das magnetische Feld darstellen zu können.
Ein weiterer Punkt ist das Verarbeiten mehrerer Dateien mit Felddaten um Animationen zu erzeugen. Damit kann gezeigt werden, wie sich das elektrische Feld in Abhängigkeit der von der Zeit oder der Frequenz verändert.
Um die Darstellungsergebnisse auch remote zugänglich zu machen, ist ein Web-Interface geplant. Dies wird durch den Einsatz von Java erleichtert.

Literatur

[1] Three-Dimensional Finite-Difference Methode for the Analysis of Microwave-Device Embedding

A. Christ and H. L. Hartnagel
IEEE Trans. Microwave Theory Tech. Vol. 35, No. 8, August 1987

[2] Theorie und Anwendung der digitalen Bildverarbeitung

Prof. Dr. A. Erhardt-Ferron

[3] Documentation for the Java Program F3D2VRML (Projectwork)

P. Luchner, R. Stader, I. Kolemanov, M. Feißt
Fachhochschule Offenburg, 2000

fh
AALEN

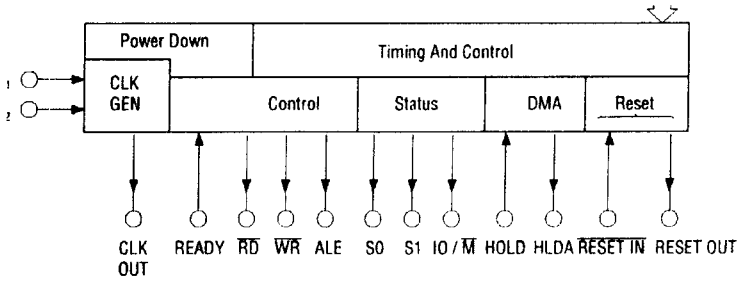
Simulationsmodell des 8085 in VHDL

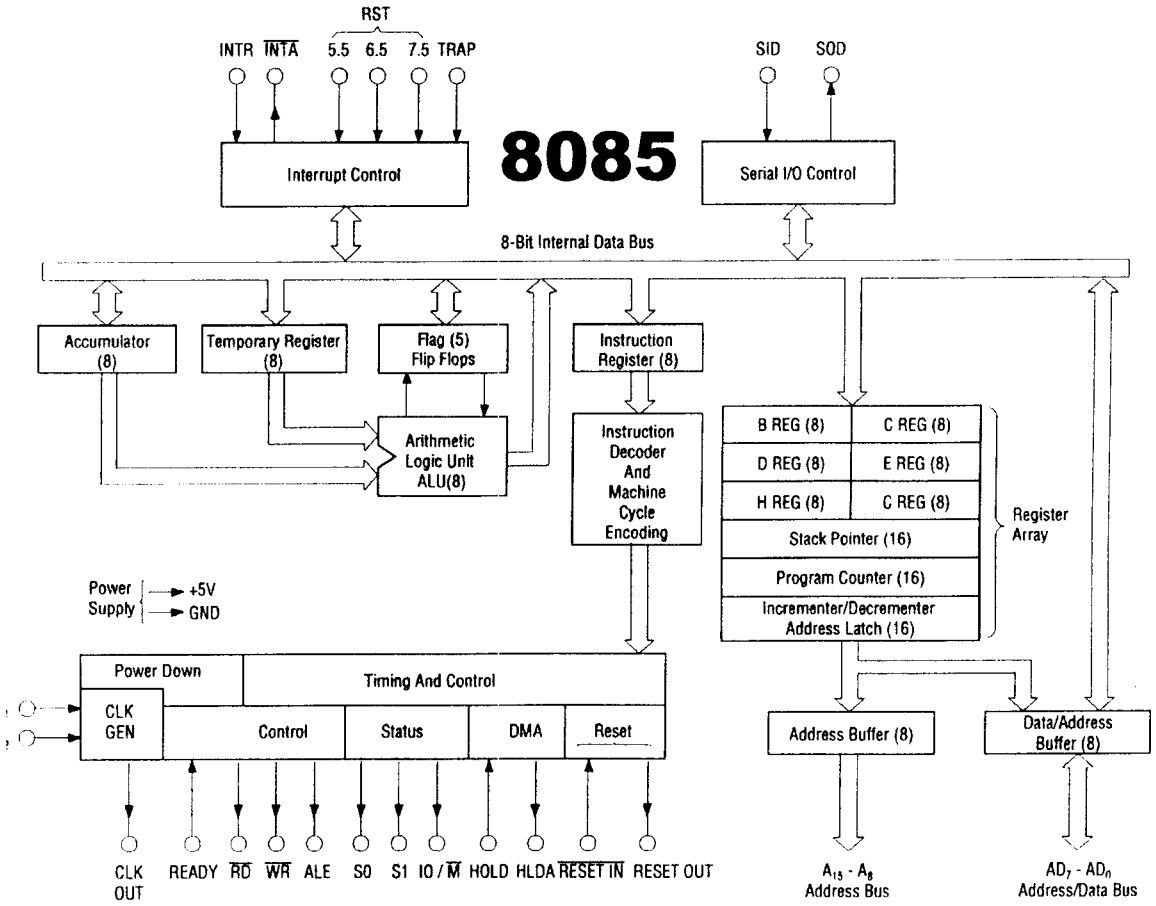
Diplomarbeit von Markus Fidelak

Zeit- und Ablaufsteuerung

- Mit **welchen** Daten muss zu **welcher** Zeit **was** geschehen?

- **Verwaltung** des Adress- und Datenbusses, sowie die **Steuerung** der Peripherie.





Der Befehl

- **ACI** (Addieren einer Konstanten zum Accu mit Carry)
- (A) + (byte2) + (C) > A

- **CNC** (Call if not Carry)

(PC) > (SP)
- (SP) – 2 > SP

(byte3), (byte2) > PC

- 2 Operationszyklen
  - FR
- 7 Operationsschritte

- 2 / 5 Operationszyklen
  - SR o. SRRWW
- 9/18 Operationsschritte

Operationszyklen (Maschinenzyklen)

Typen

- Opcodefetch **F** und **S**
- Memory **R**ead und **W**rite
- E/A Read und Write **I** und **O**
- Bus Idle **B**

Information vom Befehlsdekoder

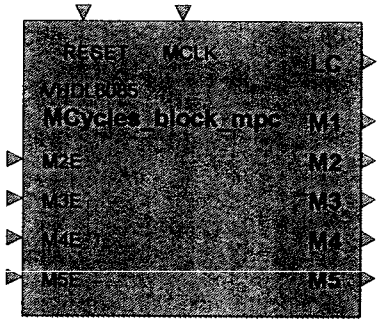
MaschinenzyklusEnable
M2E, M3E, M4E und M5E

MaschinenzyklusTyp
M2R, M2W ... M3O ... M5W

Zusammenhang

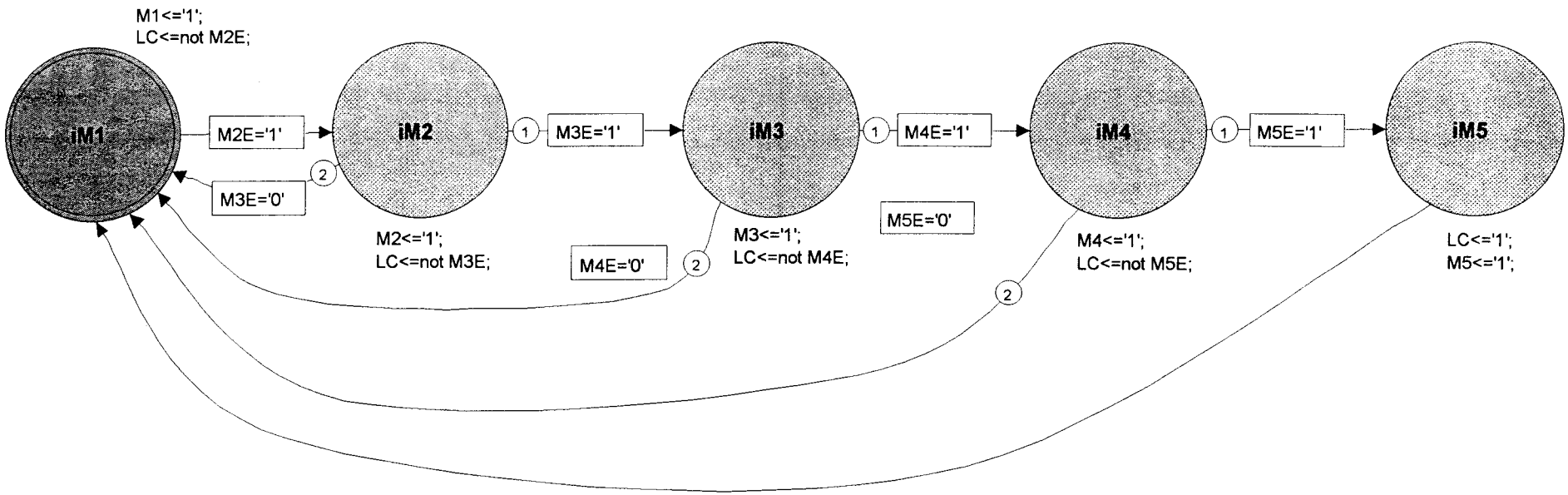
F: benötigt **4** Operationsschritte
S: hat **6** Operationsschritte
R, W, I, O und B: haben **3** Operationsschritte

Maschinenzyklusautomat



Package List

ieee std_logic_1164
ieee numeric_std



Operationsschritte (T-States)

T-State-Typen

normale: T1 – T6
spezielle: T_{Wait}, T_{Reset}, T_{Halt}, T_{Hold}

Informationen

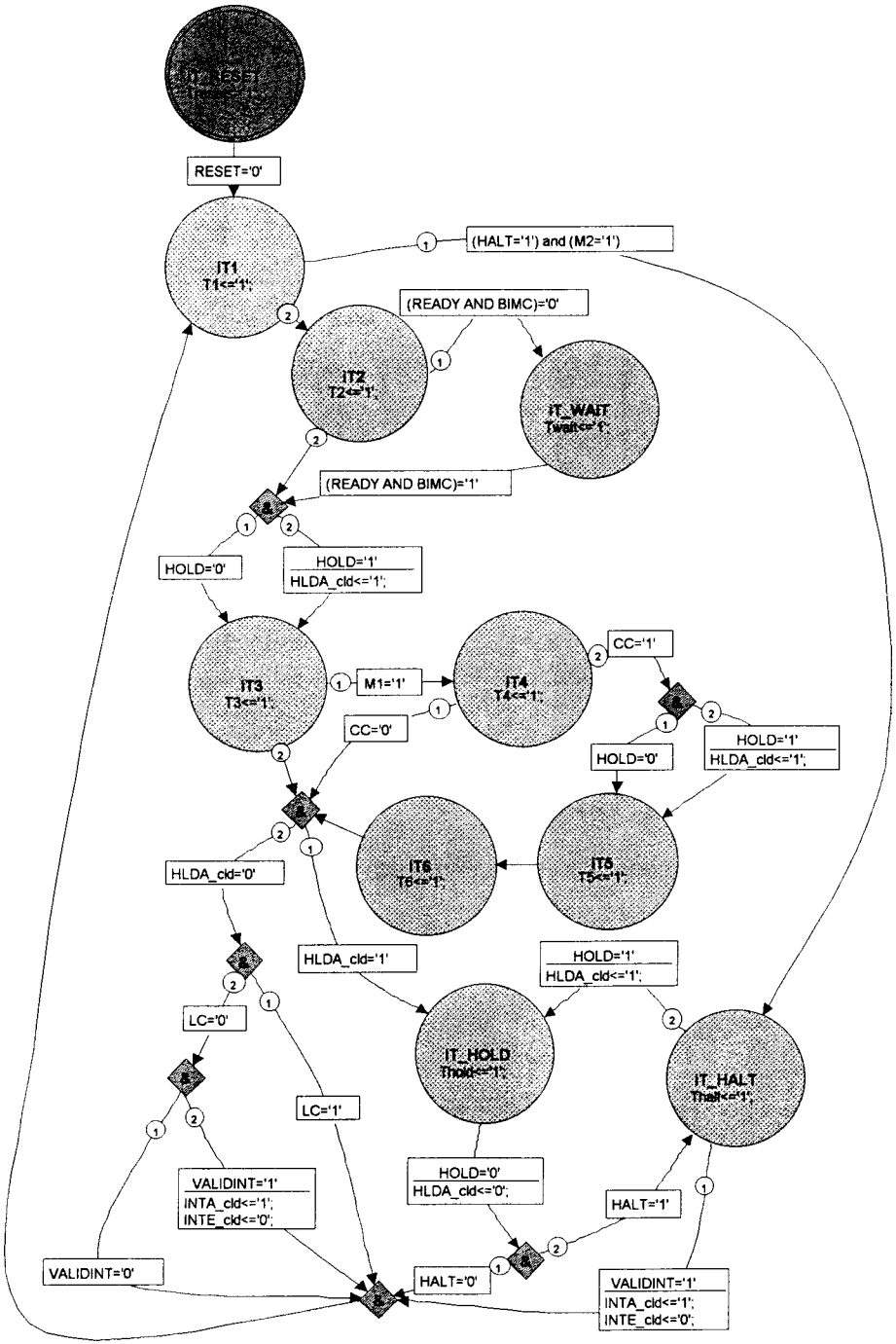
Maschinenzyklusautomaten
CC, M1, M2 und

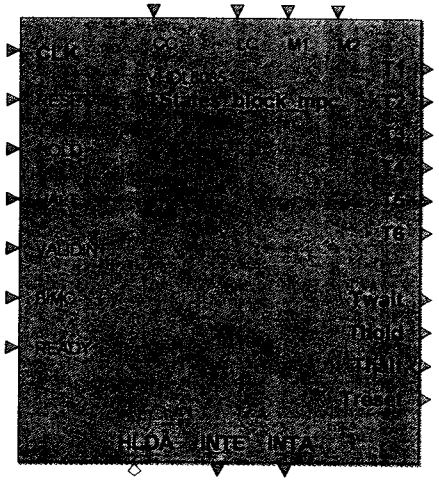
Außen
READY, HOLD, RESET

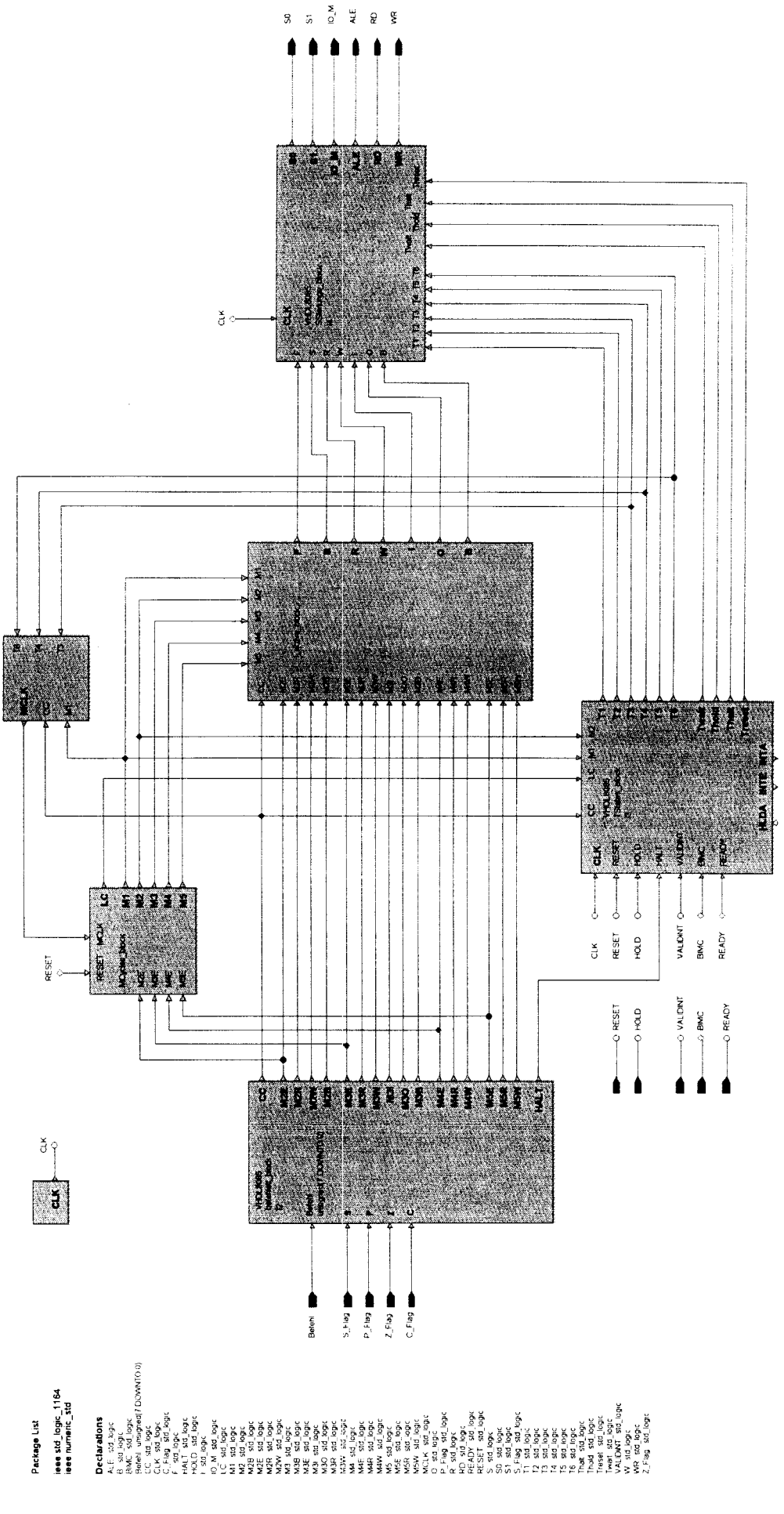
Zusammenhang

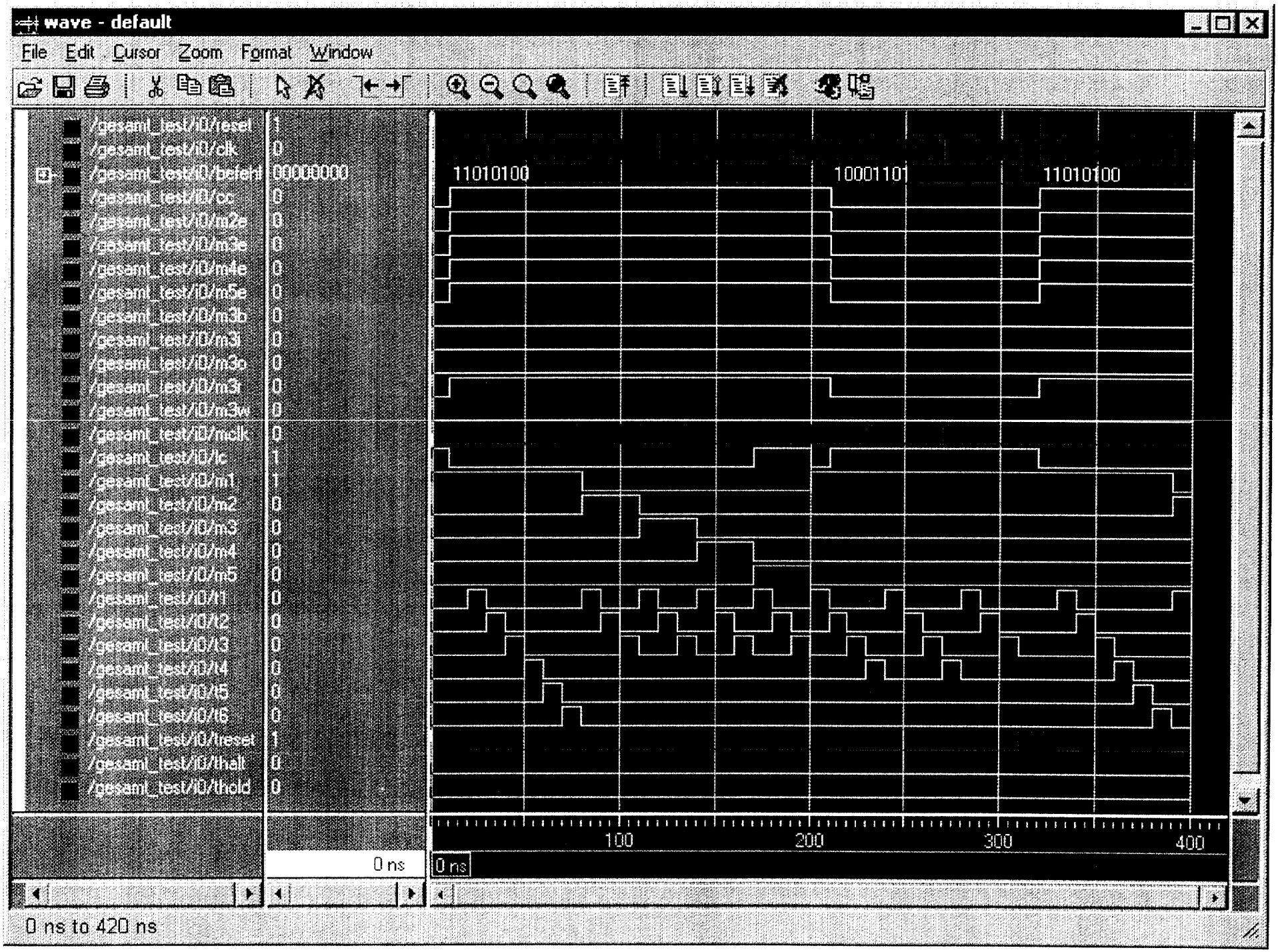
Ein Clock-Signal entspricht einem T-State
Ein Operationszyklus setzt sich aus T-States zusammen.

T-State-Automat






S0
S1
IO_M
ALE
RD
WR
CLK
RESET
HOLD
HLDA
READY
BMC
VALIDNT
Beleh
S_Flag
P_Flag
Z_Flag
C_Flag
Package List
ieee.std_logic_1164
ieee.numeric_std
Declarations


wave - default
File Edit Cursor Zoom Format Window
/gesamt_test/i0/reset
/gesamt_test/i0/clk
/gesamt_test/i0/befehl
/gesamt_test/i0/cc
/gesamt_test/i0/m2e
/gesamt_test/i0/m3e
/gesamt_test/i0/m4e
/gesamt_test/i0/m5e
/gesamt_test/i0/m3b
/gesamt_test/i0/m3i
/gesamt_test/i0/m3o
/gesamt_test/i0/m3t
/gesamt_test/i0/m3w
/gesamt_test/i0/mclk
/gesamt_test/i0/ic
/gesamt_test/i0/m1
/gesamt_test/i0/m2
/gesamt_test/i0/m3
/gesamt_test/i0/m4
/gesamt_test/i0/m5
/gesamt_test/i0/t1
/gesamt_test/i0/t2
/gesamt_test/i0/t3
/gesamt_test/i0/t4
/gesamt_test/i0/t5
/gesamt_test/i0/t6
/gesamt_test/i0/treset
/gesamt_test/i0/thalt
/gesamt_test/i0/thold
1
0
00000000
11010100
10001101
11010100
100
200
300
400
0 ns
0 ns
0 ns to 420 ns

13,56 MHz Transceiver nach ISO 14443-A

Josef Zimmermann
Prof. Dr.- Ing. Dirk Jansen
Fachhochschule Offenburg, ASIC-Design-Center
Badstr. 24, 77652 Offenburg
☎ :0781/205-274, Fax: 0781/205-174

Im Folgenden wird die Entwicklung eines Modems zur induktiven Datenübertragung nach ISO 14443 für kontaktlose Chipkarten beschrieben [4].

Ein System besteht aus zwei Komponenten, einer kontaktlosen Chipkarte (Transponder) und einem Erfassungs- oder Lesegerät. Der Transponder ist vollständig von dem Lesegerät abhängig, sowohl die Energie, als auch der Systemtakt des Transponders werden durch das Lesegerät übertragen. Der Datenaustausch erfolgt allerdings in beide Richtungen.

Das Lesegerät sendet die mit einem modifizierten Millercode codierten Daten durch eine Modulation der Amplitude des Trägers.

Der Transponder codiert die Daten nach dem Manchestercode. Diese werden dann mit Hilfe eines Lastmodulationsverfahrens mit Hilfsträger gesendet.

1. Einleitung

Die ISO 14443 ist eine Norm aus dem Bereich der RFID (Radio Frequenz Identifikation [3]). Sie spezifiziert die kontaktlose Kommunikation zwischen Chipkarten und deren Lesegeräten. Diese Art der Chipkarten erfreut sich wachsender Akzeptanz. Die Vorteile dieses kontaktlosen Systems liegen klar auf der Hand. So haben Nässe und Schmutz keinen Einfluss auf die Funktion. Auch ist die Handhabe dieser Karten komfortabel, z.B. beim Einsatz in bargeldlosen Kassensystemen verbleibt die Karte in der Brieftasche. Diese wird dann einfach auf das Lesegerät gelegt. Sobald eine Karte in den Einflussbereich eines Lesegeräts gelangt, wird eine Kommunikation aufgebaut. Der maximale Abstand zwischen Karte und Lesegerät dieses Systems beträgt 10 cm.

Auf Grund dieser Vorteile und der Tatsache, dass es sich hier um eine aktuelle Norm handelt, lag es nahe, eine Schnittstelle dieses Prinzips auch für den FHOP zu realisieren.

2. Grundlegende Funktionsweise

Ein Lesegerät besteht typischerweise aus einem Hochfrequenzmodul, einer Kontrolleinheit und einem Koppelelement. Über dieses wird eine Frequenz von f_T = 13,56 MHz ausgestrahlt. Die Chipkarte besteht ebenfalls aus einem Koppelelement. Die Eingangskapazität des Mikrochips und die Leiterschleife bilden einen Schwingkreis, welcher auf die Trägerfrequenz des Lesegerätes abgestimmt ist. Die beiden Geräte sind induktiv gekoppelt.

Aus der Trägerschwingung von f_T = 13,56 Mhz kann im Mikrochip der Karte der Systemtakt 106 kHz abgeleitet werden (13,56 Mhz /128). Die Energieversorgung der Chipkarte (Transponders) ist durch Gleichrichtung der Trägerschwingung des Lesegerätes gewährleistet. Der Datenaustausch erfolgt bidirektional nach dem Master- Slave Prinzip, wobei das Lesegerät die Rolle des Master übernimmt.

3. Datenübertragung

3.1. Lesegerät -> Transponder

Bei diesem Typ wird eine 100 % Amplitudenmodulation (on/off keying) zur Datenübertragung von Lesegerät zur Karte verwendet. Hierzu wird die Trägerschwingung mit einer codierten Bitfolge im Lesegerät getastet. In Abbildung 1 ist eine Sendefolge des Lesegerätes als Spannungsverlauf an der Antenne des Lesegeräts dargestellt.

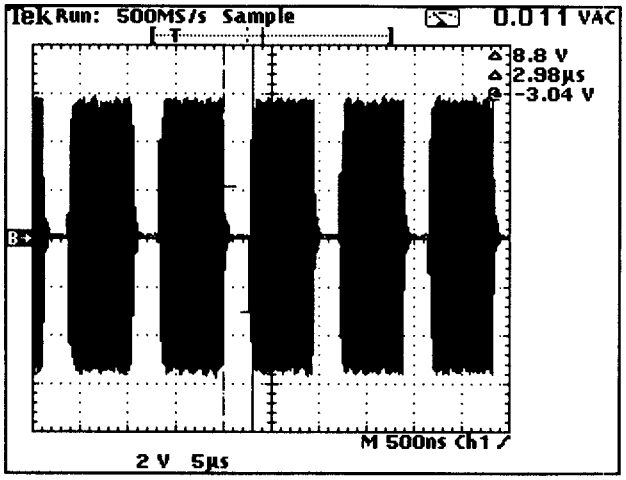


Abbildung 1: getasteter Träger

Zur Codierung der zu sendenden Bits wird ein modifizierter Millercode eingesetzt. Ein Vorteil dieses Codes besteht darin, dass die Austastlücken eine maximale Dauer von nur 2,6 µs haben. Dadurch ist eine konstante Energieversorgung des Transponders aus dem HF - Feld auch während des Sendevorgangs durch das Lesegerät gewährleistet.

Die Decodierung der Bits nach modifiziertem Millercode erfolgt nach folgendem Schema (siehe Abbildung 2).

Jede logische 1 der Bitfolge hat eine Austastlücke ab der Mitte eines Taktes. Eine einzelne logische 0 hat keine Veränderung Ab der zweiten Null hat jede weitere eine Austastlücke zu Beginn eines Taktes.

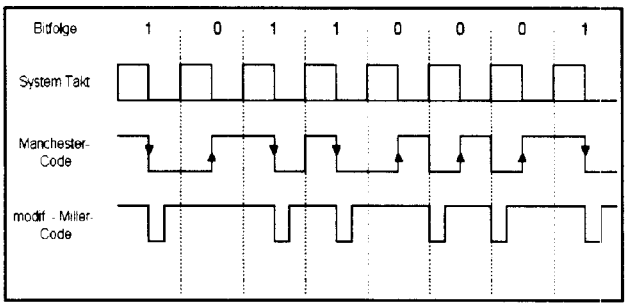


Abbildung 2: benutzte Codes

3.2. Transponder -> Lesegerät

Die Datenübertragung von der Chipkarte zum Lesegerät erfolgt mit Hilfe eines Lastmodulationsverfahren mit Hilfsträger. Die Hilfsträgerfrequenz beträgt dabei f_{HT} = 847 kHz (13,56 Mhz /16). Die Modulation des Hilfsträgers erfolgt durch Austastung mit dem Manchester codierten Datenstrom. Mit diesem getasteten Hilfsträger wird die Trägerschwingung in der Amplitude moduliert.

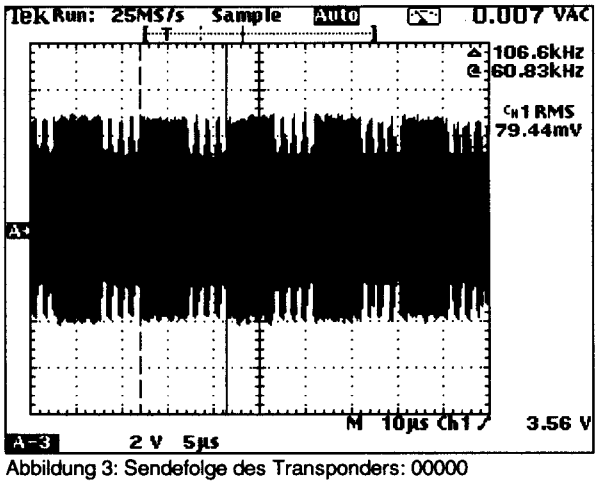


Abbildung 3: Sendefolge des Transponders: 00000

Bei der Manchester Codierung (Bi Phase Code) wird jede logische 1 durch eine negative Flanke in der Halbbit-Periode, eine 0 durch eine positive Flanke in der Halbbit-Periode dargestellt (siehe Abbildung 2).

4. Realisierung

In Abbildung 4 ist das realisierte Modem als Blockschaltbild dargestellt. Links ist der analoge Frontend als Schwingkreis angedeutet, in welchem die Signale zur digitalen Weiterverarbeitung analog aufbereitet werden müssen. Der Tiefpass dient hier als Hüllkurvendemodulator, um den modifizierten Millercode zu extrahieren. Ein Schmitttrigger dient zur Aufbereitung der 13,56 MHz Schwingung in ein digitales Signal, welches dann intern auf die Bitfrequenz 106 kHz heruntergetaktet wird.

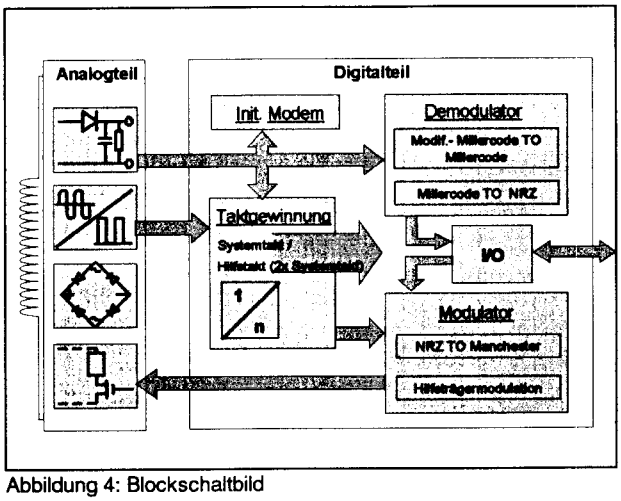


Abbildung 4: Blockschaltbild

Der Brückengleichrichter dient der Energieversorgung des Transponders.

Der unterste Block des Analogteils zeigt einen Schalter mit Widerstand. Diese Baugruppe dient dem Sender zur Modulation der Trägerfrequenz.

Auf die Blöcke des Digitalteils - Taktgewinnung, Demodulation und Modulation - wird im Folgenden genauer eingegangen. Der Block Init. Modem dient zur Generierung des internen Reset-Signals, der I/O Block der Kommunikation mit dem FHOP.

4.1. Taktgewinnung

Zur Taktgewinnung wird ein Synchronteiler benutzt. Dieser teilt die f_T = 13,56 MHz auf den Systemtakt 106 kHz herunter (13,56 / 2^7).

Allerdings stellen die Austastlücken in der Trägerschwingung (siehe Abbildung 1) ein Problem dar. Da in diesen Momenten kein Takt anliegt, halten die Flip-Flops den letzten Zustand.

Dieses Problem lässt sich mit der Hilfe des modifizierten Millercodes lösen. Untersucht man diesen, stellt man fest, dass eine Ähnlichkeit zu dem 212 kHz Takt besteht, welcher dem doppelten Systemtakt entspricht. In Abbildung 5 sehen wir den Systemtakt, darunter ist die Bitfolge: 100011 mit dem modifizierten Millercode dargestellt. Eine Affinität zu dem 212kHz Takt ist deutlich zu erkennen. Eine Austastlücke des modifizierten Millercodes dauert genauso lange wie ein HIGH Zustand des betreffenden Taktes. Desweiteren fällt jede Austastlücke exakt auf ein High des 212 kHz Taktes.

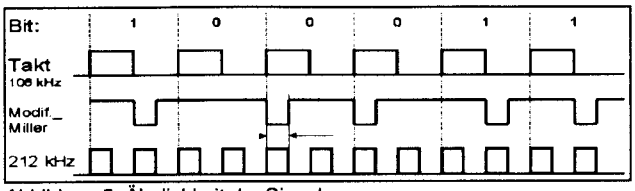


Abbildung 5: Ähnlichkeit der Signale

Der modifizierte Millercode wird als Reset- Signal für die State-Machine des Taktteilers benutzt. Für die Dauer des Resets wird ein logisches High ausgegeben, daraufhin wird synchron weiter geteilt bis zum nächsten Reset u.s.w.

In Abbildung 6 ist das Simulationsergebnis der Teilerschaltung dargestellt.

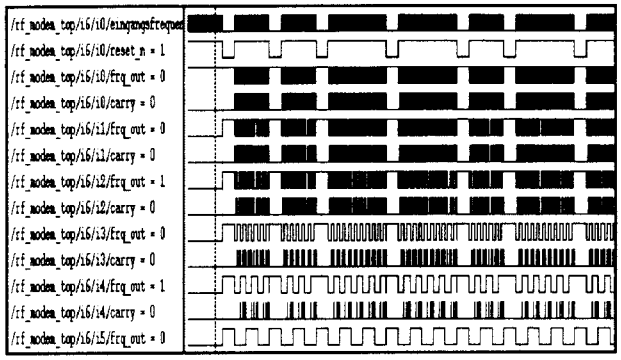


Abbildung 6: Simulationsergebnis

Oben sehen wir die getastete Eingangsfrequenz darunter den modifizierten Millercode als Reset Signal, ganz unten ist der generierte 212 kHz Takt zu erkennen. Dazwischen sind die Zwischenergebnisse der Teilung dargestellt.

4.2. Demodulation

Um das empfangene Signal zu decodieren, wird zuerst der modifizierte Millercode in den Millercode umgewandelt.

Der Millercode hat für logisch 1 eine beliebige Flanke in der Halbbitperiode. Eine 0 nach einer 1 wird durch eine Verlängerung vorhergehenden Pegels in die nächste Bitperiode dargestellt. Jede weitere 0 hat einen Flankenwechsel zu Beginn des Taktes.

Die Umwandlung geschieht, indem zunächst die fallenden Flanken des modifizierten Millercodes, dann die steigenden ausgewertet werden. Die zwei

Varianten des Millercodes sind somit um ¼ Takt zueinander verschoben (Abbildung 7).

Auf diese Weise sind definierte High oder Low Zustände zur Auswertung gewährleistet. Dadurch kann ein Entstehen von Spikes vermieden werden.

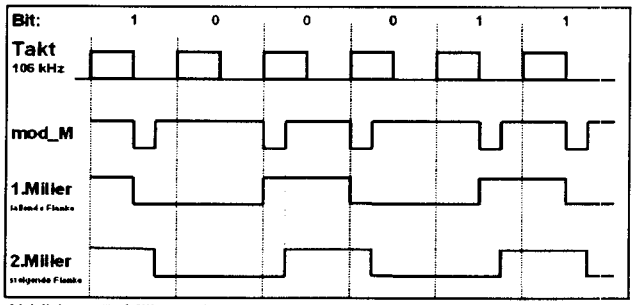


Abbildung 7: Millercode

Für die Abfrage, ob ein Flankenwechsel innerhalb der Bitperiode (für logisch 1) stattgefunden hat, wird der um ¼ Takt verschobene Millercode ausgewertet. Zur Abfrage, ob es einen Flankenwechsel zu Taktbeginn gegeben hat, muss die Flanke der ersten Variante des Millercodes ausgewertet werden.

In der Abbildung 8 ist der prinzipielle Vorgang der Decodierung des Millercodes in NRZ dargestellt. Als Takt zur Decodierung dient der 212 kHz Takt.

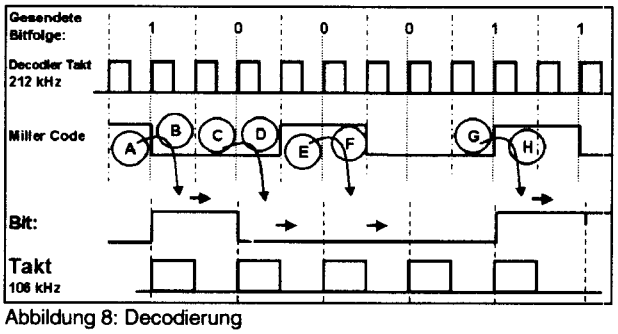


Abbildung 8: Decodierung

Ist der Millercode auf High wie bei „A", wird in dem folgenden State „B" abgefragt, ob der Millercode Low ist. Ist dies der Fall, wird ein High ausgegeben. Anderenfalls wird ein Low ausgegeben wie bei „E" und „F". Im darauf folgenden State wird der jeweilige Zustand gehalten, um auf die Datenrate von 106 kBit/s zu kommen.

Ist der Millercode auf Low wie bei „G", wird in dem folgenden State „H" abgefragt, ob der Millercode High ist. Trifft dies zu, wird ein High ausgegeben. Trifft dies nicht zu, wird ein Low ausgegeben wie bei „C" und „D". Im nächsten State wird der letzte Zustand beibehalten.

4.3. Modulation

In dem Modulator wird der von der Schnittstelle kommende Bitstream zunächst in einen Manchester Code (Abbildung 2) gewandelt.

Abbildung 9 zeigt den prinzipiellen Verlauf der Codierung nach Manchester. Zur Codierung wird der selbe Takt wie zur Decodierung verwendet.

Eine logische 1 wird in eine High Low Folge gewandelt, eine 0 in eine Low High Folge.

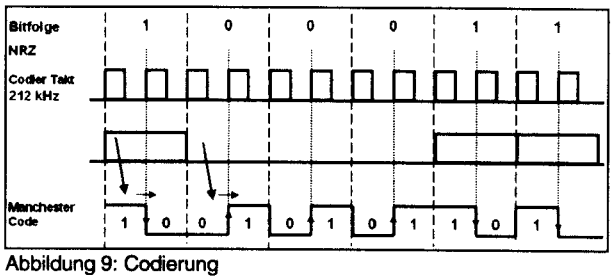


Abbildung 9: Codierung

Nach der Codierung in Manchester wird dieses Signal benutzt, um den Hilfsräger zu tasten. In Abbildung 10 ist dieser Vorgang dargestellt. Dieser so modulierte Hilfsträger wird dann als serielles Signal auf einen Schalter (MOSFET) gegeben.

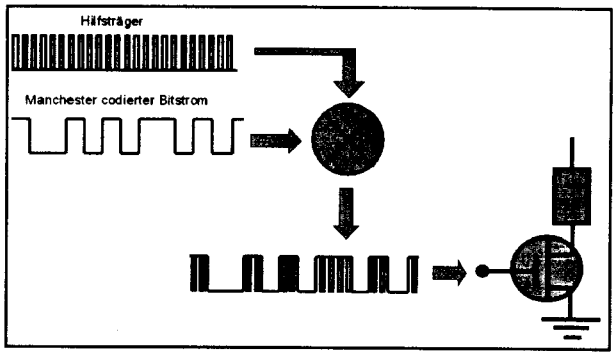


Abbildung 10: Modulation

Dadurch wird dem Schwingkreis im Takt des modulierten Hilfsträgers ein Lastwiderstand parallel geschaltet.

Die positiven Auswirkungen dieser Sendeart werden erst in der Betrachtung des Frequenzbereiches deutlich.

Durch die Lastmodulation mit Hilfsträger entstehen zwei Seitenbänder im Abstand der Hilfsträgerfrequenz um die Sendefrequenz des Lesegerätes (Abbildung 11). Die relevante Information für das Lesegerät steckt in den Seitenbändern.

Dabei ist zu beachten, dass nicht die Position der Hilfsträger relativ zu dem Träger die Information darstellt, sondern die Tastung des Hilfsträgers (Zeitbereich). Dies wird in der Abbildung 3 deutlich, welche eine Nullfolge darstellt.

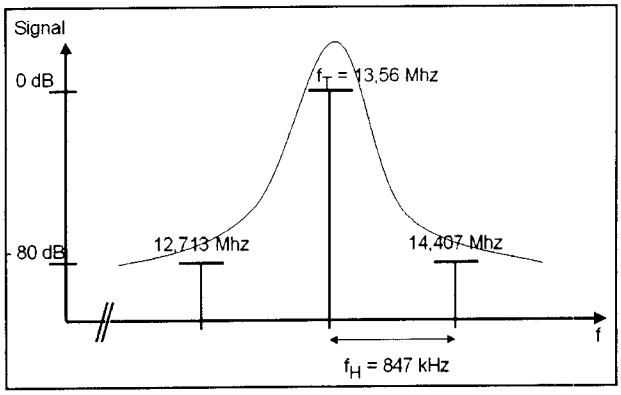


Abbildung 11: Frequenzbereich

Die Daten können dann empfängerseitig durch Filterung und Demodulation gewonnen werden.

5. Literatur:

[1] Tietze, Schenk:

Halbleiterschaltungstechnik, Springer Verlag

[2] Dirk Jansen:

Vorlesungsskript Elektronische Schaltungstechnik

[3] Klaus Finkenzeller:

RFID – Handbuch, Hanser Verlag

[4] Mifare: Produkt Specification Rev. 1.3

DIRECT SYNTHESIZED GMSK GENERATOR

Prof. Gisbert Glasmachers
Analog- und Hochfrequenztechnik
Fachhochschule Heilbronn, Max-Planck-Strasse 39, 74081 Heilbronn

Einführung

Die Übertragung von Digitalsignalen auf Funkkanälen erfordert eine möglichst gute spektrale Ausnutzung der verfügbaren Bandbreite. Zu diesem Zweck werden geeignete Modulationsverfahren wie z.B. GMSK eingesetzt. Zur Erzeugung der Steuersignale und zur Modulation des Trägers werden bisher mehrere Schaltkreise in analoger und digitaler Technik eingesetzt. Das Ziel des hier vorgestellten Projektes ist es, eine Schaltung aus Generator und Modulator in einem einzigen integrierten Baustein in reiner Digitaltechnik vorzustellen. Dadurch ergeben sich für die Entwicklung von Datenfunk-Geräten deutliche Vorteile

Digitale Phasenmodulation mit GMSK

Digitale Modulation bedeutet in vielen Fällen, dass nur ein Bit, das die Werte „Null" oder „Eins" annehmen kann, als Information auf einen HF-Träger aufmoduliert wird. Dies kann durch Änderung der Amplitude, der Frequenz oder der Phase erfolgen. Man spricht dann von Amplituden-Umtastung, Frequenzumtastung oder Phasenumtastung. Die Amplitudenumtastung scheidet von vornherein aus dem Wettbewerb aus, weil sie mit der höchsten Störanfälligkeit behaftet ist. Die Frequenzumtastung ist am leichtesten verständlich. Die nachfolgende Abbildung zeigt einen möglichen Zeitverlauf:

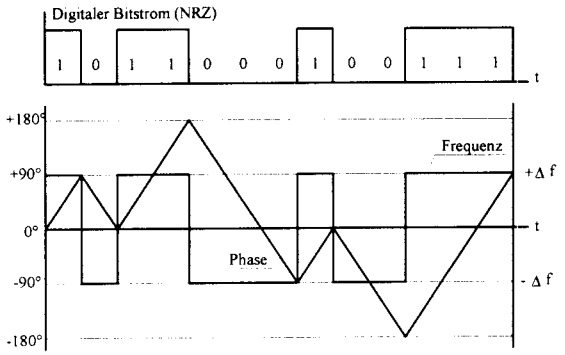


Abb. 1 Frequenz und Phase als Funktion der Zeit bei Frequenzumtastung mit MSK

Bei der Frequenzumtastung entspricht der Verlauf der Trägerfrequenz als Zeitfunktion unmittelbar dem digitalen Bitstrom. Die Phase als Integral der Frequenz verläuft dann rampenförmig. Wenn man nun festlegt, dass grundsätzlich jedes Bit mit einer Phasenänderung von 90° korreliert sein soll, so spricht man von „Minimum Shift Keying", kurz MSK. Die obige Abbildung 1 zeigt genau diesen Fall. Ein wesentlicher Nachteil von MSK besteht in der relativ großen Bandbreite, die ein Übertragungskanal bereitstellen muss. Im Bereich der Funktechnik kann man sich den Luxus unnötig großer Kanalbreiten einfach nicht leisten. Deshalb versucht man, das Spektrum bei gleicher Informationsdichte schmaler zu gestalten Hier ein Vergleich, wobei jeweils die Leistungsdichte als Funktion der Frequenz aufgetragen ist.

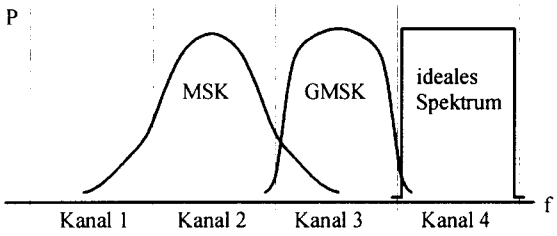


Abb. 2 Spektrale Energieverteilung bei MSK und GMSK im Vergleich zum idealen Spektrum

Die spektrale Verteilung sollte einem Rechteck möglichst nahe kommen, damit die volle Energie in den eigenen Sendekanal kommt, ohne den Nachbarkanal zu stören. Aus der Abbildung 2 erkennt man, dass bei MSK noch eine erhebliche Ausweitung des Spektrums bis in den Nachbarkanal auftritt. Durch Filterung des Digitalsignals mit Gauß-Filtern kommt man vom "Minimum Shift Keying" (MSK) zum "Gauß Minimum Shift Keying", kurz GMSK. Bei GMSK kommt man dem Ziel des idealen Spektrums ein Stück näher. Die nachfolgende Abbildung 3 zeigt den Unterschied zwischen den beiden Signalen. Bei GMSK sind die Ecken im Phasenverlauf sinusähnlich ausgerundet. Je geringer die Bandbreite des Gauß-Filters ist, umso stärker werden die Ecken abgeschliffen. Dies führt zu einem schmaleren Spektrum mit steileren Flanken. Kennzeichnend hierfür ist das BT-Produkt, wobei B für die Bandbreite des Gauß-Filters und T für die Bitdauer steht.

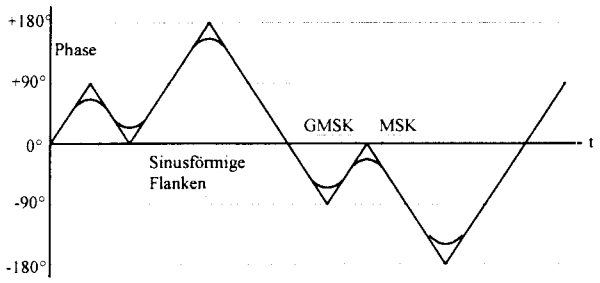


Abb. 3 Unterschied zwischen MSK und GMSK

Die europäischen Mobilfunk-Netze verwenden GMSk als Modulationsstandard mit einem BT-Produkt von BT = 0,3 und einer Datenrate von 270 kBit pro Sekunde

Die einfachste Konfiguration zur Erzeugung eines GMSK-Signals, die in diesem Vortrag vorgeschlagen wird, erfordert demnach die Zuführung oder direkte Erzeugung eines Trägerssignals, am besten aus einem Quarzoszillator, ferner einen Dateneingang und einen Takteingang. Alle Komponenten zur Erzeugung des Ausgangssignals, einschließlich der Filter und Modulatoren, befinden sich in einem einzigen Chip, der direkt GMSK-Signale erzeugt.

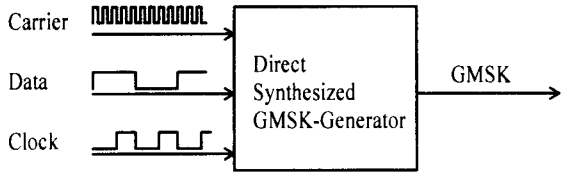


Abb. 4 Signale an einem GMSK-Generator

Viele Entwickler und viele Anwender im Bereich der Daten-Funkübertragung können mit der Handy-Technik nichts anfangen. Zum einen sind die meisten der dort verwendeten Chips nicht auf dem freien Markt verfügbar, zum anderen passen sie oft nicht zu den veränderten Anforderungen. Es gibt aber einen großen und ständig zunehmenden Markt für die Daten-Funkübertragung. Daher ist es sinnvoll, einen Chip zu entwickeln, der universell verwendbar ist. Dies gilt vorallem in Hinblick auf die der privaten Nutzung zugänglichen Frequenzbereiche. Hier sind in erster Linie die Bereiche bei 433 Mhz, 860 Mhz und 2,4 Ghz zu nennen. Für diese Bänder sind Spezifikationen bezüglich der abgestrahlten Leistung und der Kanalbreite vorgegeben. Die im Mobilfunknetz verwendete Bitrate von 270 kBaud ist für die meisten Anwendungen in Hinblick auf die benötigte Bandbreite zu groß.

Konventionelle Erzeugung eines GMSK-Signals.

Bevor ich auf die Einzelheiten meiner Idee eingehe, möchte ich den bisher üblichen Weg zur Erzeugung eines GMSK-Signals kurz vorstellen.

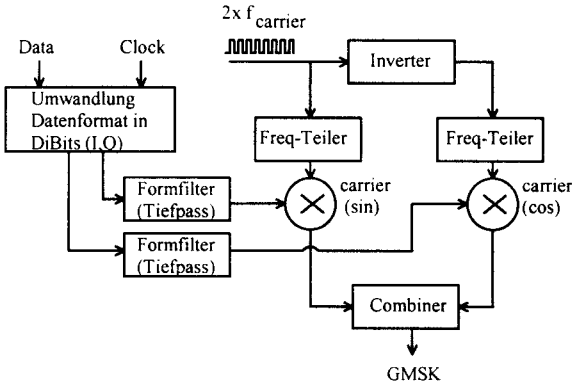


Abb 5 Schaltungsstruktur bei der konventionellen Erzeugung von GMSK

Die Erzeugung eines Zeigers mit beliebig einstellbarer Phase erfolgt durch die Addition zweier ortogonaler Komponenten, die als I (In Phase) bzw Q (Quadratur Phase) bezeichnet werden. Man kann sie auch als Sinus bzw Cosinus-Schwingung auffassen. Um sie zu generieren, wird allgemein die doppelte Trägerfrequenz zugeführt. Nach einer Invertierung des Signals und anschließender Frequenzteilung ergeben sich zwei um 90° versetzte Trägerschwingungen.

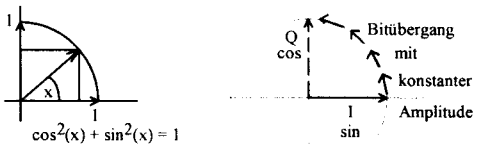


Abb. 6 Bitübergang mit 90° - Winkel bei konstanter Amplitude

Um die Amplitude unabhängig von der Phase zu halten, wird die allseits bekannte trigonometrische Beziehung:

$$\sin^2(x) + \cos^2(x) = 1$$

ausgenutzt. Dazu wird das Datensignal so aufgearbeitet, dass zwei Komponenten mit der passenden Filterung den beiden Mischern zugeführt werden.

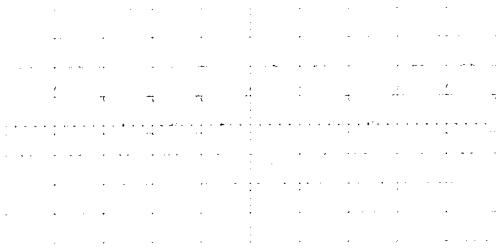


Abb 7. Oszillogramm der Steuersignale I und Q zur Erzeugung von MSK

Die Abbildung 7 zeigt ein Beispiel für den Verlauf der beiden Steuersignale. Die spitzen Ecken verdeutlichen, dass es sich um ein MSK-Signal handelt. Mit einer entsprechenden Filterung ergibt sich der folgende Verlauf:

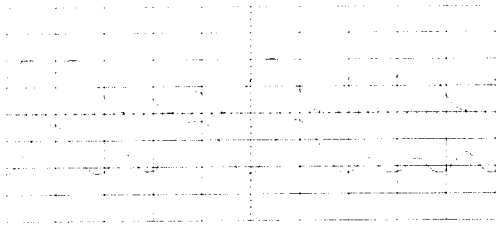


Abb. 8 Oszillogramm der Steuersignale I und Q zur Erzeugung von GMSK

Für die Erzeugung solcher Signale aus einem digitalen Datenstrom ist ein erheblicher Aufwand erforderlich. Es müssen nämlich zwei Forderungen erfüllt werden:

- Reduzierung der Bandbreite durch Gauß-Filter. Diese müssen üblicherweise als digitale Filter realisiert werden, da man mit passiven Bauteilen keine Gauß-Funktion $G(\omega) = G_0 * \exp-(*\frac{\omega}{\omega_0})^2$ erzeugen kann.

- Einhaltung einer konstanten Amplitude bei der Bildung des Quadrates der orthogonalen Komponenten.

Eine spezielle Steuerung sorgt dafür, dass diese Forderungen ständig realisiert werden.

Man erkennt, dass ein erheblicher Aufwand erforderlich ist, um aus einem digitalen Logik-Signal einen HF-Träger mit GMSK-Modulation zu erzeugen. Mit dem neuen Konzept kann dieser Aufwand erheblich reduziert werden.

Konzept eines vollintegrierten GMSK-Generators

Das hier vorgeschlagene Konzept sieht folgende Merkmale vor:

Direkte Erzeugung eines Trägers mit konstanter Amplitude auf einer Zwischenfrequenz

Dazu ist im einfachsten Fall nur ein Quarzoszillator erforderlich. Dieser liefert selbstverständlich eine konstante Amplitude. Anstelle der doppelten Trägerfrequenz wird die einfache Frequenz zugeführt. Die beiden Mischer entfallen. Die Signalaufbereitung von gefilterten I- und Q-Komponenten wird vereinfacht.

Phasenverschiebung entsprechend einer GMSK-Funktion

Die Phasenverschiebung, die ja gleichzeitig die Modulation darstellt, wird ausschließlich durch Laufzeit-Verzögerung innerhalb der integrierten Schaltung realisiert.

Phasenverschiebung in integrierter Technik

Man kann eine Phasenverschiebung durch Laufzeiten von Leitungen und durch die Schaltzeiten von Gattern erzielen, in beiden Fällen natürlich nur als Verzögerung, also als nacheilende Phase. Außerdem kann man Schaltflanken durch Tiefpässe abflachen. Dies führt ebenfalls zu einer Verzögerung. Leitungsverzögerungen, eventuell kombiniert mit Tiefpässen, eignen sich in integrierten Schaltungen nur für ganz kurze Zeiten, weil sie ansonsten zu viel Platz benötigen und die Freiheitsgrade beim Design unnötig stark einengen. So bleiben für größere Verzögerungszeiten vorallem Tiefpässe und Gatter als verzögernde Elemente übrig.

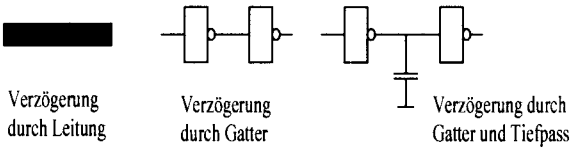


Abb. 9 Schaltungselemente zur Signalverzögerung

Solche Strukturen lassen sich fast beliebig verlängern. Die Verzögerungszeit muss zudem einstellbar sein. Eine einfache Möglichkeit besteht in der Konstruktion von umschaltbaren Ketten aus Verzögerungselementen.

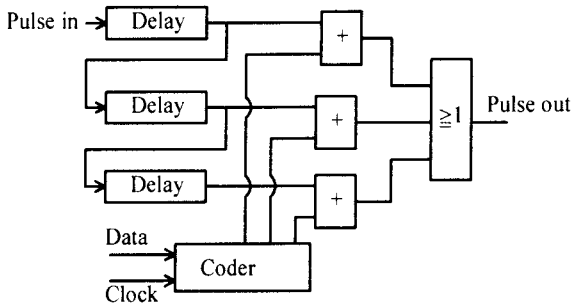


Abb. 10 Umschaltbare Delay-Elemente mit programmierbarer Verzögerungszeit

Es sind auch komplexere Strukturen denkbar, bei denen ein Puls sich durch ein Labyrinth von Verzögerungsgliedern seinen Weg suchen muss.

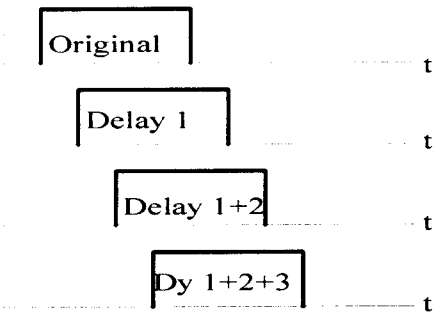


Abb 11. Beispiel zur Verzögerung von Pulsen.

Strategie zur Erzeugung eines GMSK-Signals

Der Träger wird mit der Originalfrequenz, beispielsweise aus einem Quarzoszillator, zugeführt. Daraus werden im Chip dann zwei Signale mit 180° Phasenversatz erzeugt. Die beiden Signale können über Verzögerungsketten bis über einen Winkel von über 180° hinaus verzögert werden. Damit ist es möglich, vor- und nacheilende Phasen bis zu beliebig großen Winkeln zu erzeugen.

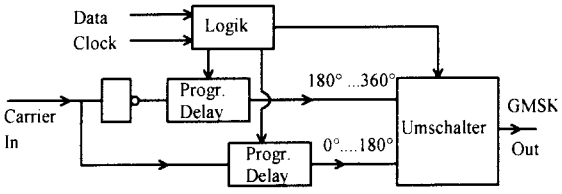


Abb. 12 Schaltung zur Erzeugung beliebiger Phasenverschiebungen

In Abhängigkeit von der Bitfolge muss die Verzögerung einem unterschiedlichen Zeitverlauf folgen. Am einfachsten ist eine Folge von mehreren „Einsen", die zunächst einmal betrachtet werden soll.

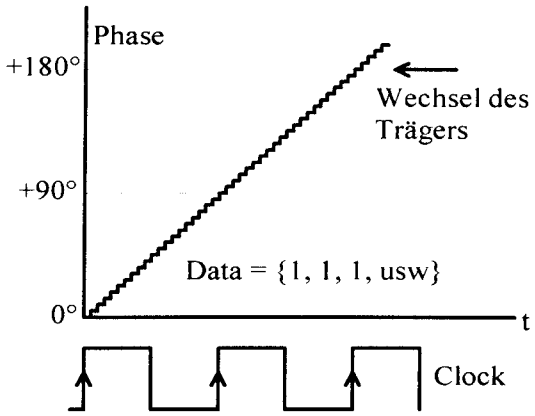


Abb. 13 Phasenverlauf mit diskreten Inkrementen zusammen mit dem Datenstrom bei MSK

Das Signal wird in gewissen Zeitabständen um ein zusätzliches Increment verzögert, so dass sich eine quasi-lineare Phasenverschiebung ergibt. Prinzipiell sollte die verbleibende Stufung so fein wie möglich ausgeführt werden. Bei Winkeln größer als 180° wird zwischen den beiden Trägern umgeschaltet. Grundsätzlich ist bei GMSK der Phasenverlauf abhängig vom vorhergehenden und vom nachfolgenden Bit. Deshalb ist eine Speicherung um mindestens zwei Bit erforderlich. Damit gibt es insgesamt acht verschiedene Phasenverläufe, die in Abhängigkeit von der Bitfolge eingesetzt werden müssen. Für den Phasenverlauf einer "Eins" gibt es somit vier verschiedene Varianten, siehe die nachfolgende Abbildung

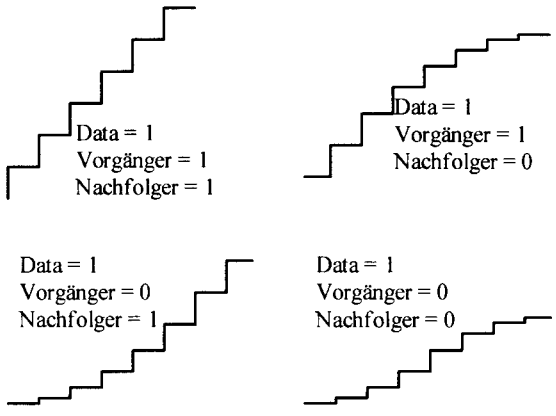


Abb. 14 Phasenverlauf bei GMSK

Im Chip müssen also diese acht Zeitfunktionen als Verzögerungsketten vorrätig gehalten werden. Die "Ausrundung" des Phasenverlaufs beim Bitwechsel von Null mach Eins muss für einen speziell festgelegten BT-Faktor berechnet sein. Eventuell lassen sich aber auch mehrere BT-Datensätze in einem Chip implementieren. Dadurch könnten z.B. MSK und auch mehrere GMSK Funktionen zur Verfügung stehen, was ein erheblicher Vorteil für die universelle Verwendbarkeit des Chips bedeuten würde.

Technische Spezifikationen

Es gibt einige Daten, die durch die Konstruktion eines solchen Chips festgelegt werden. Dazu gehören:

- Träger-Frequenz
- Datenrate (evtl. mehrere zur Auswahl)
- BT-Verhältnis (evtl. mehrere zur Auswahl)

Die Trägerfrequenz sollte bei 10,7 Mhz liegen. Diese Frequenz ist seit den Anfängen des UKW-Rundfunks eine Quasi-Norm. Für 10,7 Mhz gibt es sehr preisgünstige Bandfilter und Verstärker. Deshalb werden auch heute, wo die eigentliche Übertragung im GHz-Bereich stattfindet, sehr viele Geräte noch auf diese Frequenz hin ausgelegt. Die Festlegung auf diese Frequenz hat auch direkte Auswirkungen auf die Feinheit der Phasensprünge.

Die Datenrate sollte sich unbedingt an der binär gestaffelten Norm von 9600, 19200, 38400.. usw Baud orientieren. Dies erleichtert den Anschluss von UARTs an das System.

Ein kleines BT-Verhältnis verringert die Bandbreite, erhöht aber die Schwierigkeiten bei der Demodulation und damit verbunden auch die Bitfehlerrate. Der Mobilfunk verwendet allgemein ein BT von 0,3. Dieser Zahlenwert und vielleicht noch
BT = 0,5; 0,8; 1 und ∞ sind vernünftige Werte für allgemeine Anwendungen. Bezogen auf die Verzögerungsketten im Chip bedeuten kleinere BT-Faktoren, dass die Unterschiede zwischen den Laufzeiten immer feiner gestaffelt werden müssen.

Eine ganz wesentliche Größe ist die Anzahl und der zeitliche Abstand der Phasensprünge. Sie hängen von verschiedenen Parametern ab. Die maximal mögliche Zahl der Phasensprünge ist durch die Menge der im Chip implementierten Verzögerungselemente festgelegt. Die Dauer einer mit Gattern realisierten minimalen Verzögerungseinheit ergibt sich aus der doppelten Schaltzeit eines Inverters. Je nach Technologie kann man dabei Werte zwischen 2 und 6 ns annehmen. Für die weiteren Überlegungen soll ab jetzt mit einer Verzögerungszeit von 2 ns gerechnet werden. Geringere Verzögerungszeiten müssen mit Leitungselementen gebildet werden. Eine Verzögerung von 180° bei 10,7 Mhz entspricht einer Zeit von ca. 47 ns. Um diese Verzögerung mit Gattern zu erzielen, werden demzufolge 24 Delay-Elemente benötigt. Der zeitliche Abstand der Phasensprünge ist umgekehrt proportional zur Baud-Rate. Eine im Chip integrierte Steuerung sorgt dafür, dass die passende Umschaltfrequenz angewendet wird.

Spektrale Darstellung

Die Stufung der Phase anstelle eines glatten Verlaufes hat natürlich Auswirkungen auf das Spektrum. Es gibt zwei Faktoren, welche das Spektrum gegenüber einem rein analogen Schaltkreis verschlechtern:

- Abweichen der Signalform von dem durch Gauß-Filter erzeugten Verlauf

- Stufung durch Phasensprünge anstelle eines glatten Verlaufs

Die Abweichung von der idealen Gauß-Form macht sich durch Harmonische der Bitfrequenz bemerkbar, während die Phasensprünge sich weiter weg vom Träger durch ihre Schaltfrequenz und deren Harmonische wiederfinden. Insgesamt wird dadurch das Spektrum um einige diskrete Linien breiter. Die nachfolgende Skizze zeigt den prinzipiellen Verlauf, hier unter der Annahme eines periodischen Datensignals.

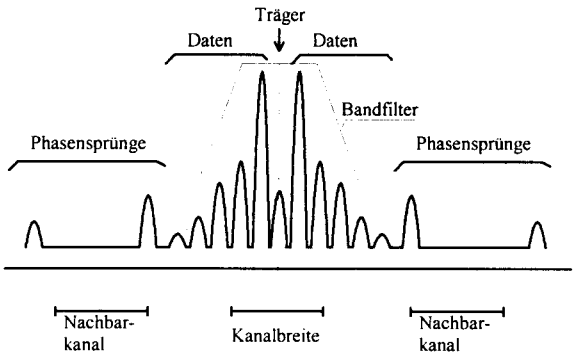


Abb. 15 Erwartetes Sektrum des GMSK-Generators

Die Seitenlinien der Daten stammen teilweise aus der Hüllkurve der Bessel-Funktion. Sie sind grundsätzlich ein Bestandteil des Spektrums, auch bei völlig glattem Phasenverlauf. Ihre Amplitude wird aber etwas vergrößert sein, weil es nicht gelingt, den Anteil der Harmonischen aus dem Datensignal so gering zu halten wie bei einer genauen Gauß-Filterung. Bezüglich der Kanalbandbreite ergeben sich dadurch aber keine nennenswerten Nachteile, denn dieser sollte ohnhin so breit sein, dass er einige Seitenlinien der Daten noch mit überträgt. Aufgrund ihrer höheren Frequenz finden sich die Phasensprünge mit ihren Harmonischen in größerem Abstand vom Träger. Sie könnten zu Nachbarkanalstörungen führen. Deshalb sollte grundsätzlich ein Bandfilter am Ausgang des Generators vorgesehen werden. Damit lassen sich diese Komponenten nahezu vollständig aus dem Spektrum entfernen.

Auto-Kalibration

Verzögerungszeiten in einer integrierten Schaltung sind von einer Vielzahl von Einflussgrößen abhängig. Dazu zählen die verwendete Technolie mit ihren Parametern, die unter Umständen einer erheblichen Streuung unterliegen, und die Schaltungstechnik. Es erscheint also nicht möglich, per Design definitive und reproduzierbare Verzögerungszeiten zu erhalten. Aus diesem Grund sieht mein Konzept eine periodisch wiederholte Autokalibration vor, die sich an der Quarzfrequenz des Trägers orientiert. Dies könnte so aussehen, dass in jedem Fall mehr als die im ungünstigsten Fall erforderliche Anzahl von Verzögerungselementen im Chip vorrätig sind. Wie viele dann tatsächlich zur Modulation eingesetzt werden, müsste durch eine automatische Kalibrier-Routine festgestellt werden. Diese könnte auch still im Hintergrund ablaufen und dauernd wiederholt werden. Im Normalfall bleibt natürlich ein kleiner Phasenfehler übrig. Dieser wird durch die Toleranz des Demodulators aber problemlos aufgefangen. In einem noch darüber hinaus gehenden Konzept ist auch eine Auto-Konfiguration denkbar, bei der aus der Vielzahl der im Chip verfügbaren Verzögerungselemente genau diejenigen für die tatsächliche Funktion herangezogen werden, die optimal zu dem geforderten BT-Faktor passen.

Erste Versuche

Mit einem FPGA wurden erste Versuche gemacht, einen phasensteuerbaren Generator zu bauen. Insgesamt zeigte sich, dass das Prinzip realisierbar ist. Der für diese Versuche zur Verfügung stehende FPGA war jedoch kaum für höhere Ansprüche auf diesem Gebiet geeignet. Außerdem war der Aufwand, den wir in die Entwicklung stecken konnten, dadurch begrenzt, dass uns nur ein Student im Rahmen eines seminaristischen Projektes zur Verfügung stand. Wir haben eine 101010..Endlos.- Bitfolge mit dreickförmigen Rampen fest einprogrammiert, siehe Abb. 16. Die Ergebnisse entsprechen den Erwartungen. Die feine Stufung der Phase zeigt sich im Spektrum erst in einem entsprechenden Abstand vom Träger.

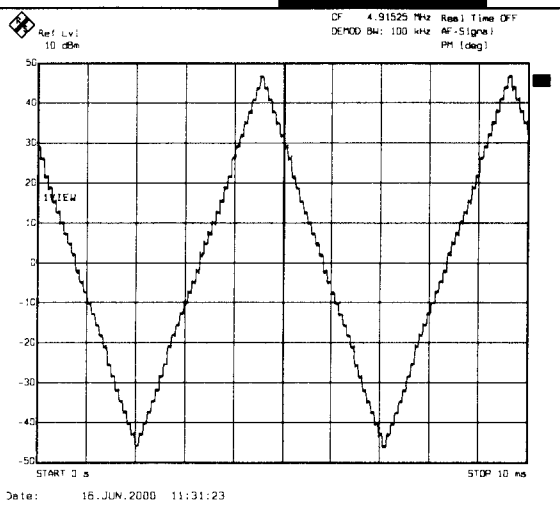


Abb. 16 MSK-Signal aus einem FPGA

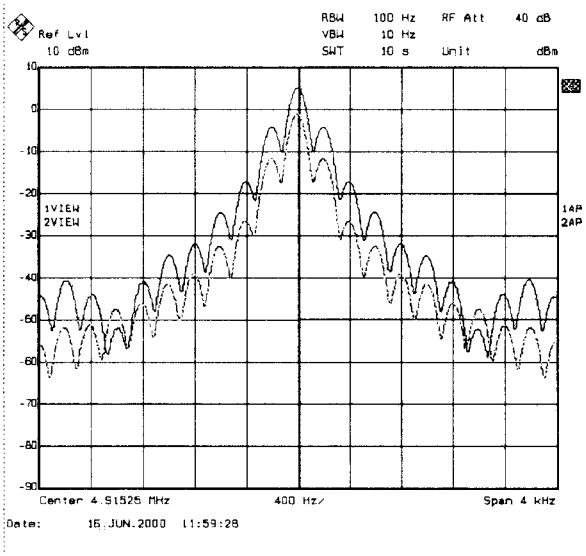


Abb. 17 Spektrum am Ausgang des FPGA (oben) im Vergleich zu einem Signal aus einem Testgenerator

Zusammenfassung

Die bisher vorgestellten Ansätze zeigen, dass es möglich sein sollte, einen vollständig integrierten digitalen GMSK-Generator als Single-Chip zu realisieren. Ich bin überzeugt, dass ein solcher Chip für viele Schaltungsentwickler sehr willkommen wäre. In Anbetracht der bedeutenden Steigerungsraten, die im Markt der Daten-Funkübertragung sich vollziehen und für die Zukunft in noch stärkerem Maße zu erwarten sind, könnte ein solcher Chip auch wirtschftlich eine beachtliche Bedeutung erlangen.

Entwicklung eines Uhrenmakros mit eingebautem Selbsttest

Hans-Jürgen Jahn
Fachhochschule Ulm
Labor Schaltungsintegration

1 Einleitung

Die hier beschriebene Uhrenschaltung entstand als Diplomarbeit im Sommersemester 2000. Zur Schaltungseingabe wurde "Renoir" verwendet. Damit entstand eine technologieunabhängige VHDL-Beschreibung (Softmakro), die mit "Leonardo" (Zieltechnologie FPGA) und "Synopsys" (Zieltechnologie Standardzellenschaltung) synthetisiert wurde. Für den "csd"-Prozeß der Austria Mikrosysteme International AG (AMS) wurde ein Hardmakro erstellt. Eine Applikationsschaltung zeigt die Verwendbarbeit des Uhrenmakros. Diese Schaltung wurde sowohl mit einem FPGA als auch mit einem ASIC realisiert.

Der folgende Beitrag beschreibt die Funktionen des Uhrenmakros und erläutert dessen Aufbau.

Ein Datenblatt des Uhrenmakros ist unter `http://www.asic.lab.fh-ulm.de` verfügbar.

2 Eigenschaften

Im folgenden sind die wesentlichen Eigenschaften des Uhrenmakros aufgelistet:

Zeit: Sekunden, Minuten, Stunden Datum: Tag, Wochentag, Monat, Jahr
Taktfrequenz 32,768 kHz oder höher
asynchrone Schnittstelle
Zeit und Datum bcd-codiert
zwei Ausgänge mit 32 Hz und 1 Hz Rechteck-Signalen
eingebauter Selbsttest

3 Aufbau

Den Aufbau des Makros zeigt Abbildung 1. Im Blockschaltbild ist links die Synchronisation der Eingangssignale dargestellt. In der Mitte befinden sich die einzelnen Zähler für Sekunden, Minuten, Stunden, Tag, Monat und Jahr. Ein 15stufiger Binärteiler erzeugt das Enable-Signal für den Sekundenzähler. Der Wochentag wird aus dem Datum (Tag, Monat und Jahr) berechnet. Vor den Ausgängen befindet sich ein Register, mit dem alle Werte zwischengespeichert werden können.

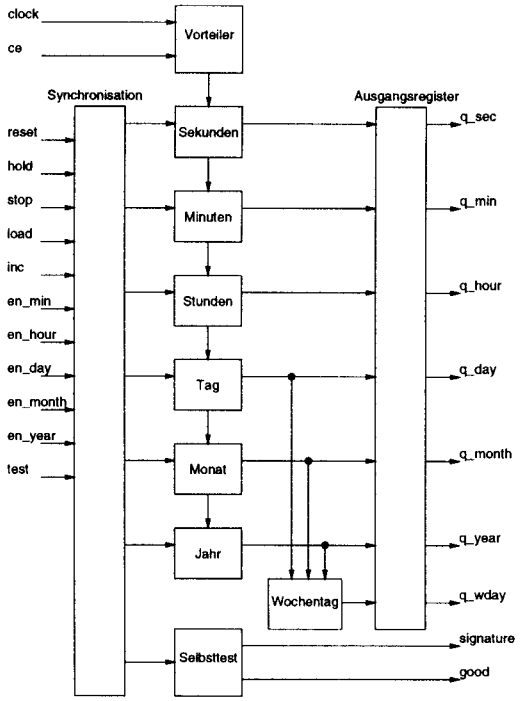


Abbildung 1: Blockschaltung der Uhr

3.1 Funktionen

Mit dem "reset"-Signal wird die Uhr auf den 1.1.2000, 00:00:00 Uhr gestellt. "reset" ist der einzige low-aktive Eingang.

Das Signal "ce" dient zur Freigabe des Vorteilers. Dadurch wird die Verwendung höherer Taktfrequenzen als 32768 Hz ermöglicht. Bei Betrieb mit 32768 Hz muß "ce" ständig '1' sein.

Mit dem "hold"-Signal werden Zeit und Datum in den Ausgangsregistern gespeichert. "hold" muß gesetzt werden, wenn nicht alle Werte während eines Taktzyklus gelesen werden[1]. "hold" muß auch gesetzt werden, falls die Ausgänge asynchron zum Takt des Uhrenmakros gelesen werden.

Zum Stellen der Uhr muß mit "stop"='1' in den Stellmodus gewechselt werden. Mit den einzelnen Enable-Signalen "en_min", "en_hour", "en_day", "en_month", "en_year" wird eine Stelle ausgewählt. Die Sekunden werden im Stellmodus immer auf 0 rückgesetzt. Das "load"-Signal dient zum Laden des an "d" anliegenden BCD-Werts. Zum Stellen der Uhr von Hand dient das "inc"-Signal. Mit jeder pos. Flanke von "inc" wird der Wert des ausgewählten Zählers um eins erhöht.

Zum Start des Selbsttests muß bei "test"='1' ein Reset ("reset"='0') durchgeführt werden. Nach dem Ende des Reset-Impulses beginnt der Test. Das Ergebnis wird an "good" ausgegeben. Bei gesetztem "good" wurde der Test erfolgreich durchlaufen.

3.2 Synchronisation

Alle Eingänge (außer "ce") verfügen über eine Synchronisation (Abbildung 2). Wenn die Eingangssignale bereits synchron zum Takt des Makros sind, ergibt sich dadurch eine Verzögerung von einer Taktperiode. Bei asynchronem Betrieb müssen alle Eingangssignale für mindestens eine Taktperiode anliegen.

Wenn "hold" zum Lesen der Uhrzeit gesetzt wird, müssen diese Zeiten berücksichtigt werden.

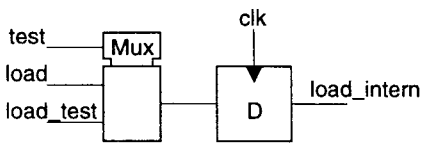


Abbildung 2: Synchronisation mit Multiplexer für den Selbsttest (siehe auch Abschnitt 4)

3.3 Zähler

Die Uhr besteht vor allem aus zweistelligen BCD-Zählern, die allerdings verschiedene Zählbereiche aufweisen. Zum Beispiel bei Sekunde und Minute von 0 bis 59 oder beim Monat von 1 bis 12. Eine Sonderstellung nimmt der Tag ein, da der Zählbereich von 1 bis (28,29,30,31) abhängig von Monat und Jahr ist. Durch die Verwendung der "Generic"-Anweisung[2] war es möglich, für alle Zähler die gleiche VHDL-Beschreibung zu verwenden (Abbildung 3). Nur der Tageszähler mußte entsprechend erweitert werden.

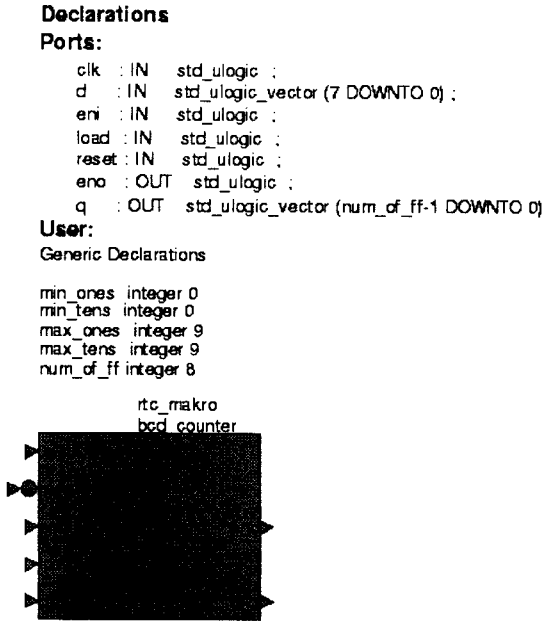


Abbildung 3: BCD-Zähler

3.4 Wochentag

Der Wochentag wird aus Tag, Monat und Jahr berechnet. Das Prinzip zeigt folgende Formel:

$$Wochentag = (\text{Tage seit } x + \text{Wochentag an } x) \bmod 7 \quad (1)$$

Dabei ist x ein beliebiges Datum (gewählt wurde der 1.1.2000). Die Wochentage sind dabei wie folgt codiert:

Montag	0
Dienstag	1
Mittwoch	2
Donnerstag	3
Freitag	4
Samstag	5
Sonntag	6

Nach dem Einsetzen von Tag, Monat und Jahr und einigen Vereinfachungen bleiben vor allem die Bildung

[1] Ansonsten können nicht zusammengehörende Werte gelesen werden, z. B. beim Wechsel von 13:59 auf 14:00 entweder 13:00 oder 14:59 (abhängig davon, welcher Wert zuerst gelesen wird).

[2] Synopsys unterstützt bei "Generic"-Werten nur den Datentyp Integer.

einer Summe mit fünf Operanden und die Modulo-Berechnung übrig. Die direkte Realisierung erfordert vier Addierer und ein Schaltnetz für die Modulo-Bildung.

Da der Wochentag nicht in einer Taktperiode berechnet werden muß, nimmt der Aufwand durch eine sequentielle Berechnung wesentlich ab:

- Die Summation wird mit einem Addierer in mehreren Schritten durchgeführt. Nach dem Addierer befindet sich ein Register, in dem die Summe gespeichert wird. Das Register liefert einen Operanden. Zu Beginn wird es mit 0 initialisiert. In den darauffolgenden Takten werden die einzelnen Summanden an den Addierer angelegt und die Zwischenergebnisse jeweils im Register gespeichert.

- Die Modulo-Berechnung wird als fortlaufende Subtraktion ausgeführt. Dazu kann der Addierer verwendet werden, wenn die Subtraktion von +7 durch die Addition von -7 (Zweierkomplement) ersetzt wird.

Damit ergibt sich die in Abbildung 4 gezeigte Blockschaltung. Mit einem Multiplexer werden die einzelnen Eingänge ausgewählt, danach erfolgt die Umsetzung vom BCD- ins Dualformat und schließlich der Addierer mit nachgeschaltetem Register. Ein Steuerwerk führt die vorher beschriebenen Funktionen aus.

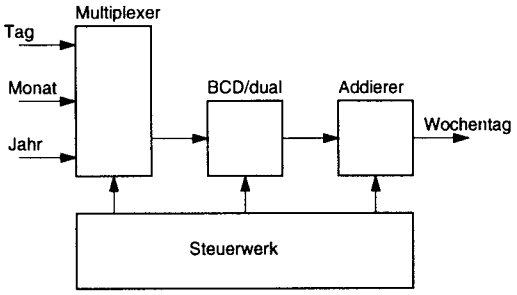


Abbildung 4: vereinfachtes Blockschaltbild der Wochentags-Berechnung

Die Berechnung wird durch zwei Ereignisse ausgelöst:

1. beim Umschalten vom Stellmodus in den Normalbetrieb (fallende Flanke an "stop")

2. beim Tageswechsel

Während der Berechnung, die maximal ca. 40 Takte dauert, wird "hold" intern gesetzt. Beim Tageswechsel erfolgt die Ausgabe der neuen Werte damit um diese Zeit verzögert, dafür ist der Wochentag aber immer gültig.

4 Selbsttest

4.1 Prinzip[1][2]

Integrierte Schaltungen müssen bei der Produktion getestet werden. Beim Test werden Stimuli an die Schaltungseingänge angelegt und die Schaltungsausgänge mit Sollwerten verglichen.

Beim Selbsttest sind Teile der Testschaltung mitintegriert (Abbildung 5). Ein wichtiger Grund für den Einsatz eines Selbsttests bei dieser Schaltung liegt in der Wiederverwendbarkeit. Bei der Einbettung in ein größeres System kann der Test mit minimalem Aufwand durchgeführt werden. Wenn mehrere Module mit dem gleichen Testverfahren ("test"-Eingang setzen, danach Reset durchführen, nach bestimmter Wartezeit "good"-Ausgang prüfen) eingesetzt werden, genügt ein UND-Gatter zur Verknüpfung der einzelnen Testergebnisse zu einem Gesamtergebnis.

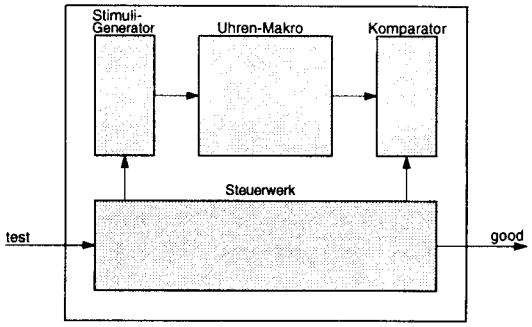


Abbildung 5: Integration der Testschaltung

Zur automatischen Erzeugung von Testmustern wird oft das Scanpath-Verfahren verwendet. Der Scanpath wird bei der Synthese automatisch erstellt und verbindet alle verwendeten Flipflops zu einem Schieberegister. Damit können beim Test seriell Daten geschrieben und gelesen werden. Durch den Scanpath besteht die restliche Schaltung nur noch aus kombinatorischen Elementen (Schaltnetz). Die Erzeugung von Testmustern für kombinatorische Schaltungen wird von entsprechenden Programmen automatisch durchgeführt. Die Testmuster werden dabei aber erst nach der Synthese entwickelt. Bei einem technologieunabhängig einsetzbaren Makro ist dieses Vorgehen deshalb nicht möglich.

Da beim Selbsttest keine langen Stimuli gespeichert werden können, werden z. B. Zufallsmuster zum Test benutzt. Diese können mit linear rückgekoppelten Schieberegistern einfach erzeugt werden. Zur Bestimmung der Fehlerüberdeckung wird eine Fehlersimulation durchgeführt.

Der im folgenden beschriebene Selbsttest verwendet keinen Scanpath. Während beim Scanpath-Verfahren entsprechend der Schaltungsstruktur geprüft wird, beruht dieser Test auf der Schaltungsfunktion. Zum Beispiel werden Zähler durch Laden eines bestimmten Anfangswertes und anschließendes

Zählen geprüft.

Der Selbsttest wurde komplett mit "Renoir" erstellt. Das Makro ist damit unabhängig von einer bestimmten Technologie. Bedingt dadurch kann die Fehlerüberdeckung bei verschieden synthetisierten Schaltungen (unterschiedliche Technologien und Einstellungen) geringfügig verschieden sein.

4.2 Fehlersimulation mit extern angelegten Stimuli

Die Uhrenschaltung wurde zuerst ohne Testfunktionen beschrieben. Nach der Synthese für den "csd"-Prozeß wurde die Fehlersimulation mit Quickfault durchgeführt. Quickfault verwendet das "stuck at"-Fehlermodell.

Die Stimuli wurden in einer Datei gespeichert und an die normalen Schaltungseingänge angelegt. Während der Simulation wurden alle Schaltungsausgänge geprüft.

Die nicht erkannten Fehler sind nach einer Simulation im Schaltplan markiert. Damit können Rückschlüsse auf die Testbarkeit einzelner Schaltungsteile gezogen werden. Erwartungsgemäß wurden bei den Zählern die meisten der möglichen Fehler erkannt, hingegen blieben bei den Blöcken "Vorteiler", "Sekunden" und "Wochentag" viele Fehler unerkannt.

Beim Vorteiler liegt die Ursache darin, daß der Freigabe-Ausgang für den Sekundenzähler nicht geprüft wird. Dieser Ausgang wird erst 32768 Takte nach einem Reset gesetzt, die Testdauer ist aber wesentlich kürzer. Der Test der Zähler wird im Stellmodus durchgeführt. Im Stellmodus wird der Sekundenzähler nur auf 0 rückgesetzt. Die Zählfolge kann nicht geprüft werden. Schließlich wird die Berechnung des Wochentags nur einmal durchgeführt.

Um diese Mängel zu beseitigen, wurde die Uhrenschaltung erweitert.

4.3 Erweiterung der Schaltungsbeschreibung

Die folgenden Änderungen wurden vorgenommen:

- Multiplexer an allen Eingängen schalten zwischen externem Signal und Stimuli um.
- Der Vorteiler wird während des Tests in drei parallel laufende Stufen aufgeteilt.
- Der Sekundenzähler zählt während des Tests dauernd.
- Der Wochentag wird während des Tests mehrmals mit verschiedenen Werten für Tag, Monat und Jahr berechnet.
- Zur Komprimierung der Ausgangssignale wurden die Ausgangsregister während des Tests zu einem Signaturregister verbunden. Allerdings ergaben sich bei dieser Anordung sehr viele nicht erkannte Fehler in diesem Bereich, sodaß das Signaturregister schließlich getrennt realisiert wurde. Das Signaturregister besteht nun aus neun Flipflops und hat neun Eingänge. Deshalb wurden die ca. 45 Ausgänge des Makros zu 9 Gruppen zusammengefaßt. Jede Signalgruppe wurde mit EXOR-Gattern zu einem Signal zusammengefaßt (Abbildung 6).
- Die Schaltungsbeschreibung wurde schließlich um ein Steuerwerk erweitert. Dieses kontrolliert den Test und erzeugt die Stimuli.

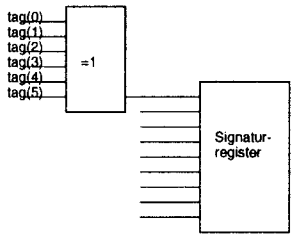


Abbildung 6: mehrere Ausgänge werden vor dem Signaturregister zu einem Wert zusammengefaßt

4.4 Fehlersimulation mit Selbsttest

Bei der Fehlersimulation der erweiterten Schaltung müssen nur noch einige externe Stimuli definiert werden:

```
//clock
$set_clock_period(100);
$force("clk", "0", 0, void, void, @absolute, @repeat, void);
$force("clk", "1", 50, void, void, @absolute, @repeat, void);
//Eingangssignale
FORCE reset       0       0
FORCE test        1       0
FORCE ce          1       0
FORCE reset       1     500
```

Geprüft wird nur noch der "good"-Ausgang.

Da das erreichte Ergebnis nicht ausreichend war, mußte in mehreren Iterationen die Schaltungsbeschreibung geändert, die Synthese durchgeführt und schließlich die neue Schaltung simuliert werden.

4.4.1 Verkürzung der Simulationszeit

Beim beschriebenen Vorgehen beträgt die Simulationszeit über eine Stunde. Wesentlich kürzere Simulationszeiten (um eine Minute) ergeben sich, wenn alle Schaltungsausgänge geprüft werden. Dadurch wird die erreichte Fehlerüberdeckung im allgemeinen etwas besser sein, als wenn nur der "good"-Ausgang geprüft wird. Unterschiedliche Ergebnisse sind durch die Fehlermaskierung des Signaturregisters bedingt.

4.5 Testablauf

Während des Tests werden die folgenden Punkte geprüft:

- Laden der Zähler mit "00000000" und "11111111"
- Zählfolge über den gesamten Zählbereich (der Übertragsausgang wird nicht geprüft)
- mehrfache Berechnung des Wochentags mit verschiedenen Werten für Tag, Monat und Jahr
- Funktion der Ausgangsregister
- Vorteiler (aufgeteilt in zwei Stufen)

4.6 Ergebnisse

Quickfault liefert die folgenden Daten:

```
Undetected:   7,59 %
Detected:    82,39 %
Possible:    10,02 %
```

Zur Bedeutung der einzelnen Angaben:

- Undetected: Dies sind Fehler, die während des Tests unentdeckt bleiben.
- Detected: Diese Fehler werden während des Tests sicher erkannt.
- Possible: Diese Fehler werden möglicherweise erkannt. Sie können in zwei Gruppen eingeteilt werden. Zum einen sind es Fehler an den Clock-Eingängen von Flipflops, d. h. dieser Eingang ist entsprechend dem verwendeten Fehlermodell immer '0' oder immer '1', das Flipflop schaltet also nie. Deshalb werden diese Fehler mit hoher Wahrscheinlichkeit erkannt. Zum anderen handelt es sich um Fehler an den Select-Eingängen von Multiplexern.

Durch die Testschaltung nahm der Flächenbedarf um ca. 22% zu. Der Test dauert ca. 1140 Taktperioden, dies sind bei 32 kHz ca. 35 ms.

Die Fehlerüberdeckung läßt sich sicher noch verbessern. Allerdings war dies aus Zeitgründen nicht möglich.

5 Applikation

Zur Demonstration der Funktionen des Uhrenmakros wurde eine Applikationsschaltung entwickelt. Die Bedienung erfolgt über Taster und die Anzeige von Zeit bzw. Datum auf einem 6stelligen LC-Display. Diese Schaltung besteht aus einem Teil, der zusammen mit dem Uhrenmakro auf einem Chip integriert wird und einigen externen Bauteilen (Taster, Anzeige, ...). Abbildung 7 zeigt das Blockschaltbild. Auf den Abdruck des Schaltplans wurde aus Platzgründen verzichtet. Die zugehörige Leiterplatte zeigt Abbildung 8.

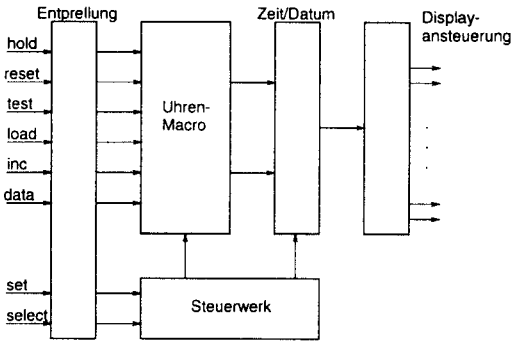


Abbildung 7: Blockdiagramm der Applikationsschaltung

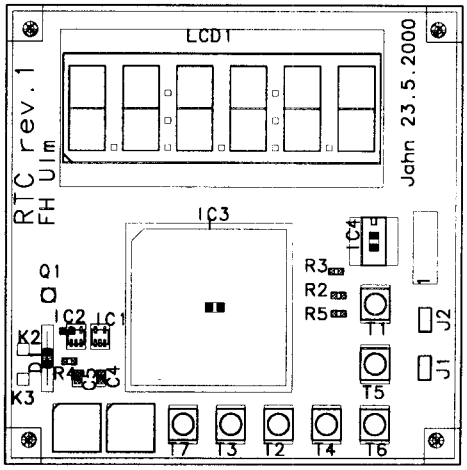


Abbildung 8: Leiterplatte der Applikationsschaltung

Beim Entwurf der Schaltung wurde darauf geachtet, daß sowohl ein FPGA (XC3195A von Xilinx) als auch ein ASIC eingesetzt werden können. Das Gehäuse und die Anschlußbelegung des ASICs mußten dabei entsprechend dem FPGA gewählt werden.

Mit dieser Schaltung kann der Selbsttest durchgeführt werden. Das Ergebnis wird als '0' oder '1' auf dem LC-Display ausgegeben. Die Signatur selbst wird ebenfalls angezeigt.

6 Layout

Das Layout des Uhrenmakros zeigt Abbildung 9. Dazu einige Daten:

- Technologie: AMS "csd", 0,35 μm CMOS, 3 Metall- und 2 Polylagen, 3,3 Volt Versorgungsspannung
- Größe: ca. 507 x 442 μm^2
- Gatter: ca. 840

Das Layout der Applikationsschaltung zeigt Abbildung 10:

- Größe: 2,7 x 2,8 mm^2

- Pads: 70
- vorgesehenes Gehäuse: JLCC84

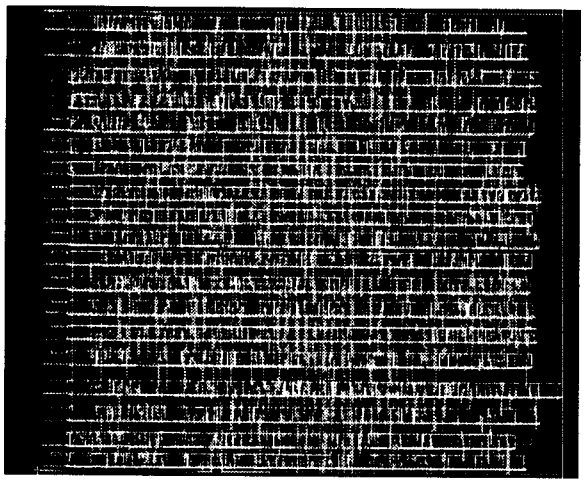


Abbildung 9: Layout des Makros

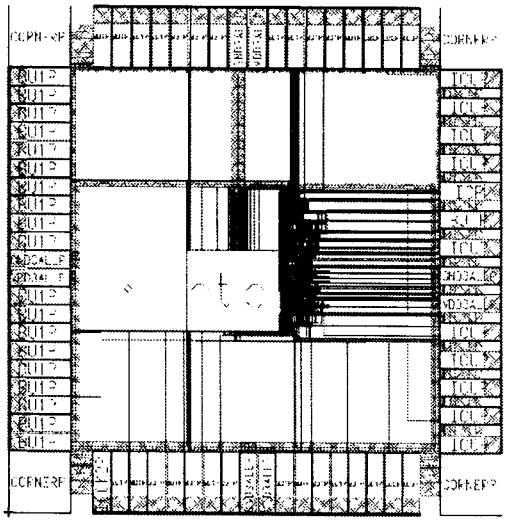


Abbildung 10: Layout der Applikationsschaltung

Bei der Erstellung des Layouts traten trotz automatischem Place&Route beim Design Rule Check etliche Fehler auf. Viele davon sind zwar auf die Blackbox-Darstellung der Standardzellen zurückzuführen, aber dennoch mußten etliche Fehler von Hand beseitigt werden.

7 Zusammenfassung

Das beschriebene Uhrenmakro steht als technologieunabhängige VHDL-Beschreibung oder als fertiges Layout für den "csd"-Prozeß von AMS zur Verfügung. Die Anwendung des Makros wird durch den eingebauten Selbsttest erleichtert.

Eine Applikationsschaltung zeigt die Funktionen des Makros. Die Fertigung dieser Schaltung über Europractice wird durch Mittel der MPC-Gruppe ermöglicht.

Literatur

[1] Wojtkowiak, Hans: *Test und Testbarkeit digitaler Schaltungen*, Stuttgart, Teubner, 1988

[2] Wunderlich, Hans-Joachim: *Hochintegrierte Schaltungen: Prüfgerechter Entwurf und Test*, Berlin, ..., Springer 1991

Fragen richten Sie bitte an:
Fachhochschule Ulm
Prof. Führer
Postfach 3860
89028 Ulm
Tel. 0731/50-28338

Three-Port Oscillator Design with Puff

Horst Nielinger, *Senior Member, IEEE*

***Abstract*— The principle of a very simple oscillator circuit is explained using h-parameters with a first order MOSFET transistor model. A design procedure for the realization of this oscillator in microstrip (Euroboard, $f = 750$ MHz) is presented in a step by step approach using the Caltech simulation and layout program PUFF.**

***Index Terms*—Microwave oscillator, PUFF application.**

I. Introduction

IN [1] a very attractive microstrip oscillator circuit (realized for $f = 5$ GHz with HEMT-transistor FHX35G) is discussed, not to be found elsewhere in literature. In designing a new course "Microwave Circuits" in which the CAD-program for Microwave Integrated Circuits PUFF [2] was to be used and experiments on cheap glass fiber material were to be done it was tried to realize the oscillator circuit with the same transistor for $f = 750$ MHz. This did not work and caused some thoughts about the principle of operation of the oscillator mentioned above. The explanations given in [1] are short on detail and at a high level of abstraction and in particular do not discuss the requirements for the oscillator transistor. Therefore it was felt appropriate to give a down-to-earth explanation of the oscillator operation and its design procedure in order to give not only the "future Caltech Nobel laureates" [1] but also students and lecturers "with very little brain," the possibility to enjoy an RF energy producing circuit of unbeatable simplicity and elegance.

II. The Oscillator Principle

It is well known [3] that an ideal transistor with two capacitors appropriately added produces a negative resistance essential for oscillator design. The main idea of the oscillator discussed here is to use the internal transistor capacitances in order to produce a negative input resistance. Fig. 1 shows a very simple equivalent circuit for a MOSFET transistor with floating source thus forming a three port circuit. The floating source can be achieved by a parallel resonant circuit for the design frequency so a two port analysis is still sufficient for the input–output behavior. Using hybrid parameters we have

$$v_{in} = h_{11} i_{in} + h_{12} v_0$$
$$i_0 = h_{21} i_{in} + h_{22} v_0.$$

Manuscript received August 30, 1996; revised February 16, 1999.

The author was with the School of Engineering and Manufacture, De Montfort University, Leicester, U.K. He is now with Fachhochschule Furtwangen, 78120 Furtwangen, Germany.

Publisher Item Identifier S 0018-9359(99)09005-6.

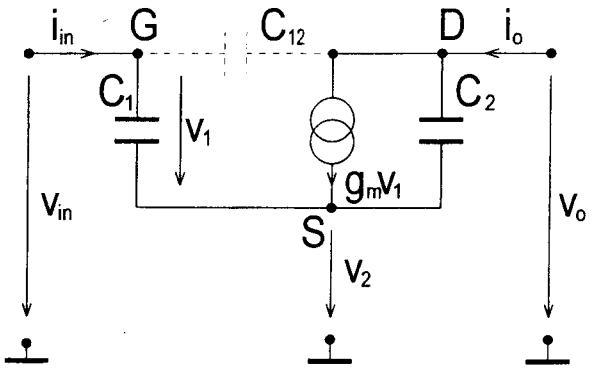


Fig. 1. Simplified equivalent circuit for a MOSFET transistor.

For h_{11} we find

$$h_{11} = \left.\frac{v_{in}}{i_{in}}\right|_{v_0=0} = \left.\frac{v_1 + v_2}{i_{in}}\right|_{v_0=0} = -g_m X_1 X_2 + j(X_1 + X_2)$$

with

$$X_1 = -\frac{1}{\omega C_1}; \quad X_2 = -\left.\frac{1}{\omega C_2}\right|.$$

Here we have generated the negative input resistance necessary for an oscillator! The other h-parameters can be found simply by inspection

$$h_{12} = \left.\frac{v_{in}}{v_0}\right|_{i_{in}=0} = 1$$
$$h_{21} = \left.\frac{i_0}{i_{in}}\right|_{v_0=0} = -1$$
$$h_{22} = \left.\frac{i_0}{v_0}\right|_{i_{in}=0} = 0.$$

If we connect the output with the desired load resistor $Z_0 (= 50\ \Omega)$ we find for the input impedance simply the series connection of h_{11} and Z_0

$$\frac{v_{in}}{i_{in}} = h_{11} + Z_0.$$

This input impedance has to be compensated in order to form an oscillator. Therefore the impedance connection to the input must be

$$Z_G = -h_{11} - Z_0 \quad \text{(Input Oscillator Condition)}.$$

If we connect the input with Z_G we find for the output impedance simply the series connection of h_{11} and Z_G

$$Z_2 = \frac{v_0}{i_0} = h_{11} + Z_G.$$

Inserting the input oscillator condition we get

$$Z_2 = -Z_0 \quad \text{(Output Oscillator Condition)}.$$

That means if the outer impedance at the input (Z_G) is equal to the negative of the input impedance, then the outer impedance of the output (Z_o) is also equal to the negative of the output impedance. The oscillator conditions are both fulfilled simultaneously.

III. The Oscillator Design

A very cheap SMD MOSFET (BF999) was chosen as active element. Information about packaging and s-parameters of this transistor can be found in the Internet [4]. The design frequency was chosen as $f = 750$ MHz because then the micro-strip circuit fits nicely on a Euroboard (160 mm $\times$ 100 mm).

From the interpolated values for 750 MHz

$$s_{11} = 0.82, -64^0$$
$$s_{22} = 0.96, -25^0$$

we can calculate or determine by means of the Smith Chart, the normalized input and output admittances

$$G_1 Z_0 + jB_1 Z_0 = 0.12 + j0.62 \approx j0.62$$
$$G_2 Z_0 + jB_2 Z_0 = 0.06 + j0.22 \approx j0.22.$$

In both admittances the capacitive component is dominant which justifies the simple equivalent circuit Fig. 1. In order to calculate the negative real part-$g_m X_1 X_2$, we need information about g_m. For low frequencies we have the well-known relation [5]

$$-2g_m Z_0 = s_{21} \approx -1.574$$

where the numerical value is taken from the list of s-parameters [4] at $f = 60$ MHz.

Therefore

$$g_m Z_0 = 0.787.$$

The negative normalized resistance we would expect at 750 MHz is therefore

$$\frac{-g_m X_1 X_2}{Z_0} = -\frac{g_m Z_0}{B_1 Z_0 \cdot B_2 Z_0} = -\frac{0.787}{0.62 \cdot 0.22} = \underline{\underline{-5.77}}.$$

IV. The Oscillator Design Procedure

Fig. 2 shows relevant parts of a PUFF screen dump of the oscillator circuit used in the first design step. Note in the PARTS window that indefinite s-parameters ("i" in line g) are used for the transistor BF999 in order to have the source floating. One can note in particular that the length of the transistor is given as only 1.5 mm. This information is necessary for the layout, but in order to see the details of the circuit, it is a good idea to switch to Manhattan Layout Mode (last line in the BOARD window). Then the distance between the connection points becomes 1/10 of the board size irrespective of the given dimensions. The simple circuit shown in the LAYOUT window of Fig. 2 consists only of

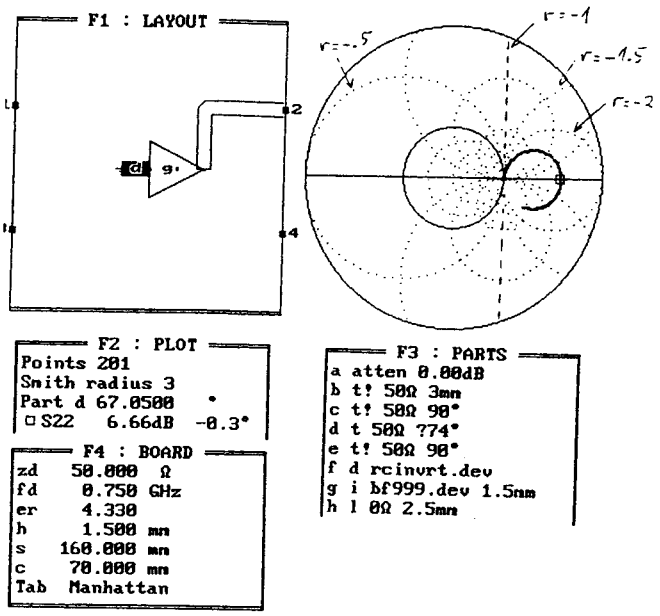


Fig. 2. Negative output impedance as function of the length of the shorted input line.

the transistor with a shorted transmission line at its input. The simulation makes use of the component sweep option of PUFF ("?" in line d) so the reflection coefficient at the output (s_{22}) is measured as a function of the input transmission line length in degrees at the input. As the Smith Chart radius is chosen to 3 (see PLOT window), the locus of s_{22} is totally outside the unit circle, the boundary of the normal Smith Chart. This signifies the existence of a negative real part of s_{22} necessary for the oscillator function. The cursor in Fig. 2 shows the real part of s_{22} which is achieved when the input shorted transmission line resonates with the transistor capacitances.

We get $\mathrm{Re}(s_{22}) = 6.66$ dB $= 2.15$ at a line length of 67° for transmission line "d" (PLOT window). Therefore

$$\frac{R_2}{Z_0} = \frac{1 + s_{22}}{1 - s_{22}} = \frac{3.15}{-1.15} = -2.74.$$

This result shows quite a remarkable deviation from the theoretically expected value of -5.77. Of course losses have to be taken into account (real parts of the transistor input and output admittances, loss resistance of the resonating transmission line) but one major deteriorating effect in the generation of the negative resistance is due to the capacitance C_{12}, shown dotted in Fig. 1 which shunts h_{11}. A simple calculation results in a corrected formula for the negative normalized output resistance

$$\frac{R_2}{Z_0} = \frac{\dfrac{-g_m Z_0}{B_1 Z_0 \cdot B_2 Z_0}}{\left(\dfrac{g_m Z_0 \cdot B_{12} Z_0}{B_1 Z_0 \cdot B_2 Z_0}\right)^2 + \left(\dfrac{B_{12} Z_0}{B_1 Z_0 + B_2 Z_0} + 1\right)^2}.$$

Even for a value as low as $B_{12} Z_o = 0.1$, the negative resistance in our case is reduced from -5.77 to -3.63. But nevertheless Fig. 2 shows that we have generated a negative resistance necessary for oscillator design. In order to fulfil the oscillator condition at the output, the normalized output

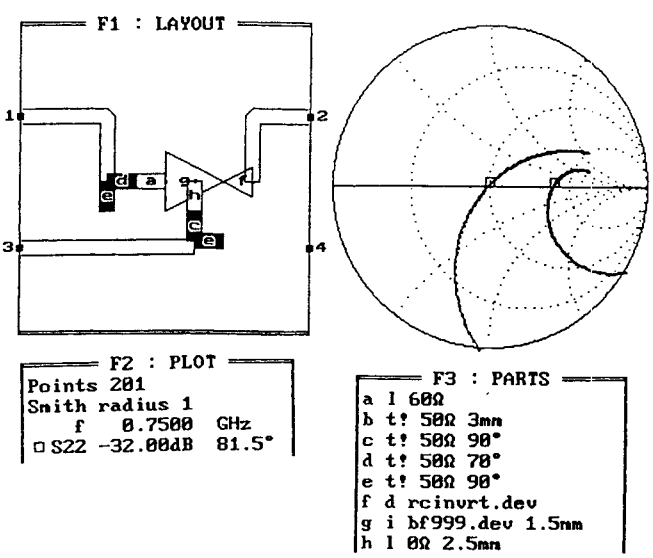


Fig. 4. Frequency response of s_{22} with and without resistor at input (simulation of transistor saturation).

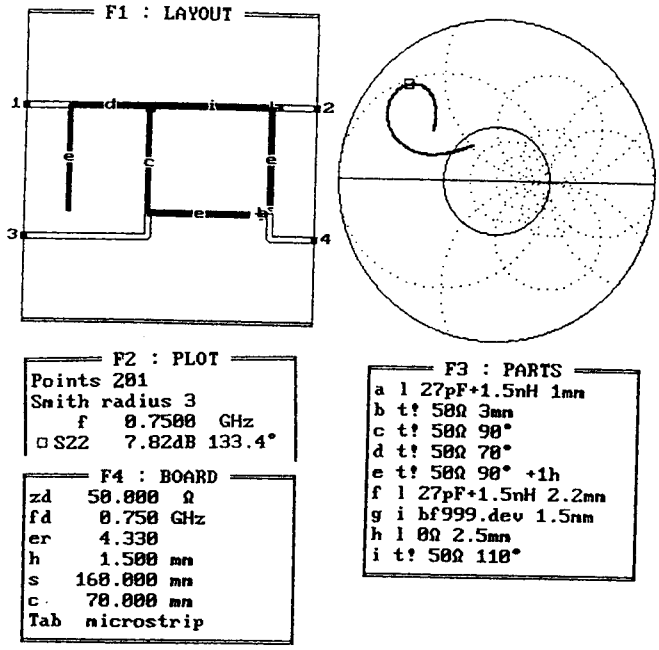


Fig. 5. Final frequency response of s_{22} with additional bias circuitry (MICROSTRIP layout mode).

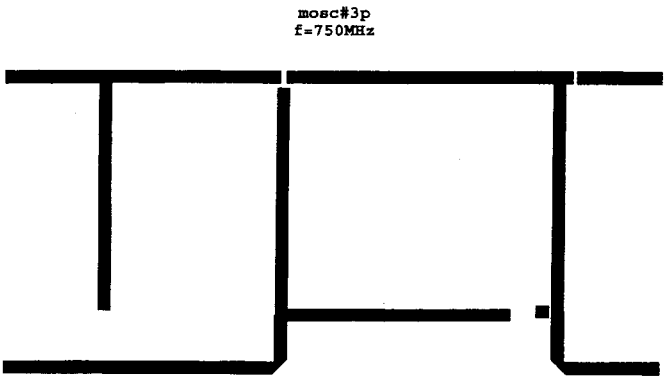


Fig. 6. Layout of the oscillator circuit.

resistance has to be -1. In Fig. 2, some circles with constant normalized negative real parts are signified in the extended Smith Chart. The circle with the -1 value is degenerated to a perpendicular line that means that the corresponding $s_{22} = \infty$. We registered above that the transistor parameter g_m is decisive for the negative resistance. When the transistor saturates g_m will become smaller automatically. That means that we can simulate the effect of saturation by adding a variable resistor to the input resonance circuit in order to fulfil the oscillator condition. Then the negative resistance is reduced by adding a positive resistance, in reality the negative resistance is reduced by the nonlinear effect of transistor saturation which reduces the parameter g_m. This approach appears simpler than the simulation of saturation effects by means of an attenuator proposed in [1].

As discussed before an increasing resistance at the input of the transistor will reduce the value of the negative resistance seen at the output. Therefore the value of s_{22} will increase remarkably by this measure. As it is very inconvenient to handle reflection coefficients near infinity, a reflection coefficient inverter can be designed which will transform any reflection coefficient Γ_L into $1/\Gamma_L$ in good approximation [1]. The design follows from the well-known formula [5]

$$s'_{11} = s_{11} + \frac{s_{12} \cdot s_{21}\Gamma_L}{1 - s_{22}\Gamma_L} = \frac{s_{11} - (s_{11}s_{22} - s_{12}s_{21})\Gamma_L}{1 - s_{22}\Gamma_L.}$$

If $s_{11} = s_{12} = 1000$ and $s_{21} = s_{22} = -1000$ the (for $\Gamma_L \gg 1/1000$)

$$s'_{11} = \frac{1}{\Gamma_L}.$$

So the reflection coefficients near infinity will be transformed into reflection coefficients near zero to be found right in the middle of the normal Smith Chart.

One more remark to Fig. 2. The circles for negative imaginary parts remain in the negative half plane, the circles for the positive imaginary parts remain in the positive half plane irrespective of the sign of the real part. This means that for a series resonant circuit with a constant negative real part, the locus of the frequency response will form a counter clockwise loop whereas normally (positive real parts, normal Smith Chart), one always experiences clockwise loops with increasing frequency. The occurrence of a counter clockwise loop can be taken as a criterion for a negative real part (as pointed out in [1] without explanation) even if this loop is to be found anywhere off the circumference of the normal Smith Chart due to phase shift by additional transmission lines (Fig. 5).

Fig. 3 shows the frequency response of s_{22} and $1/s_{22}$ after the reflection coefficient inverter was placed. The $\hat{}P$-option of PUFF allows the display of the frequency response of different circuits in one picture. In addition, the source of the transistor is connected with a resonant circuit which represents a very high impedance for the design frequency but allows a 50 Ω—connection to ground via port 3 for dc and frequencies far off

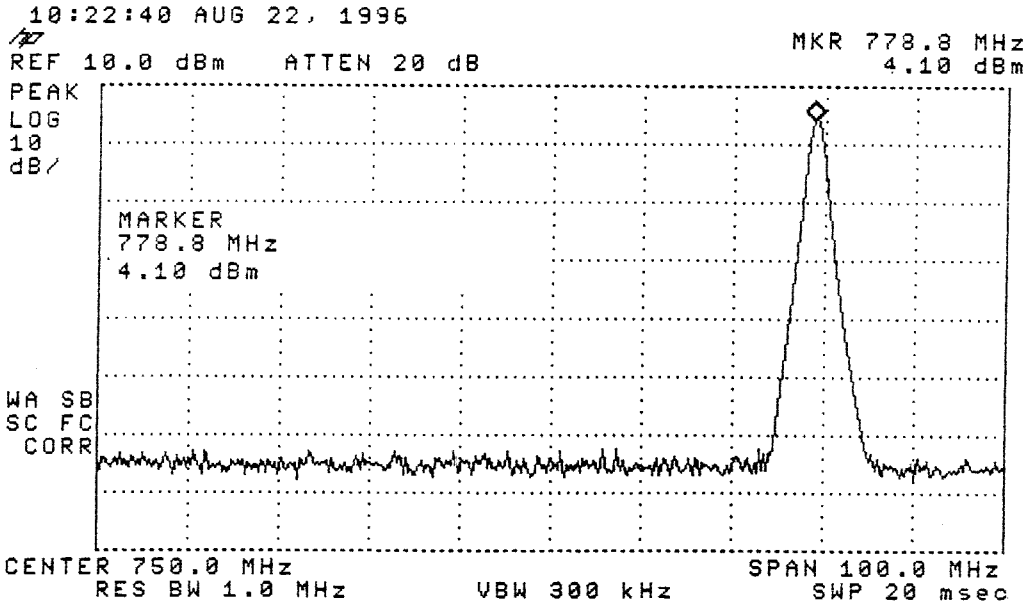


Fig. 8. Spectrum at output of the oscillator.

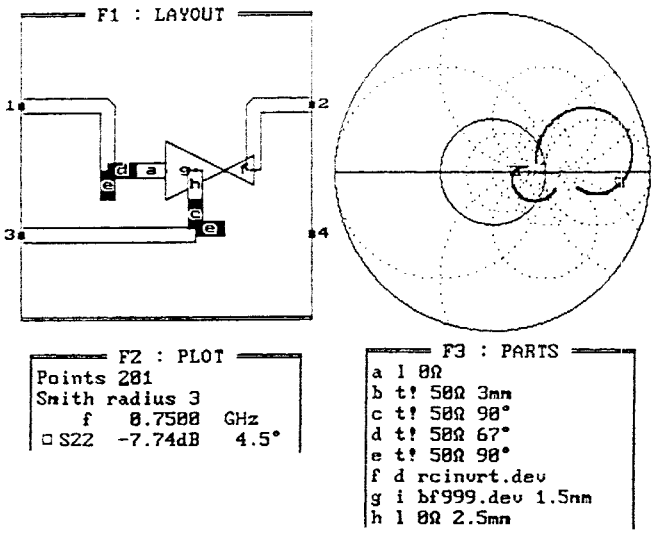


Fig. 3. Frequency response of s_{22} with and without reflection factor converter.

the design frequency. Note that the transmission lines in the PARTS window are signified with "t!" which means that losses are taken into account The lumped element h(0 Ω 2.5 mm) is necessary in order to give the right gap for the transistor in the layout. At the gate of the transistor we have added a resistor (lumped element "a" 0 Ω) in order to be able to simulate transistor saturation as discussed above.

In Fig. 4, the resistor "a" is chosen to 60 Ω. The frequency response of $1/s_{22}$ is now passing the centre of the Smith Chart. Adjustment of the line length at the input to 70° fulfils the oscillator condition at the design frequency of 750 MHz. Fig. 5 shows the microstrip design on a Euroboard (160 mm × 100 mm) where resistor and reflection coefficient inverter are removed of course. The details near the transistor cannot be seen because it is so small. At the output (port 2), a bias circuit was added consisting of a 90° transmission line grounded by

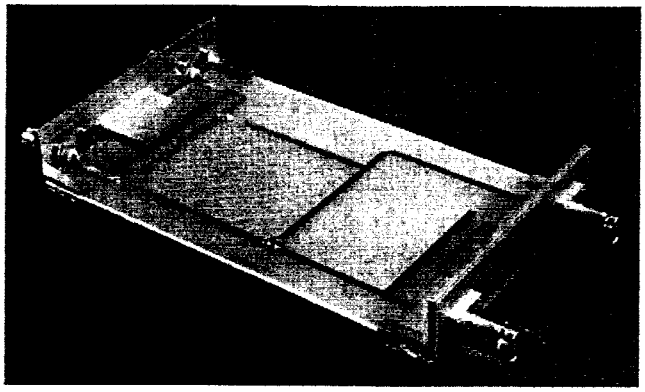


Fig. 7. Photograph of the realized oscillator.

a 27-pF capacitor and coupled to the output by another 27-pF capacitor. Note that the counter clockwise frequency response loop of s_{22} is now to be found in a totally different region of the extended Smith Chart due to the additional phase shift of line "i." Fig. 6 shows the simple layout of the oscillator. Only the little square in the bottom right corner has to be connected to ground by a pin through a drilled hole. The gaps have to be bridged by surface mounted devices (capacitors and the transistor BF999).

Fig. 7 shows a photograph of the realized oscillator together with the simple BNC adapters which are soldered to the microstrip lines on top of the board and make contact to the ground plane by means of fingerstrips. This technique works well for frequencies below 1 GHz. Fig. 8 shows a spectrum analyzer measurement of the oscillator frequency. Due to nonlinear effects the measurement result deviates slightly from the design frequency which can be corrected by experimental trimming of the length of the gate line.

V. Conclusion

The principle of a very simple oscillator circuit was ex-

plained using h-parameters with a first-order MOSFET transistor model. A design procedure for the realization of this oscillator in microstrip (Euroboard, $f = 750$ MHz) was presented in a step by step approach using the Caltech simulation and layout program PUFF. This material was developed for an undergraduate course "Microwave Technology" to be held in the Department of Electronic and Electrical Engineering, De Montfort University, Leicester, U.K.

REFERENCES

[1] D. Rutledge, "Transistor Circuits," in *Fields and Circuits*. Pasadena, CA: California Inst. Technol., 1994, ch. 12.
[2] S. W. Wedge, R. Crompton, and D. Rutledge, *PUFF—Computer Aided Design for Microwave Integrated Circuits (Version 2.0)*. Pasadena, CA: California Institute of Technology, 1991.
[3] G. Vendelin, A. M. Pavio, and U. L. Rohde, *Microwave Circuit Design Using Linear and Nonlinear Techniques*. New York: Wiley, 1990.
[4] http://www.infineon.com/products/discrete/pdf/bf999.pdf
[5] M. W. Medley, Jr., "Microwave and RF circuits," *Analysis, Synthesis and Design*. Boston, MA: Artech House, 1993.

Horst Nielinger (M'87–SM'94) was born in 1935. He received the Dipl.Ing. degree in telecommunications from Technical University Karlsruhe, Germany, in 1960 and the Dr.-Ing. degree from Technical University Munich, Germany, in 1967.

From 1960 to 1970 he was with AEG-Telefunken as Development Engineer, later as head of the Transmitter Department. Since 1970 he has been Professor for Telecommunications at the Fachhochschule Furtwangen, Germany. During 1978 to 1980 he was with the Faculty of Engineering, University of Dar es Salaam, Tanzania, and was subsequently invited again as Guest Professor in 1982, 1985, 1987, and 1988. In the first semester of 1991 he was Visiting Senior Teaching-Fellow at Curtin University of Technology, Perth, West Australia. From March to August 1996 he was a Visiting Professor at De Montfort University, Leicester, U.K. He is coauthor of *SPICE* (Berlin, Germany: Springer-Verlag, 1985) in German which was translated into Dutch in 1988.