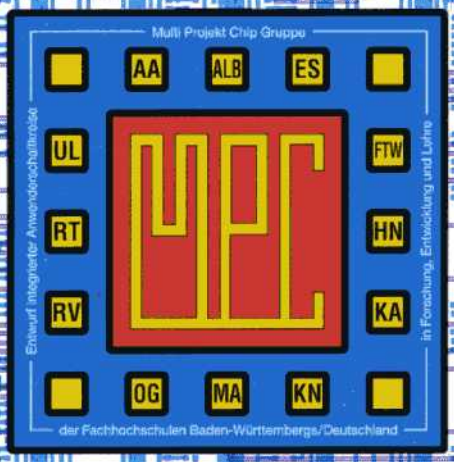
MULTIPROJEKT CHIP-GRUPPE

BADEN - WÜRTTEMBERG

Workshop Januar 1999

Furtwangen



MULTIPROJEKT CHIP-GRUPPE

BADEN - WÜRTTEMBERG

Workshop Januar 1999

Furtwangen

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 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

Gefertigte Bausteine

10 Gbit/s: Multiplexfunktionen mit VHDL

Digitale Verarbeitung von STM-64/STS-192 Rahmenstrukturen

Dipl. Ing. F. Stockmayer
Wandel & Goltermann GmbH
Tel.: 07121-86-1811
E-Mail: friedemann.stockmayer@wago.de

1 Einleitung

Die rasante Entwicklung im Bereich der Telekommunikation treibt den Bedarf an Übertragungsbandbreite in fast schwindelerregende Größenordnungen. Eine vor knapp 10 Jahren als ausreichend empfundene Bitrate von 155 Mbit/s wird heute um das 64-fache übertroffen. Jedoch erst die Einführung geeigneter Standards ermöglichten diese Bitratenexplosion. So bilden Übertragungsnetze nach dem SONET-Standard (Synchronous Optical Network) in den USA, bzw. die Synchrone Digitale Hierarchie (SDH) in Europa das Rückgrat im Weitverkehrsbereich.
Gleichzeitig wachsen aber auch die Anforderungen an die Messtechnik: Die zur Installation, Überwachung und Wartung der komplexen Übertragungssysteme notwendige digitale Signalverarbeitung bedingt einen hohen Parallelisierungsgrad (mindestens 128 Bit Busbreite) und Taktfrequenzen bis zu 80 MHz.
Der vorliegende Beitrag zeigt unter diesen Randbedingungen am Beispiel einzelner Messfunktionen Lösungsansätze zur Realisierung der SONET- bzw. SDH-Signalstrukturen STS192/ STM64 in digitaler Hardware unter Zuhilfenahme der Hardwarebeschreibungssprache VHDL.

2 Synchrone Digitale Hierarchie: SDH

SDH ist neben SONET ein Standard zur Datenübertragung über optische Netze. Der Standard enthält sowohl die Definition der optischen Signale, als auch eine Rahmenstruktur für die Aufnahme verschachtelter, „gemultiplexter" digitaler Verkehrsströme und legt damit einen Grundstein für die aktuelle physikalische Transportschicht. Der größte Vorteil dieses Transportmediums ist, dass es einerseits Schnittstellen zu bisherigen Systeme der plesiochronen digitalen Hierarchie (PDH) anbietet, als auch den Zugang moderner ATM-Übertragungstechnik gewährleistet.
Weitere Vorteile sind:

- Hohe Übertragungsraten
- Vereinfachte Add & Drop Funktionalität zum leichten Extrahieren und Einfügen von Kanälen kleinerer Bitrate.
- Hohe Verfügbarkeit und Kapazitätsauslastung durch zentrale Steuerung und Netzwerk Management.

Hierarchie SONET	Hierarchie SDH	Bitrate Mbit/s	Anzahl Sprachkanäle
STS-1		51.84	672
STS-3	STM-1	155.52	2016
STS-12	STM-4	622.08	8064
STS-48	STM-16	2488.32	32256
STS-192	STM-64	9952.00	129024

Tabelle 1: Bitraten aus SDH und SONET
STS: Synchrones Transport Signal
STM: Synchrones Transport Modul

Die folgenden Abschnitte geben eine kurze Zusammenfassung über Netzaufbau und Signalstruktur. Zwar ist bei der Standardisierung die "Gateway-Problematik" (Netzübergang mit unterschiedlichen Standards) berücksichtigt, dennoch gibt es im Detail Unterschiede, auf die hier jedoch nicht eingegangen werden soll. Der Sprachgebrauch sei daher auf die "SDH-Welt" begrenzt.

Netztopologie

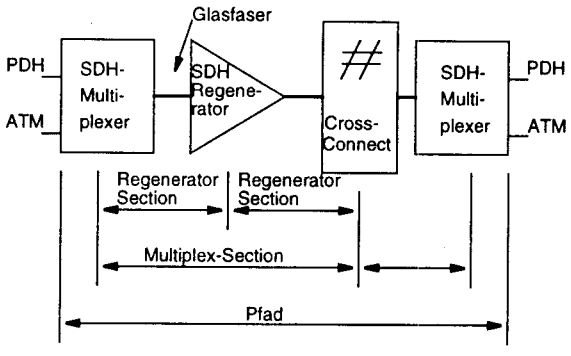


Bild 1: Übertragungsabschnitte eines SDH-Netzes

Auch SDH-Netze sind entsprechend dem Schichtenmodell unterteilt. Stellvertretend für das Übertragungsmedium, z.B. Glasfaser, repräsentiert die physikalische Ebene die unterste Schicht dieses Modells (vgl. Tabelle 2).

2 Mbit/s	ATM	IP
VC-12 Schicht		
VC-4 Schicht		
Multiplex Section		
Regenerator Section		
Physikalische Schnittstelle		

Tabelle 2: Schichtenmodell

Regenerator Section:
Beschreibt den Abschnitt zwischen zwei Regeneratoren. Ein Teil des Overhead innerhalb der Rahmenstruktur (s.a. Bild 2) ist für Signalisierung reserviert. Der Begriff "Overhead" bezeichnet dabei eine Anzahl Bytes, welche dem eigentlichen Nutzsignal hinzugefügt werden. In dem sog. Regenerator Section Overhead (RSOH, vgl. Tabelle 3 und 4) sind separate Kanäle für das Netzmanagement reserviert oder für Mechanismen zur Erkennung von Übertragungsfehlern in einem Netzabschnitt.

Multiplex Section:
Dieser Abschnitt umfasst den Anteil einer SDH-Verbindung zwischen den Multiplexern. An deren Enden stehen die Träger der Payload, das ist die eigentliche Nutzlast, als sogenannte "Virtuelle Container" (VC) zur Verfügung. Auch hier ist ein Teil des Overhead für spezielle Aufgaben freigehalten: Der Multiplex Section Overhead, MSOH, dient einerseits zur Fehlererkennung und andererseits zur Steuerung einer automatischen Ersatzschaltung beim Ausfall einer Strecke.

VC-Schicht:
Die beiden VC-Schichten stehen stellvertretend für einen Teil des Mapping Prozesses. Als Mapping wird die Anpassung von Tributary-Signalen, wie z.B. PDH- und ATM Signale, an die SDH-Transportmodule bezeichnet ggfs. mit einem Stopfalgorithmus zur Angleichung der Zubringer-Bitrate an die Container-Bitrate im SDH-Rahmen. Dem so entstandenen Container wird ebenfalls ein Overhead hinzugefügt, der über den gesamten Pfad der Ende-zu-Ende-Verbindung nicht mehr verändert wird. Container und Pfad-Overhead (POH) bilden zusammen den "Virtuellen Container".
Das VC-4 Mapping erlaubt den Einbau von 140 Mbit/s Signalen, während VC-12 das Einmappen von 2 Mbit/s-Signalen ermöglicht.

Rahmenaufbau STM-1
In der ITU-T Empfehlung G.707 ist der Rahmenaufbau mit der Grundbitrate von 155.52 Mbit/s definiert. Der Rahmen setzt sich zusammen aus einer Byte-Matrix von 9 Zeilen und 270 Spalten. Die Übertragung erfolgt zeilenweise und in jeder Zeile beginnend mit dem Byte in der Spalte 1.
Die Rahmenwiederholzeit beträgt 125 µs.

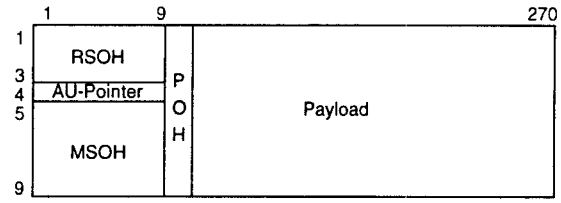


Bild 2: Rahmenaufbau STM-1 nach ITU-T G.707

Die ersten 9 Bytes in jeder Zeile werden als Overhead bezeichnet und nach RSOH und MSOH getrennt, vgl. Tabelle 3. Die Funktionen der einzelnen Bytes beschreibt Tabelle 4.

A1	A1	A1	A2	A2	A2	J0		
B1			E1			F1		
D1			D2			D3		
Au Pointer								
B2	B2	B2	K1			K2		
D4			D5			D6		
D7			D8			D9		
D10			D11			D12		
S1					M1	E2		

Tabelle 3: STM-1 Overhead

A1, A2	Rahmensynchronisation A1=0xF6, A2=0x28
B1, B2	Parity-Bytes zur Qualitätsüberwachung
D1 ... D3	Netzmanagement
D4 ... D12	Netzmangement
E1, E2	Sprech- und Dienstkanal
F1	Wartung
J0	Fortlaufende Kennzeichnung der Rahmen
K1, K2	Steuerung der automatischen Ersatzschaltung
S1	Kennzeichnung der Taktqualität
M1	Rückmeldung von Übertragungsfehlern

Tabelle 4: Funktionen der Overhead Bytes (SOH)

Transport von PDH- und ATM-Signalen
Die einzelnen Schritte zum vollständigen STM-1 Signal zeigt Bild 3. Für jedes PDH-Zubringersignal existiert ein spezieller Container C-n. Der Container ist in der Lage, deutlich mehr Übertragungskapazität aufzunehmen, als im Mittel zum Transport der Payload notwendig ist. Durch diese Eigenschaft können dann beim Einmappen durch Bitstopfen Taktabweichungen der Zubringer ausgeglichen werden.
Nach Hinzufügen des POH entsteht daraus der "Virtuelle Container" VC-n. Über den gesamten VC wird eine Prüfsumme gebildet (Even Parity) und das Ergebnis in das B3-Byte des POH eingetragen. So können am Empfangsort beim "Auspacken" des Containers Übertragungsfehler entdeckt werden.
Der nächste Schritt beschreibt eine Spezialität der SDH-Netze: Der VC-n wird über einen Pointer in den STM-1 Rahmen einghängt. Der Pointer zeigt dabei auf den Start des VC-n (erstes POH-Byte). VC-n und Pointer bilden die "Administrative Unit" AU-n bzw. "Tributary Unit" TU-n.
Mehrere solche TU-n werden in einer TU-Group TUG zusammengefasst und z.B. in einen VC-4 eingepackt, wodurch eine zweite Pointerebene entsteht (AU-Pointer).
Am Ende des komplexen Signalaufbaus kommen jetzt noch die Overheadbytes hinzu. Das B1-Byte enthält beispielsweise eine Prüfsumme über den kompletten STM-1 Rahmen. Als Bestandteil des RSOH kann dann jeder Regenerator das Signal auf Fehlerfreiheit überprüfen.

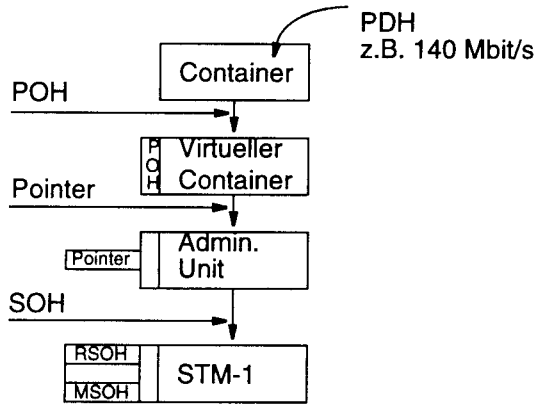


Bild 3: Beispiel eines Rahmenaufbaus

Höhere Bitraten
Reicht nun die Übertragungskapazität einer AUG nicht aus, werden mehrere AUGs byteverschachtelt (interleaved) gemultiplext. Entsprechend dem Multiplexfaktor N entsteht daraus ein STM-N Rahmen (N=1,4,16,64).

Mit N=4 wäre man somit in der Lage vier Datenströme zu je maximal 150 Mbit/s zu übertragen. Für ATM ist nun eine Bandbreite von 150 Mbit/s selten ausreichend. Der Standard sieht dafür eine Verkettung von mehreren VC-4 vor. Innerhalb eines STM-4 entsteht dadurch ein zusammenhängender Datenstrom von ca. 600 Mbit/s.

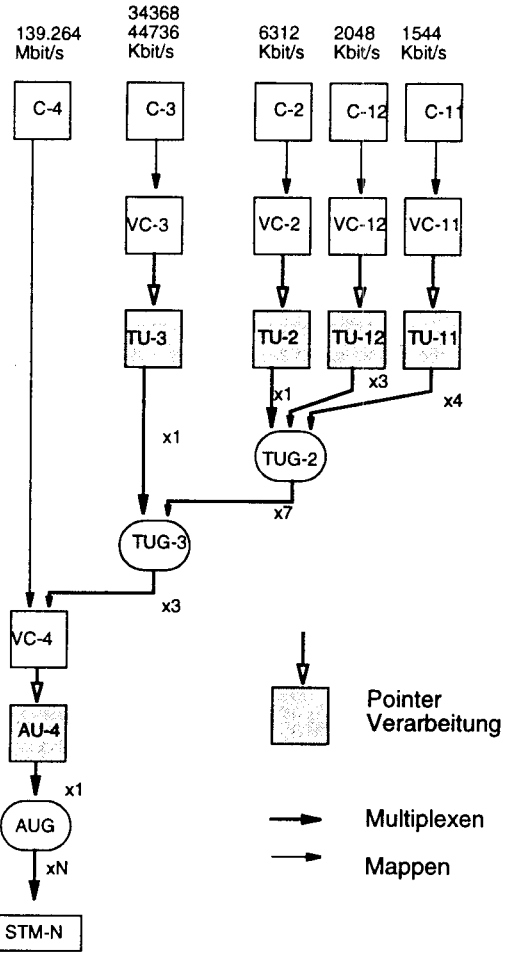


Bild 4: Ausschnitt aus der "SDH-Multiplex-Spinne"

Die Struktur des STM-N Rahmens besitzt prinzipiell die N-fache Größe eines STM-1 Rahmens. Der Overhead eines STM-64 beinhaltet z.B. insgesamt 5184 Bytes. Es liegt zwar nahe, sich den SOH dann auch aus 64-Teiloverheads zusammengesetzt vorzustellen, doch wie aus Bild 3 ersichtlich, wird der SOH am Schluss als kompletter Block hinzugefügt, bevor der Rahmen den Multiplexer verlässt.
Mit Hilfe der Parity-Bytes B2 lässt sich zwischen zwei Multiplexern, und über mehrere Regeneratoern hinweg, eine Fehlerrate berechnen. Um auch bei höheren Bitraten gleiche Verhältnisse zu schaffen, wird auch die Anzahl der B2-Byte erhöht (192 Bytes bei STM-64). Weil zwischen zwei benachbarten Regeneratoren eine kleinere Fehlerrate zu erwarten ist, ist das B1-Byte nach wie vor nur einfach vorhanden.

Der Scrambler

Jedes Netzelement ist in der Lage, seinen Systemtakt aus dem ankommenden Signal abzuleiten. Für die Taktrückgewinnung ist es daher wichtig, eine möglichst gleichmäßige Verteilung der '0'- und '1'-Bits im seriellen Datenstrom vorzufinden. Aus diesem Grund ist der Datenstrom verscrambelt, jedoch ausgenommen die erste SOH-Zeile mit den Bytes zur Rahmensynchronisation. Bild 5 zeigt die Realisierungsvorschrift (Prinzip) des bitseriellen Scramblers.

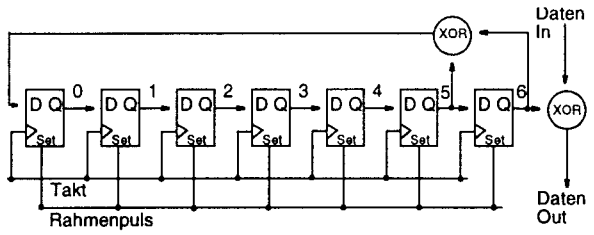


Bild 5: Bitserieller Scrambler

Parity B1

Die B1-Bildung erfolgt über den gesamten verscrambelten STM-N Rahmen in der Weise, dass alle Bytes miteinander XOR-verknüpft werden. Das Ergebnis wird dann in den nächsten Rahmen vor dem Verscrambeln eingetragen.

Parity B2

Die B2-Bildung spart die ersten 3 Zeilen des Overhead (RSOH) aus. Berechnen und Einsetzen des Ergebnisses geschieht vor dem Verscrambeln.

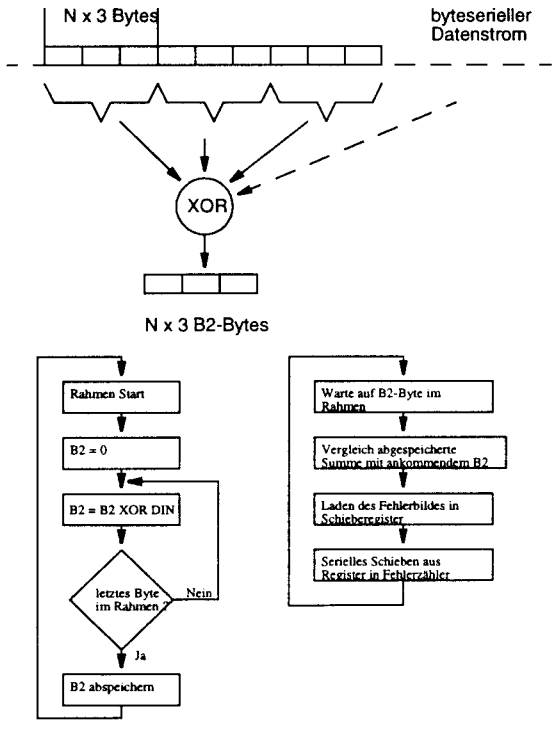


Bild 6: Prinzip der B2-Bildung

3 Messaufgaben

Das Bild 7 zeigt stark vereinfacht eine SDH-Ringstruktur mit verschiedenen Zubringern. Typisch ist dabei die Mischung von unterschiedlichen Anwendungen, die über SDH transportiert werden. Für die Messtechnik bedeutet dies eine große Vielzahl von Messaufgaben. In SDH-Netzen sind neben der Parity-Überwachung in B1 und B2 in den Overheads Alarm- und Fehlermeldungen integriert (Defekte bzw. Anomalien). Diese Meldungen sind an entsprechende Netzabschnitte gekoppelt.
Erkennt ein Regenerator z.B. eingangsseitig einen Signalausfall (Loss Of Signal), sendet er ausgangsseitig den Alarm MS-AIS (Multiplex Section Alarm Indication Signal). Das entspricht einer Dauer '1' im Rahmen, ausgenommen der RSOH.
Im Fehlerfall greift auch ein spezieller Mechanismus in SDH-Netzen: Die fehlerhafte Verbindung wird automatisch ersatzgeschaltet. Die Steuerung erfolgt mittels den Overheadbytes K1 und K2. Eine Umschaltung muss innerhalb von 50 ms erfolgen. Um dies sicherzustellen, kommen externe Messgeräte zum Einsatz. Die Missachtung dieser Randbedingung führt einerseits zu Qualitätseinbußen der Dienstleistung bzw. zu erheblichen Einnahmeausfällen beim Netzbetreiber.
Daraus leiten sich u.a. folgende generelle Funktionsblöcke in der zu realisierenden Hardware ab:

Sender:

- Multiplexer STM-1 ... STM-64
- SOH hinzufügen (teilweise aus RAM)
- Bilden und Einfügen von B1 und B2
- Einblenden von Fehler in B1 und B2
- Scrambler

Empfänger:

- Rahmensynchronisation
- Descrambler
- SOH terminieren, d.h. Auswerten der Alarme abspeichern in RAM zur Off-Line Analyse
- Auswerten der B1 und B2-Parity
- Demultiplexen STM-1 ... STM-64

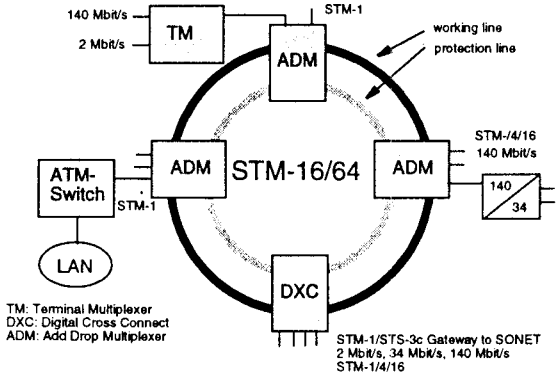


Bild 7: Prinzip eines hybriden Übertragungsnetz

4 Realisierung

Die wichtigste Frage bei der Konzeptfindung lautet: "Wie kann mit der zur Verfügung stehenden Technologie ein Datenstrom von 10 Gbit/s (STM-64) verarbeitet werden ?" Die Lösung dieses Problems liegt in einer mehrstufigen Parallelisierung und gleichzeitiger Aufspaltung in mehrere Teildatenströme, die idealerweise identisch zu bearbeiten sind. Das Resultat ist nicht nur eine Reduzierung der Komplexität, es erlaubt eine sehr effiziente Hardwareentwicklung: Die Funktionalität wird nur für einen Teildatenstrom entwickelt, die Parallelisierung entsteht anschließend durch "Copy-Paste".

Empfängerkonzept

Die erste Stufe (Chip1) im Blockdiagramm in Bild 8 wandelt den 128-Bit breiten Datenbus mit 80 MHz Taktfrequenz in vier 32-Bit breite Teildatenströme so, dass jeder Teilstrom einen zusammenhängenden STM-16 repräsentiert.
In der zweiten Stufe bearbeitet nun jeder Baustein (Chip2 ... Chip5) voneinander unabhängig einen STM-16 Teildatenstrom. Notwendig ist hier zudem eine weitere Reduzierung der Taktfrequenz auf 40 MHz und 64 Bit Datenbusbreite unmittelbar am Eingang der DMUX-Bausteine.

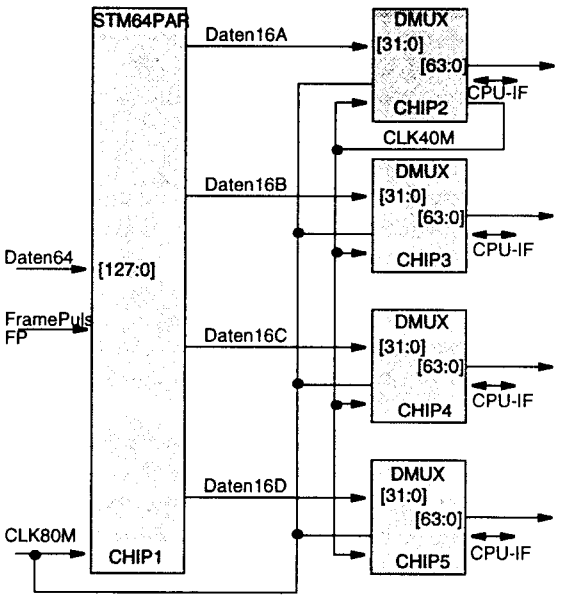


Bild 8: Baugruppen des Empfängers

Einen optimalen Schaltungsentwurf mit Hilfe von VHDL und anschließender Synthese erhält man, indem vorab die Hardwarestrukturen (Abtaststufen bzw. Flip-Flops) festgelegt werden. Grundlage für einen VHDL-Entwurf bildet dann ein Impulsdiagramm, beispielhaft gezeigt in Bild 9. VHDL bietet ja nicht nur die Möglichkeit auf sehr hohem Abstraktionslevel ein Modell zu beschreiben, besonders auf Register Transfer Ebene kann man das Synthesewerkzeug durch entsprechenden VHDL-Code so anleiten, dass zum Schluss auch die "eigene Schaltung" wiedererkennbar ist.
Im Anhang 1 ist der VHDL-Code zum Chip1 wiedergegeben. Auffällig ist darin das Registerintesive Pipelining mit geringer kombinatorischer Tiefe, um die geforderte Taktfrequenz zu erreichen.

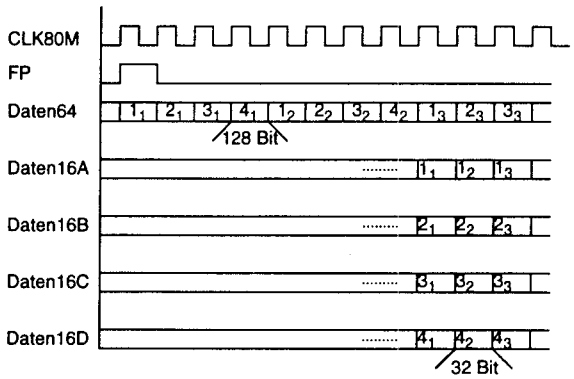


Bild 9: Impulsdiagramm für STM64PAR
Aufteilung STM-64 auf 4xSTM-16

DMUX: 1 Design, 4fach Instanziert

Folgende Funktionsblöcke sind enthalten:

- Taktteilung: Von 80 MHz auf 40 MHz
- Parallelisierung 32 Bit auf 64 Bit
- Descrambler
- B2-Bildung und Fehlerauswertung
- Zähler für B2-Fehler
- RAM zum Abspeichern des SOH
- Schnittstelle zu CPU

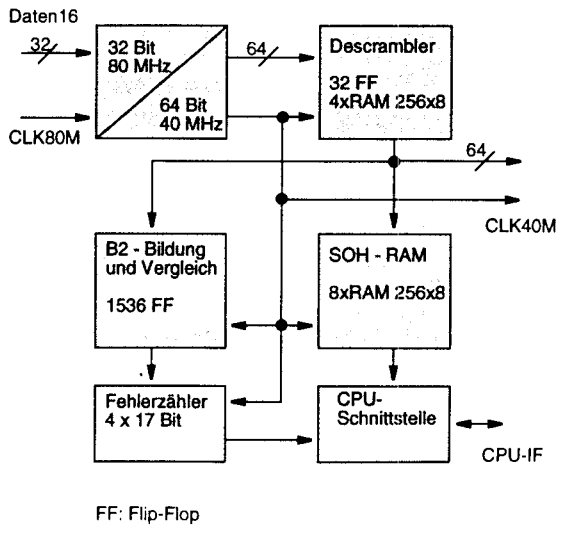


Bild 10: Blockschaltbild DMUX

Taktteiler

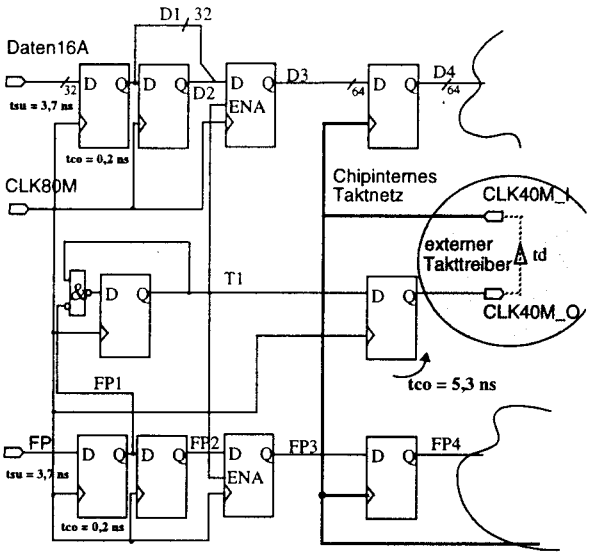


Bild 11: Taktteilung und Parallelisierung im DMUX

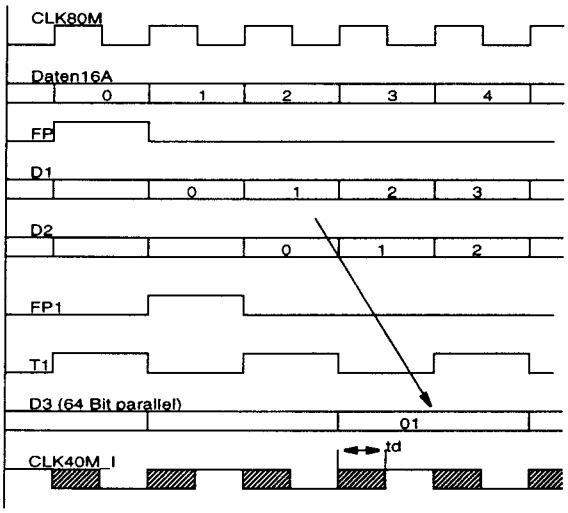


Bild 12: Timingdiagramm des Taktteilers

Der geteilte Takt aus Chip2 versorgt alle anderen Bausteine mit dem Takt 40MHz. Um auch im Chip2 gleiche Timingbedingungen zu schaffen, wird der herausgeführte Takt über einen externen Takttreiber auf das interne Taktnetz geführt.

Descrambler

Die Realisierung des Descramblers ist eine knifflige Angelegenheit: Ursprünglich über den gesamten STM-64 Datenstrom verscrambelt, "sieht" jeder DMUX davon nur ein Viertel.
Aus der kontinuierlichen Scramblerfolge muss jeder DMUX-Baustein 16 unmittelbar hintereinanderliegende Scramblerbytes berechnen mit einem anschließenden Sprung über 58 Scramblerbytes hinweg (Scramblerbytes, die parallel in den anderen drei Bausteinen anliegen).
Ausgehend von der bitseriellen Vorschrift des Scramblers in Bild 5, ist ein byteserieller Srambler mit Hilfe einer Loop in VHDL schnell realisiert:

```
-------------------------------------------------------
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY BYTE_SCRAMBLER IS
 PORT (
  CLK: IN STD_LOGIC;
  SCR_START: IN STD_LOGIC;

  SCR_DATEN: OUT STD_LOGIC_VECTOR(7 DOWNTO 0)
 );
END BYTE_SCRAMBLER;

ARCHITECTURE SCRAMBLER_ARCH OF BYTE_SCRAMBLER IS
SIGNAL SCR_START1 : STD_LOGIC;
BEGIN

 P1: PROCESS(CLK40M)
 VARIABLE SCR_TEMP : STD_LOGIC_VECTOR(7 DOWNTO 0);
 VARIABLE TEMP: STD_LOGIC;
 BEGIN

  IF(CLK'EVENT AND CLK  = '1') THEN

   SCR_START1 <= SCR_START;
   IF(SCR_START1 = '0' AND SCR_START = '0') THEN
    SCR_TEMP := "00000000";
   ELSIF(SCR_START1 = '0' AND SCR_START = '1') THEN
    SCR_TEMP := "11111110";
   ELSE

    FOR i IN 0 TO 7 LOOP
     TEMP := SCR_TEMP(5) XOR SCR_TEMP(6);
     SCR_TEMP(7) := SCR_TEMP(6);
     SCR_TEMP(6) := SCR_TEMP(5);
     SCR_TEMP(5) := SCR_TEMP(4);
     SCR_TEMP(4) := SCR_TEMP(3);
     SCR_TEMP(3) := SCR_TEMP(2);
     SCR_TEMP(2) := SCR_TEMP(1);
     SCR_TEMP(1) := SCR_TEMP(0);
     SCR_TEMP(0) := TEMP;
    END LOOP;

   END IF;

   SCR_DATEN <= SCR_TEMP;

  END IF;
 cb PROCESS P1;
END SCRAMBLER_ARCH;
-------------------------------------------------------
```

Dennoch trifft diese Lösung nicht ganz "ins Schwarze", weil die Sprünge innerhalb der Scramblerfolge noch nicht berücksichtigt sind, genausowenig die Parallelisierung auf 64-Bit. Eine diskrete Umsetzung dieser Anforderung führt jedoch zu einer unverhältnismäßig großen Schaltung (ca. 4400 Logik-Zellen) und scheidet daher aus.
Eine wesentlich kompaktere Schaltung schafft die Kombination aus diskreter Logik und im Baustein verfügbaren RAM-Blöcken.
Die RAMs sind mit einem Teil der Scramblerfolge gefüllt, die nun auch Sprünge enthalten kann, der Rest liefert eine nachgeschaltete diskrete Logik (vgl. Bild 13). Diese Maßnahme liefert ein äußerst schnelles Design und nutzt die vorhandenen Resourcen optimal.

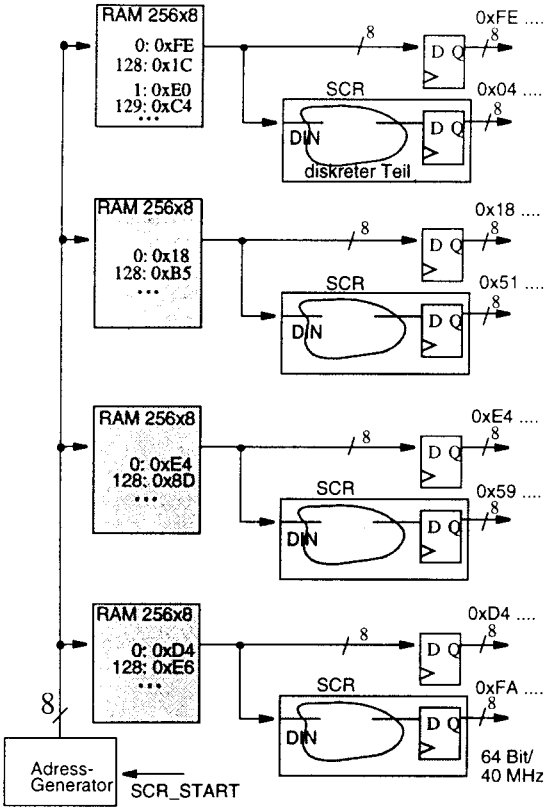


Bild 13: Aufbau des Scramblers

```
ENTITY SCR IS -- Diskreter Teil
 PORT(

   CLK40M :  IN STD_LOGIC;
   DIN:      IN STD_LOGIC_VECTOR(7 DOWNTO 0);
   DOUT :    OUT STD_LOGIC_VECTOR(7 DOWNTO 0));

END SCR;

ARCHITECTURE SCR_ARCH OF SCR IS
BEGIN

 P1: PROCESS(CLK)
 VARIABLE SCR_TEMP : STD_LOGIC_VECTOR(7 DOWNTO 0);
 VARIABLE TEMP : STD_LOGIC;
 BEGIN

 IF(CLK40M'EVENT AND CLK40M = '1') THEN

    SCR_TEMP := DIN;

    FOR i IN 0 TO 7 LOOP

       TEMP := SCR_TEMP(5) XOR SCR_TEMP(6);

       SCR_TEMP(7) := SCR_TEMP(6);
       SCR_TEMP(6) := SCR_TEMP(5);
       SCR_TEMP(5) := SCR_TEMP(4);
       SCR_TEMP(4) := SCR_TEMP(3);
       SCR_TEMP(3) := SCR_TEMP(2);
       SCR_TEMP(2) := SCR_TEMP(1);
       SCR_TEMP(1) := SCR_TEMP(0);
       SCR_TEMP(0) := TEMP;

    END LOOP;

      DOUT <= SCR_TEMP;

 END IF;
END PROCESS P1;
```

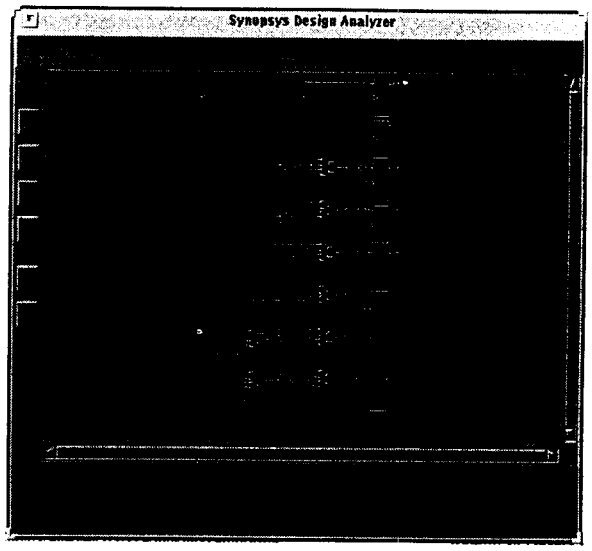


Bild 14: Synthetisierter Scrambler (diskreter Teil)

B2 - Berechnnung

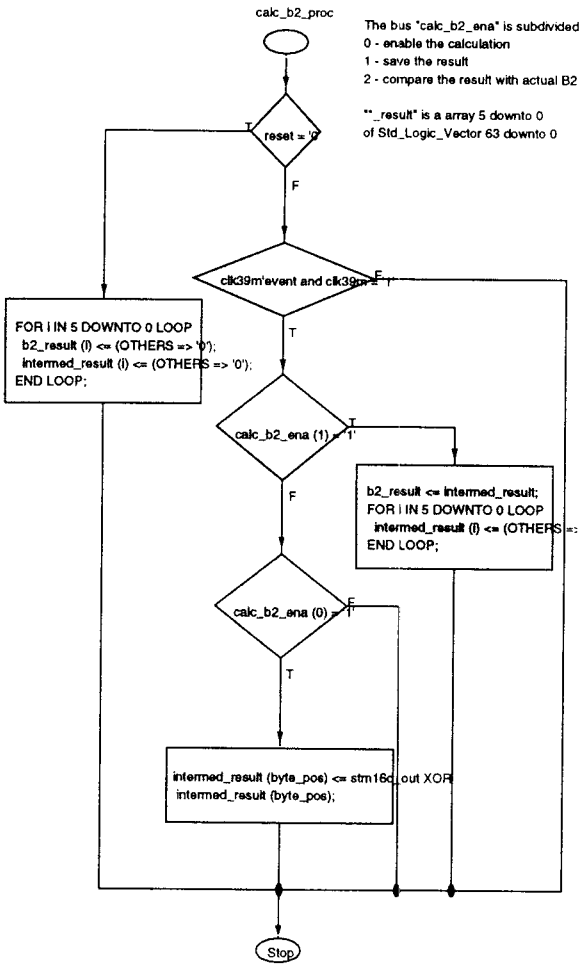


Bild 15: Flussdiagramm zur B2-Berechnung: Anwendung eines grafischen Eingabetools mit VHDL-Code Generierung

Senderkonzept

Die Funktionsblöcke des Senders sind dem Empfänger vielfach sehr ähnlich. Die Wiederverwendbarkeit bereits getesteter Komponenten kann die Entwicklungszeit erheblich reduzieren. Vorgabe in diesem Projekt war, diesen Aspekt von Beginn an entsprechend zu berücksichtigen. Nicht zuletzt durch die Unterstützung graphischer Eingabewerkzeuge, mit anschließender VHDL-Code Generierung, konnte dieses Ziel erreicht werden.
Vor allem große Blöcke, wie B2-Berechnung und Scrambler, sind im Empfangs- und Sendeteil identisch.

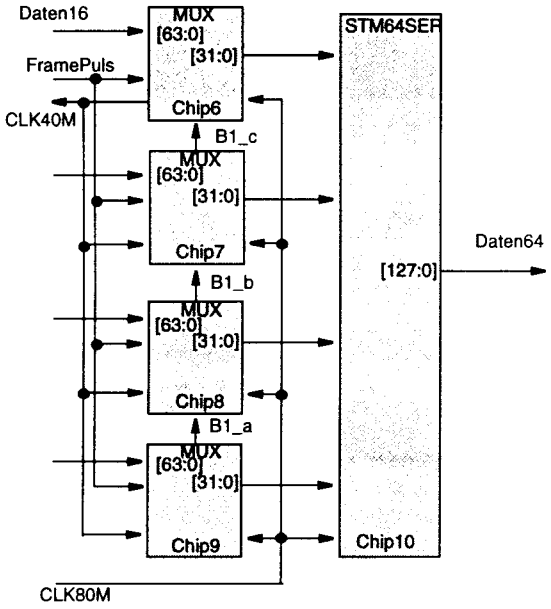


Bild 16: Blockdiagramm des Senders

Der Mux beinhaltet jeweils die Blöcke:

- Rahmengerüst STM-64 aufbauen (Unequipped) aus RAM
- Multiplexer: Addition des SOH
- B1/B2 - Berechnung
- Scrambler
- Taktteiler, Serialisierung

Als Besonderheit sei an dieser Stelle noch die B1-Berechnung erwähnt. Jeder MUX kann jeweils nur über 1/4 des Gesamtdatenstromes ein Teilergebnis ermitteln. Diese Teilergebnisse (B1_a ... B1_c in Bild 16) werden von einem Baustein zum nächsten weitergereicht, da aufgrund der Parallelisierung das B1-Byte immer im Chip6 einzubauen ist.

B1_a = D4 XOR D8 XOR ...
B1_b = D3 XOR D7 XOR ... XOR B1_a
B1_c = D2 XOR D6 XOR ... XOR B1_b
B1 = D1 XOR D5 XOR ... XOR B1_c

Register Balance

Die Synthese unterstützt aktiv die Entwicklung komplexer Logik, vor allem wenn es darum geht die Schaltung auf Geschwindigkeit hin zu optimieren. Eine große Kombinatorik, auch mit breiten Busstrukturen, kann durch die 'register_balance'-Funktion auf mehrere kleinere Blöcke aufgeteilt werden. Die Anzahl der Teilblöcke gibt ein dafür in VHDL formuliertes Schieberegister vor.

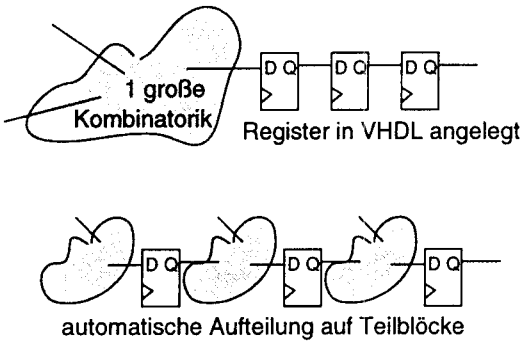


Bild 17: Register Balance

Test-Bench

Eine Test-Bench fasst alle 10 Bausteine mit insgesamt ca. 500000 Gatteräquivalente und 96 RAM-Blöcke in einer gemeinsamen Simulation zusammen. Sender und Empfänger sind in einer Schleife miteinander verbunden. Die umfangreichen Testpattern können sehr flexibel mit Hilfe eines C++-Programmes erstellt werden. Auch erfolgt die Auswertung der Simulationsergebnisse teilweise automatisch, z.B. durch Vergleich der Ergebnisfiles.

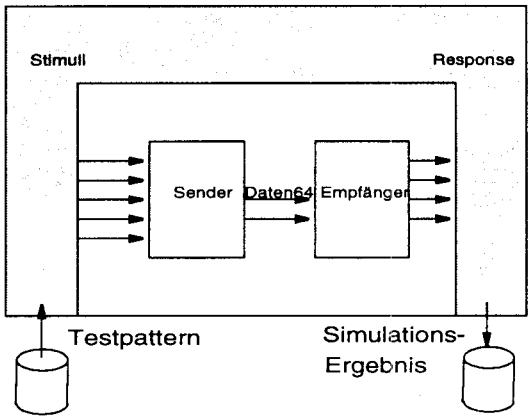


Bild 18: Test-Bench

Literatur

/1/ ITU-T Empfehlung G.707
/2/ SDH - Pocket Guide
Stephan Schultz, Wandel & Goltermann
/3/ Spezifikation STM-64
Internes Memorandum
F. Stockmayer, Wandel & Goltermann

Anhang1: VHDL Beispiel

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY stm64par IS
PORT (

 CLK80M :  IN STD_LOGIC;
 FP:       IN STD_LOGIC;

 Daten64:  IN STD_LOGIC_VECTOR(127 DOWNTO 0);

 Daten16A : OUT STD_LOGIC_VECTOR(31 DOWNTO 0);
 Daten16B : OUT STD_LOGIC_VECTOR(31 DOWNTO 0);
 Daten16C : OUT STD_LOGIC_VECTOR(31 DOWNTO 0);
 Daten16D : OUT STD_LOGIC_VECTOR(31 DOWNTO 0)
);

END stm64par;

ARCHITECTURE stm64par_arch OF stm64par IS

SIGNAL FP1, FP2 : STD_LOGIC;

SIGNAL SELA, SELB, SELC, SELD : STD_LOGIC;
SIGNAL SELA_1, SELA_2, SELA_3 : STD_LOGIC;
SIGNAL SELB_1, SELB_2, SELB_3 : STD_LOGIC;
SIGNAL SELC_1, SELC_2, SELC_3 : STD_LOGIC;
SIGNAL SELD_1, SELD_2, SELD_3 : STD_LOGIC;

SIGNAL Daten64Int : STD_LOGIC_VECTOR(127 DOWNTO 0);
SIGNAL STM16A: STD_LOGIC_VECTOR(127 DOWNTO 0);
SIGNAL STM16B: STD_LOGIC_VECTOR(127 DOWNTO 0);
SIGNAL STM16C : STD_LOGIC_VECTOR(127 DOWNTO 0);
SIGNAL STM16D : STD_LOGIC_VECTOR(127 DOWNTO 0);

SIGNAL CNT,LD : NATURAL RANGE 3 DOWNTO 0;

SIGNAL STM16SER_A : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL STM16SER_B : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL STM16SER_C : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL STM16SER_D : STD_LOGIC_VECTOR(31 DOWNTO 0);

SIGNAL TEMPA_A : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL TEMPA_B : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL TEMPA_C : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL TEMPB_A, : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL TEMPB_B : STD_LOGIC_VECTOR(31 DOWNTO 0);
SIGNAL TEMPC_A : STD_LOGIC_VECTOR(31 DOWNTO 0);

BEGIN

-- Eingangssignale abtasten
ABTAST: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  Daten64Int <= Daten64;
  FP1  <= FP;
  FP2  <= FP1;

 END IF;

END PROCESS ABTAST;

-- Synchronisation auf Rahmenpuls
FP_ALIGN:PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  IF(FP1 = '1' AND FP2 = '0') THEN
   CNT <= 1;
  ELSIF(CNT = 3) THEN
   CNT <= 0;
  ELSE
   CNT <= CNT + 1;
  END IF;

 END IF;

END PROCESS FP_ALIGN;

-- Steuersignale generieren
P1A: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  SELA_1 <= SELA;
  SELA_2 <= SELA_1;
  SELA_3 <= SELA_2;

  IF(CNT = 3) THEN
   SELA <= '1';
  ELSE
   SELA <= '0';
  END IF;

 END IF;

END PROCESS P1A;

P1B: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  SELB_1 <= SELB;
  SELB_2 <= SELB_1;
  SELB_3 <= SELB_2;

  IF(CNT = 0) THEN
   SELB <= '1';
  ELSE
   SELB <= '0';
  END IF;

 END IF;

END PROCESS P1B;

P1C: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  SELC_1 <= SELC;
  SELC_2 <= SELC_1;
  SELC_3 <= SELC_2;

  IF(CNT = 1) THEN
   SELC <= '1';
  ELSE
   SELC <= '0';
  END IF;

 END IF;

END PROCESS P1C;

P1D: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  SELD_1 <= SELD;
  SELD_2 <= SELD_1;
  SELD_3 <= SELD_2;

  IF(CNT = 2) THEN
   SELD <= '1';
  ELSE
   SELD <= '0';
  END IF;

 END IF;

END PROCESS P1D;

-- STM16 #1 zwischenspeichern
P2: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  IF(SELA = '1') THEN
   STM16A <= Daten64Int;
  END IF;

 END IF;

END PROCESS P2;

-- STM16 #2 zwischenspeichern
P3: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  IF(SELB = '1') THEN
   STM16B <= Daten64Int;
  END IF;

 END IF;

END PROCESS P3;

-- STM16 #3 zwischenspeichern
P4: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  IF(SELC = '1') THEN
   STM16C <= Daten64Int;
  END IF;

 END IF;

END PROCESS P4;

-- STM16 #4 zwischenspeichern
P5: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  IF(SELD = '1') THEN
   STM16D <= Daten64Int;
  END IF;

 END IF;

END PROCESS P5;

-- STM16 #1 serialisieren 128 Bit -> 32 Bit
P6: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  IF(SELA_1 = '1') THEN
   STM16SER_A <= STM16A(127 DOWNTO 96);
  ELSIF(SELA_2 = '1') THEN
   STM16SER_A <= STM16A(95 DOWNTO 64);
  ELSIF(SELA_3 = '1') THEN
   STM16SER_A <= STM16A(63 DOWNTO 32);
  ELSE
   STM16SER_A <= STM16A(31 DOWNTO 0);
  END IF;

 END IF;

END PROCESS P6;

-- STM16 #2 serialisieren 128 Bit -> 32 Bit
P7: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  IF(SELB_1 = '1') THEN
   STM16SER_B <= STM16B(127 DOWNTO 96);
  ELSIF(SELB_2 = '1') THEN
   STM16SER_B <= STM16B(95 DOWNTO 64);
  ELSIF(SELB_3 = '1') THEN
   STM16SER_B <= STM16B(63 DOWNTO 32);
  ELSE
   STM16SER_B <= STM16B(31 DOWNTO 0);
  END IF;

 END IF;

END PROCESS P7;

-- STM16 #3 serialisieren 128 Bit -> 32 Bit
P8: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  IF(SELC_1 = '1') THEN
   STM16SER_C <= STM16C(127 DOWNTO 96);
  ELSIF(SELC_2 = '1') THEN
   STM16SER_C <= STM16C(95 DOWNTO 64);
  ELSIF(SELC_3 = '1') THEN
   STM16SER_C <= STM16C(63 DOWNTO 32);
  ELSE
   STM16SER_C <= STM16C(31 DOWNTO 0);
  END IF;

 END IF;

END PROCESS P8;

-- STM16 #4 serialisieren 128 Bit -> 32 Bit
P9: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  IF(SELD_1 = '1') THEN
   STM16SER_D <= STM16D(127 DOWNTO 96);
  ELSIF(SELD_2 = '1') THEN
   STM16SER_D <= STM16D(95 DOWNTO 64);
  ELSIF(SELD_3 = '1') THEN
   STM16SER_D <= STM16D(63 DOWNTO 32);
  ELSE
   STM16SER_D <= STM16D(31 DOWNTO 0);
  END IF;

 END IF;

END PROCESS P9;

-- Ausrichten der Ausgangsdatenstroeme
P10: PROCESS(CLK80M)
BEGIN

 IF(CLK80M'EVENT AND CLK80M = '1') THEN

  Daten16D <= STM16SER_D;

  TEMPA_A <= STM16SER_A;
  TEMPA_B <= TEMPA_A;
  TEMPA_C <= TEMPA_B;
  Daten16A  <= TEMPA_C;


  TEMPB_A <= STM16SER_B;
  TEMPB_B <= TEMPB_A;
  Daten16B  <= TEMPB_B;

  TEMPC_A <= STM16SER_C;
  Daten16C  <= TEMPC_A;

 END IF;

END PROCESS P10;

END stm64par_arch;
```

Technology Trend in Microelectronics and its Impact on Digital Signal Processing with Emphasis on Digital Communication

Prof. Dr.-Ing. Haybatolah Khakzar, Dr. Thomas Gneiting
Fachhochschule Esslingen - Hochschule für Technik, Germany

Abstract **Microelectronics has revolutionised the role of technology in human society in the last decades. Its impact on information, communication, production and science has caused dramatic changes in human society. Stone Age, Bronze Age, Iron Age - future generations may call our time the Silicon Age inspired by the source of this technology, the controlled pollution of pure silicon. As in the last 50 years, since the invention of the first transistor, the exponential growth of integrated circuit performance is assumed to continue. Forecasts until the year 2010 predict a steady increase of the performance and size of silicon based integrated circuits, while the device size of a single transistor will be shrunk to the nanometer domain. On the other hand, the next century will be the century of information. Coding for data compression, security and error control play a great role in all types of communication. These are only possible with silicon nano technology and digital signal processing.**

1 Introduction

The history of integrated circuits shows that all their attributes such as size, integration or speed have grown at an enormous rate in the last 35 years since their invention. The future technology directions have been predicted by several organisations, in particular the Semiconductor Industry Association (SIA) in the U.S., to the year 2010 in the "National Technology Roadmap for Semiconductors" [SIA-94]. The analysis of these projections will be the basis of further investigations in this paper.

1.1 Overall trends

First the method for creating the technology roadmaps and their time range will be demonstrated. This is very important in order to compare their predictions with the results gathered within the experiments in this study.

a) Timetable

The proposals for the development of the future ULSI technology covers the next decade up to the year 2010. Such proposals are produced by extending the historic trends for dynamic random access memories (DRAM) where the capacity in bits is increased by a factor of four every three years.

Process Generation [micron]	0.8	0.5	0.35	0.25	0.18	0.13	0.10	0.07
DRAM size [Bytes]	4M	16M	64M	256M	1G	4G	16G	64G
Year of first DRAM shipment	1990	1993	1995	1998	2001	2004	2007	2010
Mature production (US)	1992	1995	1997	2000	2003	2006	2009	2012

Table 1.1 Introduction of new CMOS process generations

The survey will cover eight process generations, starting with the 0.8micron CMOS process, which was going into main production in the U.S. when this research project started in 1992. For the year 2010, it is supposed that the minimum feature size will be decreased by an order of magnitude to 0.07 microns or using another unit, to 70 nanometers. Table 1.1 shows the timetable for the eight process generations. It is divided into the introduction of a new technology in the year of the first DRAM shipment and its mature production in the world.

b) Diversification

While in the past the DRAM has been the overall measure for technological progress, this view must be revised for future technologies. It could be identified that there are other approaches to be considered because of the rapid development of new application fields. Three general types of circuits are taken to predict the way forward in semiconductor technology. These are the traditional DRAM, the microprocessor and application specific integrated circuits (ASIC). Through the very different applications, like high performance computing on the one hand and the emerging mobile communication applications on the other hand, these circuit types must additionally be divided into high performance and low power.

c) Technology drivers

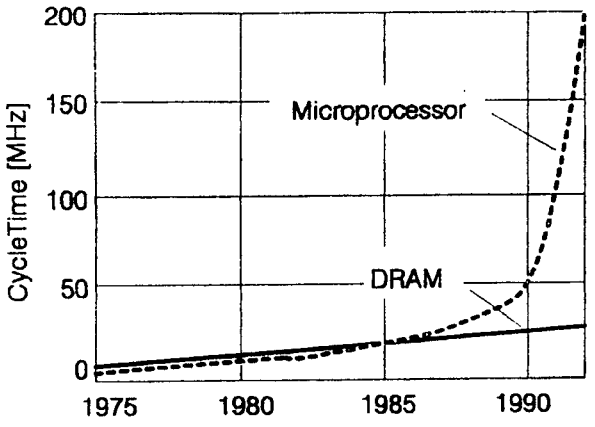


Figure 1.1 Clock frequency of µP and inverse access time of DRAMs

As pointed out above, the DRAM circuits have been the technology drivers for process development in the last two decades since these high-volume products have financed leading-edge production technology. However, more and more the logic applications such as microprocessors and ASICs are catching up as technology drivers. One reason is the rapid growth of the clock frequency of microprocessors and therefore their specific need for high performance on-chip interconnections which differentiate them from DRAMs. Figure 1.1 shows a diagram from [Com-92] where the development of the maximum clock frequency of microprocessors and the inverse access time of DRAMs is shown. It is shown clearly, that the speed of microprocessors has increased exponentially, while the access time of the DRAMs can only be increased in small steps.

2 Roadmap of ULSI technology

2.1 Trends in CMOS technology

CMOS process technology is today's dominating technology for integrated circuit manufacture. It is widely anticipated that this role will be kept in the foreseeable time range of the next two decades. The predictions related to this technology are therefor the basis of the further investigations within this paper. In the following subsections, the trends for the main technological parameters such as chip sizes, wafer sizes, chip frequency, supply voltage and process parameters are analysed.

a) Chip sizes

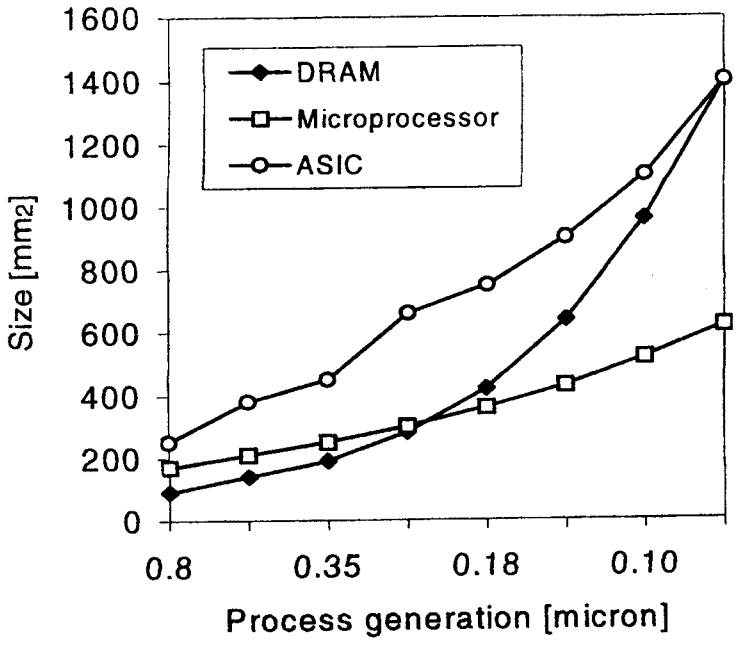


Figure 2.2 **Chip size of different circuit types as a function of process generations**

The predicted chip sizes result from the number of transistors per chip and the minimum feature size. In the case of the DRAM, the capacity in bit count is increased by a factor of four between every generation. In the same period, the area of one cell is reduced by a factor of 2.5 while the overall chip area grows by a factor of 1.5. Therefore, DRAMs are still one of the technology drivers for enhanced chip sizes. In addition, the prediction of chip sizes of application specific integrated circuits (ASIC) foresees a tremendous growth for the next 5 process generations as described in [SIA-94]. Figure 2.2 shows the proposed chip size for the 8 process generations and the three main circuit types DRAM, microprocessor and ASIC.

b) *Wafer sizes*

Due to the increasing chip sizes as shown in Figure 2.2, a conversion in the diameter of a silicon wafer appears to be necessary to achieve the required economies of scale. While a diameter of 200 mm is now state of the art, it is assumed, that future technology generations will require wafer diameters of 300 to 400 mm. The main issues of a conversion to larger diameters are the immense costs for changing the production equipment to this larger wafer diameter. The second issue that has to be considered is the flatness of such a huge plate of silicon and the contamination. Figure 2.3 gives a visualisation of the development of the wafer size during six generations of DRAM's from 64 kbit in 1968 to 4 Mbit in 1989 while Table 2.2 shows the prediction for the next process generations.

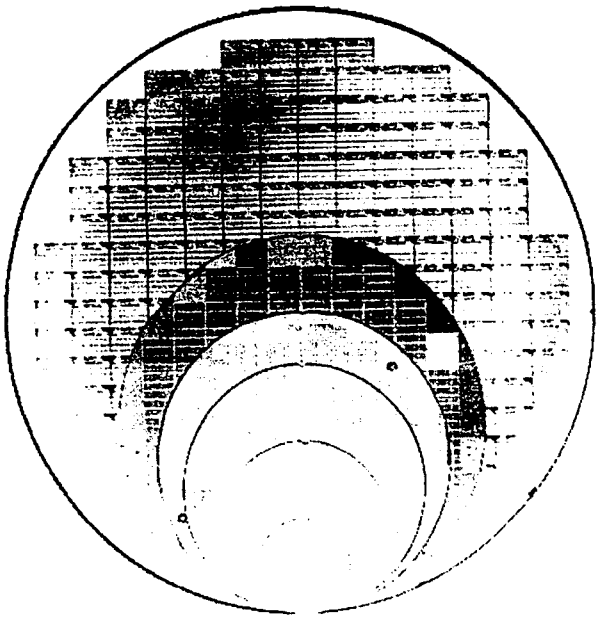


Figure 2.3 **Wafer sizes from d=32 mm (4kBit) to d=200 mm (4MBit DRAM)**

Process generation [micron]	0.8	0.5	0.35	0.25	0.18	0.13	0.10	0.07
Wafer diameter [mm]	150-200	200	200	200	300	300	400	400

Table 2.2 **Wafer diameters for 6 process generations**

c) *Chip frequency*

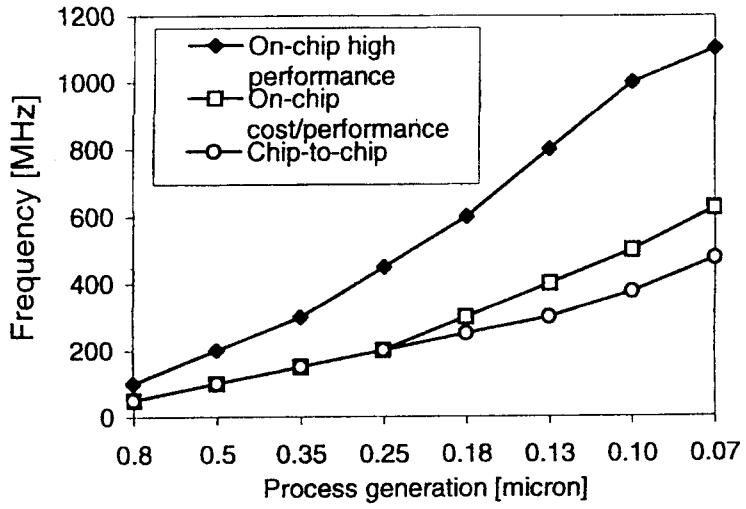


Figure 2.4 **Chip frequency for the examined process generations**

The chip frequency is mostly used as the ultimate performance indicator for digital devices. It may be divided into the on-chip clock for high performance application, (e.g. high performance processors), the on-chip clock for cost/performance (e.g. commercial ASICs), low power applications and the chip-to-board frequency, with which an integrated circuit can communicate with other circuits on a printed circuit board. Due to parasitics in the package, in bonding wires, leads and pads this chip-to-chip clock is much lower than the maximum on-chip frequency. Figure 2.4 shows all three categories for the surveyed process generations [SIA-94], [Den-95].

d) *Supply voltage*

The second important aspect of future technology parameters is the supply voltage of ULSI devices. The value of the supply voltage will affect not only the device itself but also the overall system on a printed circuit board and the integration with other logical components. For a very long time, the supply voltage V_{DD} has been kept to a constant value of 5V. Now, in the submicron regime several reasons can be found to lower the supply voltage. The first one is the scaling law. Due to the high electrical field in the gate oxide and the device degradation through hot electrons a further shrinking of the oxide thickness requires a scaling of the supply voltage. Another issue is the power management. With a large number of devices on one chip and their operation at high clock speeds a low supply voltage is required to keep the power dissipation manageable.

Device characteristic	Technology generation							
	0.8	0.5	0.35	0.25	0.18	0.13	0.10	0.07
V_{DD} high performance [V]	5	3.3	3.3	2.5	1.8	1.5	<1.2	
V_{DD} low power [V]	5	3.3	2.5	1.8	1.2	<1.2		

Table 2.3 **Predicted scaling of supply voltage V_{DD}**

The main problems induced by a reduced supply voltage are the control of the threshold voltage of a transistor and its subthreshold slope. At reduced voltages, transistor operating margins become small because threshold voltage does not scale and reducing the threshold voltage on conventional devices will increase the subthreshold current. This causes excessive standby power dissipation.

The overall trend for the supply voltage must therefore be divided into one path for high-performance circuits and another path for low power applications, where standby power dissipation may be crucial. Table 2.3 gives an overview of the expected values for V_{DD} in the observed process generations.

2.2 On-chip interconnections

While the MOS key parameters are mostly dominated by the scaling laws, dramatic changes have to be made in the area of on-chip interconnections. The traditional on-chip interconnection system is the aluminium microstrip line on a silicon dioxide isolation layer. This system has been in use since the first integrated circuits was made.

However, with smaller device dimensions the Al/SiO_2 interconnect system limits the overall system performance more and more.

Due to the overall device scaling, the width of the striplines will shrink in the same way. Although the height of the lines is increased to alleviate this effect, the total cross sectional area is lowered which causes a large increase in resistance per unit length (R'). The second effect is that the isolation layer between two metal layers will also decrease to keep the contact/via aspect ratio to a reasonable level. Due to this thinner isolation layer, the capacitance per unit length of a transmission line will also increase. Table 2.4 shows the forecast of the development of transmission line parameters according to the mechanism described.

Device characteristic	Technology generation							
	0.8	0.5	0.35	0.25	0.18	0.13	0.10	0.07
Minimum line width [µm]	0.9	0.6	0.40	0.30	0.22	0.15	0.11	0.08
Metal height/width aspect ratio	0.5:1	1:1	1.5:1	2:1	2.5:1	3:1	3.5:1	4:1
R' [Ω/µm] *)	0.12	0.13	0.15	0.19	0.29	0.82	1.34	1.34
C' [fF/µm] *)	0.13	0.15	0.17	0.19	0.21	0.24	0.27	0.27
Wire material	Al			thick Al, Cu				
Isolation material	SiO_2			SiO_2, organic or low ε dielectric				

*) assuming the Al/SiO_2 material system

Table 2.4 Electrical and material properties of future on-chip interconnections

With the propagation mode for Al/SiO_2 interconnection systems, the delay constant τ for the signal propagation delay on these transmission line is

$$\tau = R \cdot C = R' \cdot C' \cdot L^2 \qquad (2.1)$$

With the values for R' and C' for different process generations from Table 2.4, equation (2.1) gives an insight into the evolution of signal propagation speed. It can be demonstrated that the propagation velocity of the interconnections are slowed down with every new process generation while the switching speed of transistors is steadily increased.

2.3 Chip-to-chip interconnections

The traditional approach for chip-to-chip interconnections is the printed circuit board (PCB). However, regarding the required chip-to-chip frequencies, which are shown in Figure 2.4, there is clearly a different limit for the packaged chips mounted on a PCB. The limitation of chip-to-PCB speed is the package lead inductance and track capacitance. A contemporary typically loaded value for an IC lead is 10nH inductance and 10pF capacitance with a resulting resonant frequency of :

$$f_0 = \frac{1}{2\pi\sqrt{LC}} = 503\text{MHz} \qquad (2.2)$$

A rectangular wave signal will typically be limited to less than one-third of this frequency, in this case 160 MHz which is a significant limitation. A critical need is the reduction of inductance and capacitance at the chip input/output ports. Multichip Module Technology with deposited layers (MCM-D) is one of the major emerging strategies addressing the need for low inductance and capacitance in the chip-to-chip connection.

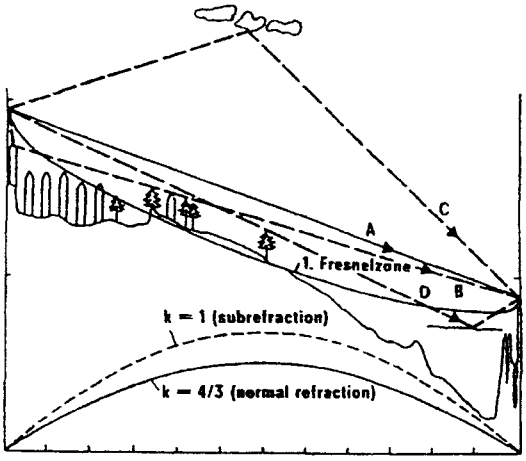
For this reason, the development of the MCM-D technology will be discussed in coming years. In contrast to the pure semiconductor technology where forecasts of technology developments are available, such a roadmap is not authoritatively published for MCM-D technology. Significantly there is a school of thought which suggests that MCM-type deposited interconnect may prove an important technology to replace the AL/SiO_2 interconnect methods on large area ULSI devices.

the need for a standard device model for a certain technology, (e.g. the CMOS technology) is clearly demonstrated. Such a universal standard model may not be as good as certain specialised models, but the inter-operability between different fabs, simulators and designer groups will provide a considerable benefit in characterisation and modelling work.

Modelling the effects of fluctuations during the manufacturing process of integrated circuits is a growing field. Statistical design techniques to avoid parametric yield loss or to predict the spread of circuit performance are strongly dependent on an accurate model of these fluctuations. In [TCA-95] four levels of statistical effects are identified. The lowest level are the process inputs like doses, implanting energies, temperatures etc., followed by process parameters like sheet resistance, oxide thickness or doping depths. The third level is represented by the compact model parameters and the fourth level is the overall circuit performance. The need is now to build a statistical model on the level of process parameters or compact model parameters in order to predict the behaviour of the overall circuit performance. The crucial point of statistical modelling techniques is that the derived statistical models must mesh with existing compact models and be as easy to use as the standard compact models already mentioned.

For future process generations, more attention has to be given to the accurate modelling and simulation of the interconnections on integrated circuits. Whilst today about 50% of the cycle-time will be dominated by the interconnection delay, it is expected that this may rise to 70% to 80% for future deep submicron processes [TCI-95]. Therefore, an improvement on the traditional approach of representing interconnections through distributed RC structures is needed in order to address signal propagation and integrity in dense, lossy interconnection technologies.

3 Impact on digital communication



A Direct path
B Fixed reflexion by buildings
C Tropospheric scattering and partial reflexion
D Total reflexion by a layer

Figure 3.6 Typical multipath propagation

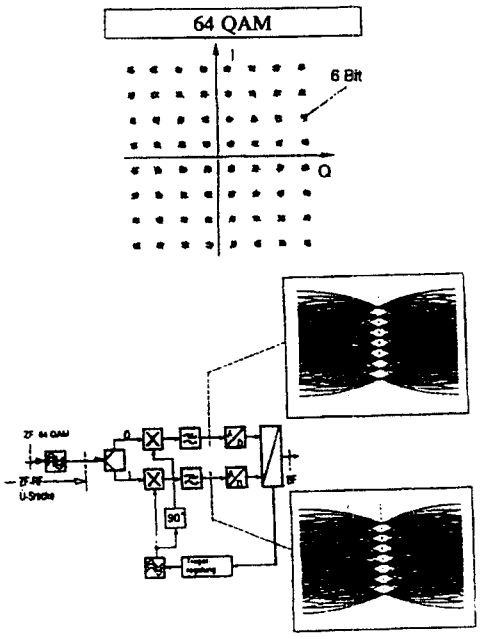


Figure 3.7 64 QAM system

Figure 3.6 shows a mobile communication system. A 64 QAM system, which is now in operation is given in **Figure 3.7**.

Figure 3.8 shows the QAM processor block diagram [DRI-98] of the described system. The total chip area is 185mm^2 and the number of transistors is 7785.380. In 2 years more than 1 billion mobile phones will exist all over the world. The realised quality of the mobile communication was only possible with the implementation of silicon nano device technology together with digital signal processing.

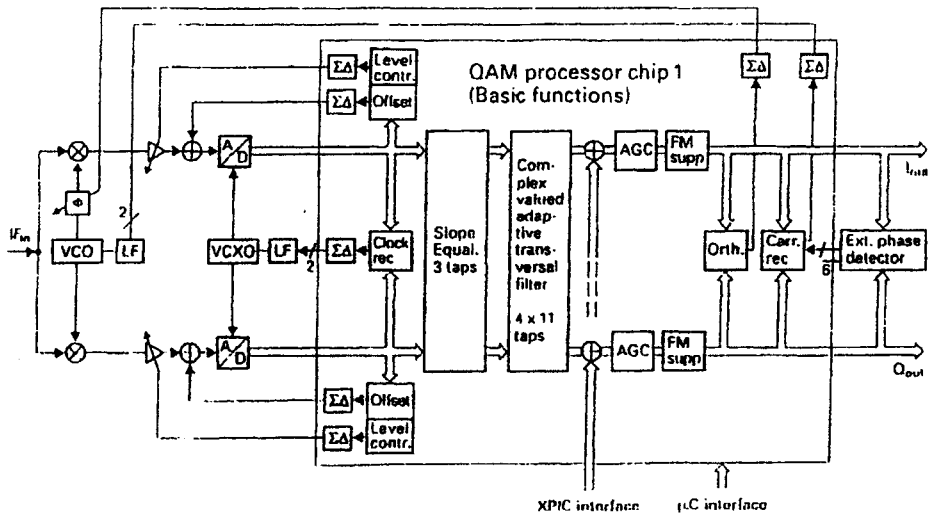


Figure 3.8 QAM processor block diagram

4 Summary

Looking at CMOS processing technology, it becomes clear that the aggressive scaling trend of device miniaturisation will keep on going for the next six forseeable process generations. This development is not only driven by the classical fields of high performance computing or DRAM technology but also by the needs of new applications like multimedia or mobile communication for an increasing amount of processing power or memory on a relatively small piece of silicon. However, looking at the roadmap for predictions of chip-performance in the form of maximum clock frequency, there are still a lot of open questions which cannot be answered by simply extrapolating the former technology development to future process and circuit generations.

Regarding CMOS technology itself, some crucial parameters especially threshold voltage, are uncertain for future technology generations since no one has produced a realistic prediction of this critical parameter. Together with scaled geometries and decreased supply-voltages the window of safe operation of a MOS device becomes even smaller.

Literature

[Com-92] R. Comerford, G. Watson, "Memory catches up", IEEE Spectrum, Vol. 29 (October 92) No. 10, p. 35

[Den-95] Dennard, Davari, Shahidi, "CMOS Scaling for High Performance and Low Power - The Next Ten Years", Proceedings of the IEEE, April 1995

[Hu-94] C. Hu, "MOSFET Scaling in the Next Decade and Beyond", Semiconductor International, June 1994

[Kha-97] R. Friedrich, G. Kampe, H. Khakzar, A. Mayer, R. Oetinger, "Entwurf und Simulation von Halbleiterschaltungen mit PSPICE", Expert-Verlag 1997

[Shc-87] B. J. Sheu, D. L. Scharfetter, P. K. Ko, "BSIM: Berkeley Short-Channel IGFET Model for MOS Transistors", IEEE Journal of Solid-State Circuits, Vol. SC-22, No. 4, August 1987, pp. 558-566

[SIA-94] The National Technology Roadmap For Semiconductors, Semiconductor Industry Association, Santa Clara, 1994

[Sin-94] P. Singer, "New Interconnect Material: Chasing the Promise of faster Chips", Semiconductor International, November 1994

[TCA-95] "Technology Computer-Aided Design (TCAD) Roadmap: A Supplement to the National Technology Roadmap for Semiconductors", Technology Transfer 95012696A-TR, SEMATECH, 1995

[TCI-95] "Technology Computer-Aided Design (TCAD) Interconnect Modelling Supplement to the National Technology Roadmap for Semiconductors", Technology Transfer 95042800A-TR, SEMATECH, 1995

[DRi-98] Digital-Richtfunk (Digital Microwave), Short course at Technical Academy Esslingen, 1998

System On Chip (SOC)

Tendenzen beim Entwurf komplexer ICs mit IPs

Bericht von der IP 98 in Frankfurt

Vortrag auf dem XX. Workshop der MPC – Gruppe in Furtwangen

22. Januar 1998

Dirk.Jansen

ASIC - Design - Centre
Fachhochschule Offenburg
D 77652 Offenburg, Germany email: d.jansen@fh-offenburg.de

Mit abnehmender Strukturbreite können auf einem Siliziumchip heute weit mehr Funktionen integriert werden, als in einer vernünftigen Entwicklungszeit durch ein Designteam entworfen werden können. Dieser als "Productivity Gap" bezeichnete Unterschied fordert eine signifikante Steigerung der Designerproduktivität und verändert die bisher akzeptierten Optimierungs – Paradigmen vom flächenoptimierten Entwurf hin zu Verfahren, die den Entwicklungsaufwand minimisieren. Wesentliches Element hierbei ist die Wiederverwendbarkeit (Reuse) von Modulen , sogenannten IPs, die bereits als Handelsware auf dem Markt sind. Der Vortrag gibt eine Übersicht über die angebotenen Module, Standardisierungstendenzen und die mit der Verwendung einhergehenden Probleme.

1. Einführung

Mit abnehmender Strukturbreite können auf einem Siliziumchip heute weit mehr Funktionen integriert werden, als in einer vernünftigen Entwicklungszeit durch ein Designteam entworfen werden können. Die Produktivität eines Entwicklers oder auch eines Entwicklungsteams reicht nicht mehr aus, um in den immer kürzeren vom Markt geforderten Entwicklungszeiträumen Entwürfe von Grund auf neu aufzubauen Abb 1.

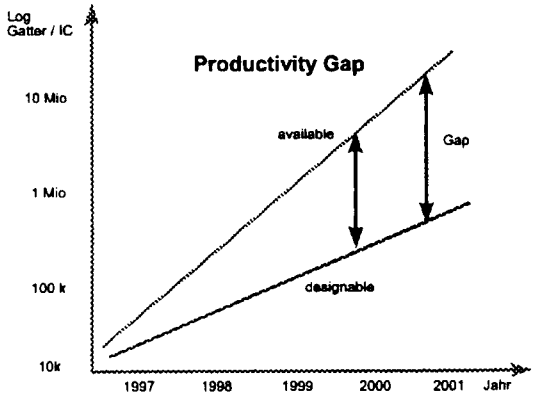


Abb 1: Der sogenannte Productivity Gap

Nach heutigem Stand enthalten ASICs durchschnittlich 150k Gatter oder 400 kbit Speicher, die Entwürfe werden in typisch 10 000 Zeilen Kode (VHDL) beschrieben. Schon heute könnten wirtschaftlich fünfmal so große und komplexe Chips entworfen werden, in Zukunft werden 100 000 Zeilen Kode, 500k Gatter und rund 1 Mbit Speicher typisch sein. Dies füllt aber immer noch nicht das aus, was technologisch möglich ist (Tabelle 1 und Abb 1).

Tabelle 1: Technologieentwicklung

Technologie	N Gatter	SRAM Mem/cm²
0.35	1 M	3.3 M
0.25	2 M	6.6 M
0.20	2.6 M	11 M
0.15	6.2 M	22 M

Mit der Verwendung programmierbarer Strukturen und Prozessorkernen, die als Hardmacros eingesetzt werden, wird auch heute schon auf bewährte Zellen zurückgegriffen. Hierbei handelt es sich um bereits geroutete Zellen, die in einer bestimmten Technologie vorliegen und validiert sind. Eine Übertragbarkeit in andere Technologien ist nicht ohne weiteres gegeben und erfordert erheblichen Entwicklungsaufwand. So zwingt der Übergang von einer 0.7 CMOS – Technologie mit 2 Metall – Lagen zu einer modernen 0.5µ oder 0.35µ - Technologie mit drei oder mehr Metallagen zu einer totalen Neuordnung der Netzliste, einer Neuplazierung und ein komplett neues Routing. Was bleibt über vom Design? Die Antwort lautet: **der eigentliche geistige Inhalt**

des Entwurfs, die Idee oder das, was heute als IP oder „Intellectual Property" bezeichnet wird.

Mit der Verbreitung von VHDL und anderen Entwurfssprachen ist es möglich, diese Idee in ausführbarer und weiterverarbeitbarer Form zu formulieren. Mit Hilfe von Simulatoren kann die Funktion dann nachvollzogen und gegen eine Spezifikation verifiziert werden. Syntheseprogramme erstellen daraus Netzlisten und erzeugen letztlich nach Plazieren und Routen funktionsfähige Zellen in der Zieltechnologie.

Die Beschreibungsform in Hochsprache ist damit technologieunabhängig, leicht modifizierbar und bezüglich des Entwicklungsaufwands ähnlich effektiv wie die Hochsprachen C oder PASCAL es im Bereich der Programmierung sind. IPs könnten damit die gleiche Rolle spielen wie Softwarebibliotheken, sie können insbesondere übertragbar von einem Designer zum andern sein und insbesondere Gegenstand von Handel und Lizensierung werden.

2. Was sind IPs?

Der Begriff IP kann am besten übersetzt werden mit „Urheberrechtssubjekt" und verallgemeinert eigentlich das, was schon lange unter den Begriffen „Softmacro" oder „Hardmacro" in der Designwelt verstanden wurde. Nach Lewis 1997 ist

„ein IP ein Entwurfsobjekt, dessen überwiegender Wert aus dem Wissen des Erzeugers hervorgegangen ist und dessen Neuentwurf eine bedeutende Zeit erfordern würde".

Das vom IP durch Synthese, Plazieren und Routen abgeleitete Silzium gewordene Objekt, z.B. die Zelle, der Kern oder die Komponente wird dann als Instanzisierung des IPs, als IP – Instanz (IPI) bezeichnet. Kleinere IPs, insbesondere Schnittstellenmodule, werden auch als virtuelle Komponenten (VC = Virtuell Components) bezeichnet, entsprechend den reelen Komponenten durch Datenblatt und definierte Eigenschaften beschrieben.

Man unterscheidet heute:

- **Hard IPs** werden auf Maskenebene beschrieben, z.B. durch GDSII – Daten. Typische Beispiele sind die Standardzellen der Hersteller, größere Prozessorkerne wie ARM, 186, 486, 320C25, analoge- und Mixed-Signal-Zellen mit erheblicher Abhängigkeit zur Technologie sowie die lange bekannten Generatoren für RAM, ROM, MPY usw. Auch ganze Transceiver und RF – Komponenten sind als Hard – IPs erhältlich.
- **Firm IPs** beschreiben den Modul auf Netzlistenebene mit festen Plazierungsvorgaben, jedoch ohne Details des Routings. Typisches Beispiel sind fast alle Kerne bei FPGAs. Firm IPs sind nur begrenzt auf andere Technologien abbildbar.
- **Soft IPs** beschreiben die Funktion des Moduls auf VHDL oder Verilog – Hochsprachenebene. Netzlisten werden erst durch Synthesetools erzeugt. Eine Abbildung auf die Zieltechnologie und Optimierung erfolgt erst in geeigneten Programmen. Typisch sind Interfacemodule, digitale Grundelemente wie Zähler, kleine Controller wie 8051, 6805, digitale Filter, in zunehmendem Maße auch komplexere Funktionseinheiten wie MPEG-Dekoder, Viterby–Decoder usw. Die IPs sind häufig parametrisierbar und damit flexibel an den Einsatzfall anpaßbar.

Im Einzelfall kann es fließende Übergänge geben. So sind bei Soft-IPs meist Mechanismen vorgesehen, die ein Verstehen und damit Offenlegen des eigentlichen Inhalts aus Gründen des Schutzes des IP – Providers verhindern oder sehr stark erschweren. Häufig sind auch aus Performancegründen große Teile der IPs in einem strukturellen VHDL – Stil abgefaßt, der einer Netzlistenbeschreibung schon sehr nahe kommt. Zusammen mit vorgegebenen Syntheseskripten, die detailiert die Abbildung (mappen) auf die Zieltechnologie steuern und Vorgaben für die Plazierung enthalten, müsste man diese Form eigentlich eher den Firm IPs zurechnen.

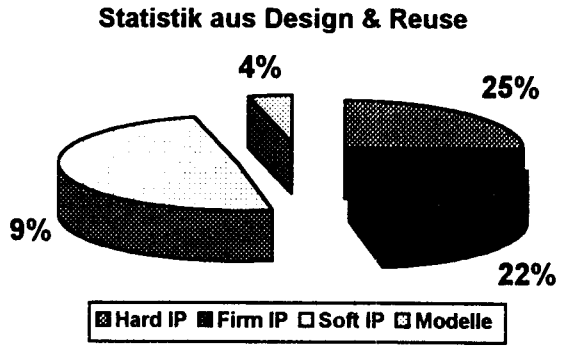


Abb 2: Statistik aus Design & Reuse

3. Standardisierung

Basis für jede Weiterverbreitung des IP – Ansatzes ist eine einigermaßen konsequente Standardisierung

- der Begriffe,
- der Beschreibungsformen,
- der Modelle,
- des Lieferumfangs,
- der rechtlichen Bedingungen,

- der EDA – Aspekte in Verbindung mit der Verwendung von IPs.

Zu diesem Zweck wurde von namhaften auf diesem Gebiet tätigen Firmen die

VSIA : Virtuel Socket Interface Alliance

http://www.vsia.com

gegründet, die sich zur Aufgabe gemacht hat, die zentrale Erstellung und Verteilung von Standards, Informationen und Angeboten zu IPs zu organisieren:

„... to specify or recommend a set of hardware and software interfaces, formats, and design practices for the creation of functional blocks that enable the efficient and accurate integration, verification, and testing of multiple blocks on a singel piece of silicon."

Nähere Einzelheiten dazu sind der Web-Site zu entnehmen, wo sich auch einige offen zugängliche Dokumente herunterladen lassen. Die detaillierteren Spezifikationen sind allerdings nur für Mitglieder zugänglich, ebenfalls die Mitarbeit in den Workinggroups. VSIA vertritt dabei sowohl die Interessen von IP–Providern bzw Herstellern wie auch von IP–Anwendern bzw Kunden und versucht, die Interessen beider Seiten auszugleichen. Bereits heute ist ein maßgeblicher Einfluß von VSIA im Bereich der Modellierungskonzepte, des Lieferumfangs (deliverables) und der EDA – Implementierungsverfahren zu spüren.

4. IPs als Handelsobjekte

IPs werden heute von allen wichtigen EDA – Firmen, die meist als Broker auftreten sowie den klassischen ASIC – Fabs angeboten. Hinzu kommt eine Reihe kleinerer Spin-Offs und Spezialanbieter, die ihr in verschiedenen Projekten erworbenes Zellen – Know – How zu vermarkten versuchen oder sogar speziell für den IP – Markt entwickeln. Typische Anbieter sind:

- Altera
- Xilinx
- Mentor Graphics (Inventra)
- Cadence
- Synopsis
- Dolphin Integration
- Phoenix Technologies
- Summit Design
- Frontier Design
- ISS
- Synchonicity
- ARM
- Argonaut RISC Cores

Eine ziemlich komplette Übersicht gibt die von TIMA/Grenoble gepflegte Web-Site:

http://www.design-reuse.fr

mit derzeit ca 1700 IPs, jeweils Kurzbeschreibung und Links zu den Providern. Der Zugriff auf die umfangreiche Datenbank ist derzeit noch kostenlos nach einfacher Online – Registrierung möglich. Die Datenbank enthält z.B. über 69 Controller und 88 Mikroprozessorkerne Abb 2 und Abb 3.

Bei der Vermarktung wird heute insbesondere das Geschäftsmodell der Lizensierung verfolgt, wobei hier ähnliche Bedingungen wie bei der Software vorliegen. Die Lizenz bezieht sich gewöhnlich nur auf die Verwendung des Kerns in einem Design, weitere Designs erfordern erneute, allerdings reduzierte Lizenzkosten.

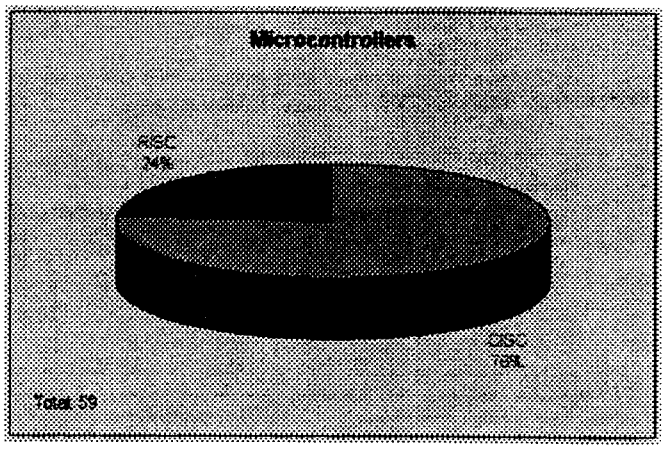


Abb 3: Statistik der in Design & Reuse enthaltenen Controller – IPs

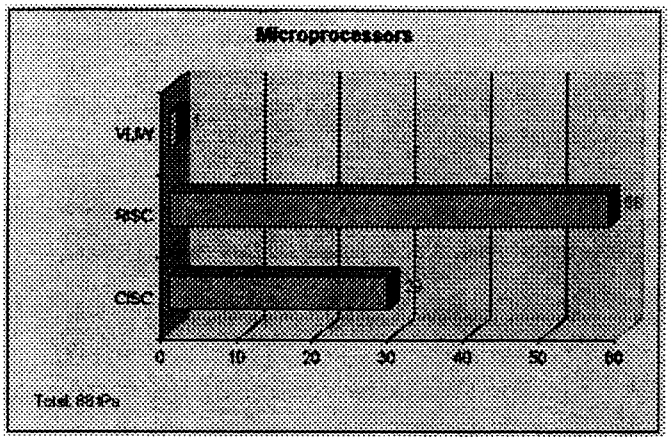


Abb 4: Statistik der in Design & Reuse enthaltenen Prozessorkerne, gegleidert nach Typ.

Die Preise sind noch signifikant hoch und können Tabelle 2 entnommen werden, allerdings mit fallender Tendenz durch die aufkommende Konkurrenzsituation. Generell bewegt der Preis als Daumenregel sich bei 15 ... 20% des Aufwands, den ein Neuentwurf der Zelle erfordert hätte, was immer noch sehr effektiv sein kann.

Tabelle 2: Arten und typische Lizenzkosten von IPs

IP – Typ	Typischer Preis in $ (1998)
FPGA- IPs (firm)	< 10 k
Interface Module (soft)	10 k ++
Controller (soft, hard)	20 k ++
Prozessoren ARC (soft)	120k
Prozessoren MIIC (soft)	650k
Prozessoren ARM 32b (hard)	1300 k
A/D-/Mixed Signal (hard)	40 k ++
RF /TM, Modem (hard)	80 k ++
JPEG, MPEG (soft)	500 k

Bei den erwähnten Summen darf man nicht vergessen, daß mit der Übergabe des Kerns gewöhnlich eine Woche Schulung, Anpassung des Kerns an die Forderung des Kunden sowie erhebliche Verfikationsunterstützung verbunden ist.

Der Lieferumfang umfaßt gewöhnlich mindestens:

Soft-IPs:

- VHDL – Kode,
- Synthese – Skript
- Stimuli für Simulation (Testmuster)
- Verhaltensbeschreibung für Signale
- Bus – Modell mit Timing
- Dokumentation

Hard – IPs:

- Geometriebeschreibung (GDS II)
- LVS – Files
- Design Rules
- Simulationsmodell (C, VHDL-Behave Style)
- Stimuli oder Workbench
- Testmuster
- Dokumentation

Dies sind nur Stichworte, im Detail können hierzu noch zahlreiche Unterlagen kommen.

5. Sondersituation FPGAs

Bei Field Programmable Gate Arrays (FPGA) liegt insofern eine Sondersituation vor, als die Hersteller wie ALTERA, XILINX oder ACTEL primär am Verkauf von Chips interessiert sind und selbst keine Design House Aktivitäten vorweisen. Die hier angebotenen Bibliotheken wie „Mega Core" sind deshalb preislich sehr interessant und nicht primäres Handelsobjekt. Die Anbieter übernehmen auch die Verteilung von Kunden – Kernen und Uni - Kerne (ohne Gewährleistung) und Brokerfunktionen und fördern allgemein die Anwendung von IPs.

Den Durchbruch dieser Bibliotheken brachte vor allem das PCI – Interface, welches dem Kunden relativ einfachen Zugang zum komplexen PCI – Standard ermöglicht und damit den Entwurf von Einsteckplatinen für PCs mit FPGAs sehr vereinfacht. An diesem Beispiel kann auch gut der Benefit für den Kunden, der sich vor allem in verkürzter Entwicklungszeit und verringertem Entwicklungsrisiko darstellt, nachgewiesen werden.

Die vielen Designern aus dem FPGA Bereich vertrauten parametrisierbaren IPs werden zunehmend auch für den klassischen ASIC – Bereich gefordert, insofern sind die FPGAs hier bahnbrechend.

Die FPGA – Kerne sind meist als Firm–IPs in verschlüsselter Form erhältlich und können auf den dedizierten Entwicklungsplätzen der Hersteller komplett auch vor Ankauf der Lizenz simuliert und in den Design integriert werden, sodaß nur geringes Risiko für den Kunden besteht. Die Lizenz ist erst zur Programmierung des Designs in den Zielbaustein erforderlich (Open Core–Programm von ALTERA).

Ein nicht unerheblicher Markt für große FPGAs mit mehreren 100k Gattern ist die Emulation von Softkernen, die zur

Erprobung auf FPGAs gemapped werden. Damit können signifikant Simulationszeiten eingespart und die Designsicherheit erhöht werden. Die Verwendung dieser großen FPGAs bleibt wegen der hohen Preise und großen Chipfläche ansonsten auf Sonderanwendungen mit geringen Stückzahlen beschränkt.

6. Probleme beim Design - Reuse

Bei der Wiederverwendung von Blöcken und Designs gibt es zahlreiche Probleme. So sind die meisten Module nicht von vorneherein für eine Wiederverwendbarkeit entworfen. Der zusätzliche Aufwand für diese Eigenschaft, sofern der Kern außer Haus als Handelsobjekt vertrieben werden soll, ist erheblich. Neben einer umfangreichen Dokumentation, einer gewissen Standardisierung der Schnittstellen und Signale sind auch genaue Prüfverfahren festzulegen, die eine Verfikation der Funktionen im Zielobjekt ermöglichen.

Für eine vorgegebene Lieferqualität steht eine Qualifikation des Designs gegen eine detaillierte Spezifikation, die beispielhafte Ausführung des Kerns in einer oder mehreren Zieltechnologien mit klaren Voraussagen bezüglich Performance (z.B. Systemtaktfrequenz) oder Flächenverbrauch. Ohne Validierung auf einer eingeführten Technologie ist heute kein Kern mehr zu verkaufen.

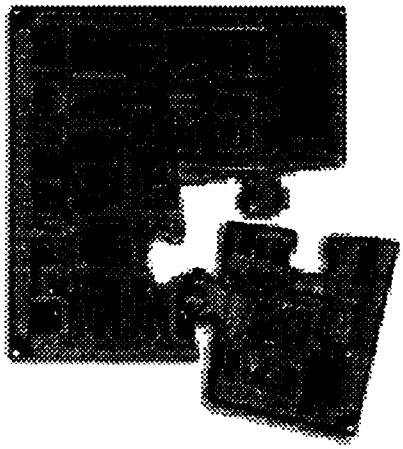


Abb 5: Entwurf Integrierter Schaltungen als Puzzle aus ineinanderpassenden IPs verstanden

Hard–IPs sind vergleichsweise anspruchsloser in Dokumentation und Verifikation, insbesondere wenn sie von diskreten Typen abgeleitet sind (z.B. 8051–Controller), bei Soft–IPs kommen zahlreiche Risiken bei Synthese und Plazierung hinzu. Die meisten Kunden haben bei Soft–IPs auch Sonderwünsche (Soft = flexibel den Wünschen anpaßbar), was aber im Gegensatz zu jeder Form der Qualifikation oder gar Funktionsgarantie steht. Die Geheimniskrämerei in Verbindung mit den Schutzmechanismen machen es dem Anwender zusätzlich schwer zu verstehen, warum manches geht und manches nicht. In der Praxis ist deshalb Vertrauen und Glauben gefragt, was im Prinzip unakzeptabel ist und zu erheblichem Betreuungsaufwand durch den Lieferanten führt. Letztlich gibt aber kein Provider auch nur irgend eine Form von Garantie für den gelieferten IP, hier muß der Kunde selbst das Restisiko tragen. Der Designtransfer ist belastet

- durch hohen Betreuungsaufwand des Providers und
- signifikante Einarbeitungszeit des Kunden.
- Restrisiko besteht durch unterschiedliche Interpretation der Dokumentation.
- Die Performance des Designs ist stark abhängig von der Synthese und den verwendeten Syntheseprogrammen sowie der Geschicklichkeit und Erfahrung des Designers.
- Automatische Skripts liefern sichere Ergebnisse, die aber nicht immer optimal sind in Hinsicht auf Fläche und Geschwindigkeit.
- Die Timingsimulation ist sehr zeitaufwendig und senkt die Effizienz des Reuse – Konzeptes.
- Durch nur teilweise bekanntes Innenleben der Kerne können unerwartete Effekte auftreten.

Typisch sind 1 ... 4 Wochen Einarbeitung. Bei einfachen Kernen oft nicht schneller als ein Eigenentwurf eines erfahrenen Designers. Hinzu kommen noch juristische Bestellballast, komplizierte Lizenzverhandlungen und die Tatsache, daß Entscheidungen im "Kilodollarbereich" geschäftpolitischen Charakter haben und gewöhnlich auf Geschäftleitungsebene mit entsprechender Zeitverzögerung entschieden werden.

7. Embedded Software

Ähnlich wie Kerne, die zu auf Silzium plazierbaren Zellen führen, kann auch eingebettete (embedded) Software als IP angesehen werden. Hiebei handelt es sich typischerweise um

- BIOS (Build In Operating Systems),
- Real Time Kernel,
- komplette Operatingsysteme,
- spezifische Applikationssoftware,
- Treiberprogramme.

Auch diese werden in zunehmendem Maße von Drittfirmen zugekauft und in die eigene Software integriert. Da mit der

Verbreitung von Prozessorkernen auf ICs auch ROM–residente Software immer mehr Bedeutung gewinnt, darf dieser Bereich nicht übersehen werden. So macht im Entwicklungsaufwand die Software mindestens 50% des Gesamtauf-

Abb 6: Software ist wesentlicher Bestandteil von SOC - Entwürfen

wands eines Projektes aus, bei SOC–Projekten also des eigentlichen Chip–Designs, sofern die Software nicht herunterladbar gestaltet wird.

Der Hardware-Software–Cosimulation muß deshalb besondere Aufmerksamkeit geschenkt werden. Große EDA–Hersteller bieten hierfür schon besondere Tools an, wobei die gleichzeitige Simulation der Chipfunktion auf einem Digitalsimulator und die Entwicklung der dabei eingesetzten Software große Anforderungen an den Rechner stellt. Die erreichbaren Taktzahlen sind heute für detaillierte Designmodelle völlig unbefriedigend. Das Vorhandensein schneller Verhaltensmodelle der Prozessoren ist deshalb von zentraler Bedeutung für diese Vorgehensweise.

Allgemein üblich ist heute der PC als Entwicklungsplattform, wobei über Cross–Compiler und Simulatoren die Anwendungsprogramme generiert und dann in die ROMs der Digitalsimulation eingespielt werden. Hochsprachen nehmen dabei immer größere Bedeutung ein, die Assemblerprogrammierung beschränkt sich zunehmend auf Treiberroutinen. Der ROM – Bedarf nimmt dabei gewaltig zu.

Abb 4: Hardware-Software Codesign muß vielfache Abhängigkeiten berücksichtigen

In Zukunft werden Treiberroutinen für Hardwarekomponenten wie z.B. FLASH – Bausteine als IPs zugekauft oder vom Hersteller der Komonenten beigestellt.

Sehr problematisch bei der Programmierung sind parametrisierbare Prozessoren. Änderungen im Instruktionssatz erfordern nicht nur eine entsprechende Änderung auch in den Entwicklungstools, sondern auch ein Umdenken des Programmierers. Allenfalls durch "retargetable" Compiler läßt sich diesem Problem begegnen, wobei die Detailstruktur des Kerns dem Entwickler auf C–Ebene verborgen bleibt. Grundsätzlich steht der Wunsch nach Flexibilität der Standardisierung entgegen. Der große Freiheitsgrad beim Entwurf von SOC steht dabei im Widerspruch zur Automatisierung, die "Erforschung des Designraums" ist heute noch Forschungsthema.

8. Zukünftige Entwicklungen bei SOC – Designs

Mikroprozessorkerne werden als Hard/ oder Soft–IPs zum Standard. Dabei werden sich diskret bekannte und lieferbare Typen weiter halten. Alle wichtigen Teilsysteme werden durch Virtuelle Komponenten VC, soweit die Funktionen blockmäßig definierbar sind, realisiert. Dabei spielen Bus-Standards, die auch innerhalb von ICs angewendet werden können, eine zunehmende Rolle. Es kann vorausgesagt werden, das der Glue Logic – Anteil, d.h. der Teil, der durch den Designer selbst erstellt wird, bei typischen SOCs unter 20% zurückgeht.

A/D – Komponenten und Zellen werden von spezifischen Lieferanten zugekauft, die das Mappen auf die gewünschte Zieltechnologie selbst durchführen.

EDA – Systeme werden so ausgelegt, daß sie die Ver-

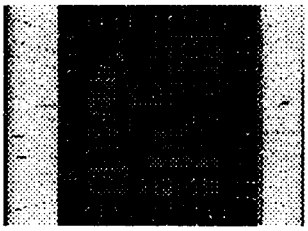


Abb 7: Typischer IC mit Hard und Soft – Ips (FHOP – PSK, Entwurf FH- Offenburg)

wendung von VC und IPs unterstützen..

Software wird in Form von BIOS, Treibern oder kompletten OS ebenfalls zugekauft.

Da die Chipfläche groß genug ist, erfolgt keine Optimierung mehr auf die Fläche des ICs, sondern auf

Time to Market !!!

Wegen der hohen Lizenzkosten können derzeit nur große Unternehmen so arbeiten (Big Business!), allerdings wird ähnlich wie im Software – Bereich durch den freien Anbietermarkt der Preis für IPs bald sinken, Unis und FHs drücken die Preise zusätzlich durch Shareware- oder Freeware–IPs.

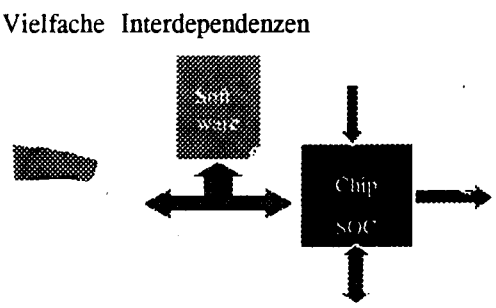


Abb 8: Vielfache Abhängigkeiten beim Hardware – Software Codesign

Bei zukünftigen Entwicklungen sind alle Designer aufgefordert, über die Weiterverwertung eigener Funktionsblöcke im Sinnen von IPs nachzudenken und zu prüfen, wieweit nicht auf auf dem Markt verfügbare IPs zurückgegriffen werden kann. Die erforderlichen EDA – Tools sind verfügbar. Nur so kann die notwendige Steigerung der Entwicklereffizienz erreicht werden und der "Productivity Gap" überwunden werden.

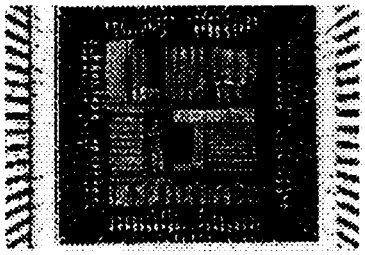


Abb 9: Integrierte Schaltung mit Hard- und Soft – IPs (Entwurf FH – Offenburg)

Zusammenfassung

Mit abnehmender Strukturbreite können auf einem Siliziumchip heute weit mehr Funktionen integriert werden, als in einer vernünftigen Entwicklungszeit durch ein Designteam entworfen werden können. Dieser als "Productivity Gap" bezeichnete Unterschied fordert eine signifikante Steigerung der Designerproduktivität und verändert die bisher akzeptierten Optimierungs – Paradigmen vom flächenoptimierten Entwurf hin zu Verfahren, die den Entwicklungsaufwand minimisieren. Wesentliches Element hierbei ist die Wiederverwendbarkeit (Reuse) von Modulen , sogenannten IPs, die bereits als Handelsware auf dem Markt sind. Der Vortrag gibt eine Übersicht über die angebotenen Module, Standardisierungstendenzen und die mit der Verwendung einhergehenden Probleme.

Literatur

[1] Proceedings IP 98, The Intellectual Propperty in Electronics, Conference and Exhibition, Frankfurt 1998

[2] Ben-Romdane, M.: Lego-Like Integration of IP – Cores, IP 98.

[3] Schlichtmann, Ulff et.alt: Implementing a Corporate Design Reuse Strategy, IP 98

[4] Martin, Grant et.alt.: The Integration Platform Approach to System On Chip Design, IP 98

[5] Saucier, G. et. alt.: Hard IP Migration. IP 98.

[6] Mentor Graphics: Inventra IP Catalog. Firemnschrift Mentor Graphics.

[7] ALTERA: Megacore Functions for High-Density Designs, Firmenschrift Altera.

[8] Dolphin Integration: Mixed signal Virtual Components. Firmenschrift.

Anwendungsspezifischer Baustein zur Anzeige von Frequenz und Zeit

W.Ludescher
FH
Ravensburg-Weingarten

22. Januar 1999

Zusammenfassung

Der anwendungsspezifische Schaltkreis ZAE3 wurde im Rahmen der CAE-Vorlesung im Wintersemester 97/98 vom 5. Semester Elektronik an der Fachhochschule Ravensburg-Weingarten entworfen. Entwicklungsziel war ein Baustein zur Anzeige der Uhrzeit sowie der Sende- oder Empfangsfrequenz eines Sende-Empfängers im Frequenzbereich ab Gleichspannung (DC) bis 200MHz. Der Baustein stellt sein Meßergebnis mit 10 Dezimalstellen dar, führende Nullen werden unterdrückt. Als Zeitnormal kommt ein 1MHz-Signal zum Einsatz. Messungen an Prototypen ergaben Bandbreiten von DC bis 300MHz.

Inhaltsverzeichnis

1 Allgemeines

Der Schaltkreis ZAE3 dient folgenden Zwecken:

- Darstellung der Uhrzeit auf einer 7-Segment-Anzeige. Es werden Stunden, Minuten und Sekunden ausgegeben.
- Überwachung und Darstellung der Sendefrequenz eines Kurzwellen- oder Ultrakurzwellensenders.
- Anzeige der Empfangsfrequenz eines Kurzwellen- oder Ultrakurzwellen-Empfängers durch Überwachung des lokalen Oszillators.
- Einsatz als Meßgerät, Messung von Frequenz, Periode, Ereignis und Zeit.

Der Schaltkreis enthält folgende Funktionsblöcke:

- Eine Zeitbasis mit sieben dekadischen Zählerstufen zur Ableitung von Auflösungen oder Meßzeiten zwischen einer Mikrosekunde und 100 Sekunden. Die Zeitbasis wird mit einem 1MHZ–Signal angesteuert.
- Einen Vorteiler mit vier dekadischen Teilerstufen zur Messung einer Multiperiodendauer.
- Zehn dekadische Zählerstufen, Ausgabe von zehn Stellen an eine Sieben-Segment-Anzeige. Die Ausgabe führender Nullen der Zählerstufen wird von der Anzeige unterdrückt.
- Zu dem Zählerstand der Zählerstufen kann ein zehnstelliger Zahlenwert addiert werden. Diese Funktion erlaubt die Anzeige der Empfangsfrequenz eines Überlagerungs–Empfängers.
- Der Baustein kann in drei verschiedenen Betriebsarten – Minimal, Boot-Mode und PC-Mode – betrieben werden. Die Betriebsarten unterscheiden sich durch Funktionalität und Schaltungsaufwand. Eine funktionsfähige Zählerschaltung besteht – je nach Betriebsart – aus einem, zwei oder drei ICs.

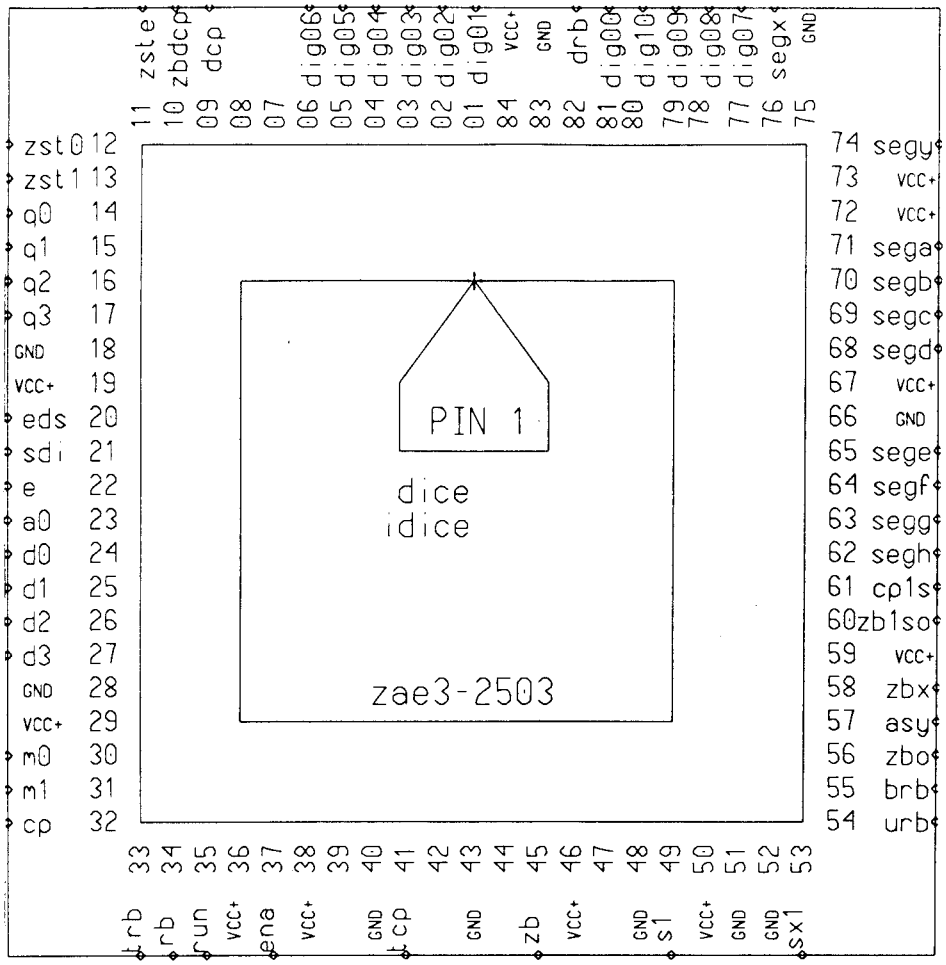


Abbildung 1: dice

2 Zur Vorgeschichte

2.1 Schaltungsentwicklung

Der Schaltungsentwurf wurde im Verlauf des Wintersemesters 1997/98 im CAE-Labor der Fachhochschule ausgeführt und im März 1998 in die Fertigungslinie des Instituts für Mikroelektronik Stuttgart eingeschleust. Der Baustein besitzt eine Komplexität von ca. 43000 Transistoren und ist auf einem 0.8um-Gate-Array realisiert. Das Gate-Array ist hoch ausgelastet.
Als Entwurfswerkzeuge kamen der Design-Architekt, VHDL, Quicksim und Quickfault in Mentor V8 zum Einsatz. Archiviert entstand eine gepackte tar-Datei von knapp 5 MegaByte. Der Test des Bausteins benötigt etwa 200k Bitmuster pro Pin, das entspricht einer ASCII-Datei von knapp 200 MegaByte. Quickfault nimmt sich für eine vollständige Fehlersimulation auf einer J200 etwa 120 Minuten Zeit und erreicht dann einen Fehlerabdeckungsgrad von 99%.

2.2 Chipfertigung und Test

Der Schaltungsentwurf durchlief die Chipfertigung am IMS in Rekordzeit. Bereits vor der Sommerpause stand Silizium zur Verifikation zur Verfügung. Der Baustein zeigte sich funktionsfähig - an Prototypen wurden Grenzfrequenzen bis 300MHz gemessen.

Die Bausteine sind in einem 84-Pin-Gehäuse aufgebaut, wobei die schnellen, hochfrequenten Eingangssignale an der Südseite, die relativ langsamen Signale der Anzeige an der Ost- und Nord-Seite und die Schnittstelle zu einem externen

Mikroprozessor auf der West-Seite angeordnet sind. Masse- und Versorgungsspannung sind besonders deutlich an der Südseite des Bausteins definiert, da hier die schnellen Eingangssignale liegen.

3 Anwendungsbeispiele

3.1 Frequenzmessung

Zur Frequenzmessung sind folgende Einstellungen ohne besonderen Aufwand möglich:

- Meßzeit (Torzeit) 100ms, 1Sekunde, 10ms
- Aktivieren dekadischer Vorteilerstufen
- Aktivieren binärer Vorteilerstufen
- Unterdrücken führender Nullen der Anzeige
- Stellen bzw. Ausgabe der Uhrzeit

3.2 Messung der Periodendauer

Zur Messung der Periodendauer sind folgende Einstellungen ohne besonderen Aufwand möglich:

- Auflösung auf Mikrosekunden, Millisekunden bzw. Sekunden.
- Aktivieren dekadischer oder binärer Vorteilerstufen, Multiperiodendauer
- Unterdrücken führender Nullen der Anzeige
- Stellen bzw. Ausgabe der Uhrzeit

3.3 Überwachung einer Empfangsfrequenz

In dieser Betriebsart wird der gemessenen Eingangsfrequenz ein Zahlenwert dazuaddiert. Schwingt z.B. der lokale Oszillator um die Zwischenfrequenz von 10.7 MHz über bzw. unter der Eingangsfrequenz, so wird bei jeder Frequenzmessung ein passender Wert zum gemessenen Signal addiert. Die Anzeige gibt dann die korrekte Empfangsfrequenz aus. Der Offset ist frei wählbar und muß daher aus einem externen Speicher geladen werden. Dies geschieht entweder durch die auf dem Baustein vorhandene Prozessor- Schnittstelle oder mittels einer eingebauten Urlade-Funktion.

3.4 Anzeige der Uhrzeit

Die Uhrzeit wird auf 8 der 10 Anzeigestellen ausgegeben. Die beiden höchstwertigsten Anzeigestellen sind dunkelgetastet. Stunden, Minuten und Sekunden sind durch Bindestriche getrennt. Die Uhr wird mit drei Tasten (Stellen, Stunden, Minuten) gestellt. Man drückt dazu einige Sekunden die Stellen-Taste, was die Sekunden-Zähler auf Null setzt. Drückt man dann zusätzlich die Stunden- oder Minuten-Taste, so werden Stunden- oder Minutenstand im Sekundentakt erhöht.

3.5 Wahl der Betriebsart

Um je nach Anwendungsfall den externen Beschaltungsaufwand zu minimieren sind drei Betriebsarten vorgesehen. Die einfachste Betriebsart - genannt `Mini-Mode` - benötigt (außer der externen Anzeige) keine weiteren Bausteine, allerdings ist die Funktionsauswahl auf typische Messungen eingeschränkt. Die Funktionsauswahl wird durch 16 Schalter oder Taster definert. Das Einlesen der Schalter erfolgt im Zeit-Multiplex, einzelne Schalter oder Taster sind durch Dioden oder einen Analogmultiplexer entkoppelt. Ist z.B. die niederstwertigste Stelle (dig00) aktiv, so können zwei Schalter S0a und S0b eingelesen werden. Auf diese Weise werden zum Einlesen von 16 Schalterstellungen nur zwei zusätzliche Eingangspins (zst0, zst1) am Baustein nötig. Das Einlesen funktioniert nur, wenn zste logisch 1 ist - im anderen Fall ist der vorhandene Wert 'eingefroren'.

Im `PC-Mode` wird der Schaltkreis über eine Prozessor-Schnittstelle konfiguriert. Die dazu notwendigen acht Eingangspins befinden sich auf der westlichen Seite des Bausteins. Der interne Zustand des ZAE3 kann über vier Ausgangspins beobachtet werden. Die Funktionsauswahl des Mini-Modes ist jetzt logisch abgetrennt, jedoch können die 16 Schalter oder Taster weiterhin durch den Prozessor beobachtet werden. Die Mini-Funktionsauswahl fungiert jetzt als Port-Erweiterung. In der Betriebsart PC-Mode ist die volle Funktionalität zugänglich - allerdings bedeutet dies auch, daß extern ein Mikroprozessor und ein geeignetes Programm bereitstehen muß. Eine gedruckte Schaltung wird den Prozessor, ein EPROM, ggf. ein RAM und den ASIC enthalten müssen.

Als dritte Betriebsart ist der `BOOT-Mode` möglich. Der Schaltkreis konfiguriert sich jetzt selbstständig und nutzt dazu Daten, die in einem externen EPROM bereitstehen. Eine Urlade-Logik addressiert dieses EPROM und liest dessen Daten in den ASIC. Somit steht die volle Funktionalität zur Verfügung, allerdings muß der Datensatz kürzer als 256 Byte und im EPROM abgelegt sein. Eine Beeinflussung des Bausteins über die Mini-Funktionsauswahl ist nicht möglich. Im Boot-Mode wird durch die Wahl einer Basis-Adresse am EPROM aus verschiedenen Speicherbereichen des EPROMs geladen, was die Funktion des ASICs definiert. Zum Betrieb des ASICs sind damit zwei ICs notwendig.

4 Zusammenfassung

Der anwendungsspezifische Schaltkreis ZAE3 wurde im Rahmen der CAE-Vorlesung vom 5. Semester Elektronik im WS97/98 entworfen und besteht aus den Baugruppen 'Vorteiler' , 'Zeitbasis' , 'Torsteuerung' , 'Zähler' , 'Summierer' , 'Ausgabe-Einheit' und 'Uhr' . Die Ausgabe-Einheit kann auf Uhrzeit oder Zählerstand umgeschaltet werden. Der Schaltkreis kann über seine Prozessor-Schnittstelle durch einen externen Mikroprozessor gesteuert werden. Ein Betrieb ohne Prozessor ist durch eine einfache Bedienschnittstelle ebenfalls möglich.

Typische Anwendungen, Funktionen, Eigenschaften und Technologiekenndaten sind:

- Bandbreite (Ziel) von DC bis ca. 200MHz
- Messung der Frequenz, Periode oder Multiperiode
- Zeitbasis 1MHz, Auflösung 1Hz bei Meßzeit 1 Sekunde
- Binäre oder dekadische Vorteiler zuschaltbar
- Anzeige über 10 Dekaden, Unterdrückung führender Nullen
- Addition einer Zwischenfrequenz
- CAE-Software, Mentor V8, VHDL
- Chipfertigung am IMS, Stuttgart
- Komplexität ca. 43000 Transistoren
- Technologie 0.8um CMOS Gate-Array

Modulo-Arithmetik für große Zahlen

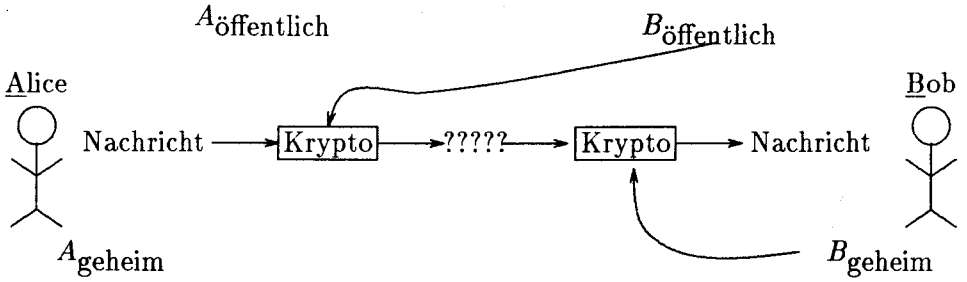
Wolfgang Rülling
Mikrosystemtechnik, Fachhochschule Furtwangen

Beim RSA-Verschlüsselungsverfahren wird als Grundoperation die Berechnung von $a \cdot b \bmod m$ für sehr große Zahlen benötigt. Besonders effizient geht dies, indem man die Multiplikation und die Modulo-Rechnung gemeinsam durchführt, so daß keine unnötig großen Zwischenergebnisse auftreten. In der Arbeit werden dazu verschiedene Realisierungsmöglichkeiten in Hardware vorgestellt. Das Ziel ist ein Chip, der die Aufgabe mit linearem Flächenaufwand und linearem Zeitaufwand in der Datenlänge n löst.

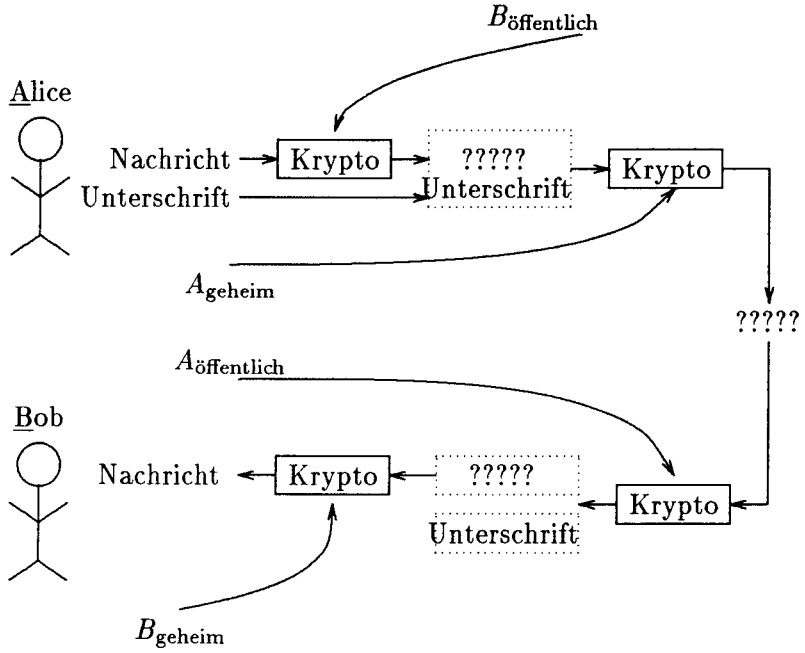
1 Einführung in die Kryptologie

Verschlüsselungsverfahren gewinnen zunehmend an Bedeutung. Neben dem naheliegenden Nutzen, daß ein unbefugtes Lesen von Nachrichten verhindert wird, werden sie auch benutzt, um sicherzustellen, daß Nachrichten nicht manipuliert werden können und daß der Absender der Nachricht zweifelsfrei identifiziert werden kann. ([2, 12])
Als Anwendungsbeispiele denke man das Bestellen von Waren über das Internet, Homebanking, das Aushandeln und Abschließen von Verträgen im Internet, oder das bargeldlose Bezahlen mit einer Cash-Karte. In diesen Fällen braucht man eine "elektronische Unterschrift" (Signatur), die auch juristisch verbindlich ist.
Um eine hohe Sicherheit gegen Fälschungen zu erreichen, müssen die zu sichernden Dokumente oder Daten mit der Signatur eine Einheit bilden, so daß zu einem geänderten Dokument auch eine neue Signatur gehört. Auf diese Weise kann jede Signatur nur genau einmal verwendet werden.
Bei den Verschlüsselungsverfahren unterscheidet man symmetrische und asymmetrische Verfahren. Symmetrischen Verfahren verwenden zum Ver- und Entschlüsseln jeweils den gleichen Schlüssel. Das bekannteste Beispiel eines symmetrischen Verfahrens ist der *DES-Algorithmus* (DES=Data Encryption Standard) (siehe z.B.[12]). Dieses Verfahren ist recht einfach in Hardware realisierbar, da nur elementare Operationen, wie Addition, Exor-Verknüpfung und Shiften benötigt werden ([10]). Ein Nachteil symmetrischer Verfahren besteht darin, daß der Schlüssel sowohl dem Sender als auch dem Empfänger bekannt sein muß. Deshalb tritt

das Problem auf, wie man diesen Schlüssel bei der erstmaligen Kontaktaufnahme auf sichere Weise übertragen kann.
Bei asymmetrischen Verfahren entfällt dieses Problem, weil es jeweils einen öffentlichen und einen geheimen Schlüssel gibt. Diese beiden Schlüssel sind in dem Sinne invers zueinander, daß man die mit dem einen Schlüssel codierten Nachrichten mit dem anderen wieder decodieren kann.

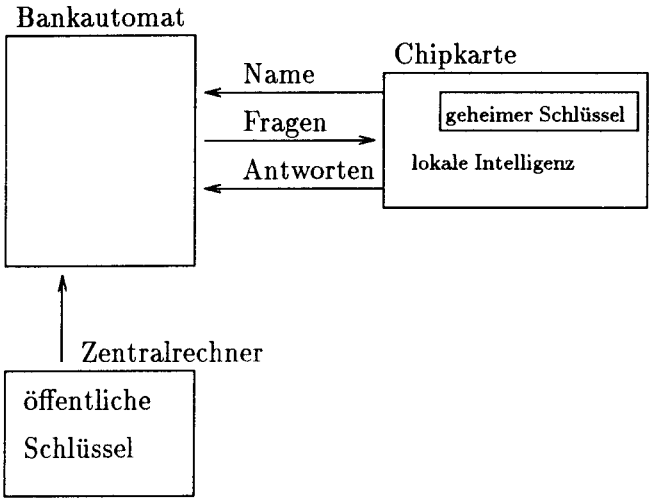


Will beispielsweise eine Person A einer Person B eine Nachricht schicken, benutzt der Sender A den öffentlichen Schlüssel $B_{\text{öffentlich}}$ von B, um damit die Nachricht zu verschlüsseln. Jetzt ist B die einzige Person, die die Nachricht entschlüsseln kann, weil nur B über den dazugehörigen geheimen Schlüssel B_{geheim} verfügt. Soll B auch sicher sein können, daß die empfangenen Nachricht tatsächlich von A stammt, dann kann A die Nachricht mit dem Absender "A" unterschreiben und anschließend mit dem eigenen geheimen Schlüssel A_{geheim} codieren. Dies entspricht einer Signatur.



Wenn B eine derartige Nachricht erhält, decodiert er sie zunächst mit dem öffentlichen Schlüssel $A_{\text{öffentlich}}$ von A und kann daraufhin die Unterschrift "A" lesen. Wenn dies gelingt, kann B sicher sein, daß die Nachricht tatsächlich von

A stammt, weil nur A über den geheimen Schlüssel A_{geheim} zum Erzeugen des codierten Textes verfügt. Anschließend decodiert B die eigentliche Nachricht mit B_{geheim}.
Als bekanntestes Beispiel für ein asymmetrisches Verfahren wird in dieser Arbeit der *RSA-Algorithmus* (RSA=Rivest, Shamir, Adleman) behandelt ([7]). Dieser Algorithmus verwendet sehr große Zahlen und ist deshalb relativ schwierig in Hardware zu realisieren ([9]).
Bei Internetanwendungen werden die Verschlüsselungsverfahren typischerweise durch Software realisiert. Dabei kann der Komfort soweit gehen, daß der Anwender die Verschlüsselungen überhaupt nicht bemerkt, sondern die Kommunikationssoftware automatisch alle ankommenden Nachrichten mit dem eingenen geheimen Schlüssel decodiert und Signaturen mit Hilfe des im Netz verfügbaren öffentlichen Schlüssels des Absenders überprüft.
Dabei steht und fällt die Sicherheit allerdings mit der Verwaltung der geheimen Schlüssel. Sie dürfen auf keinen Fall für Unbefugte erreichbar sein. Deshalb ist für die Zukunft zu erwarten, daß man *Spezialhardware* zur Sicherung von Schlüsseln einsetzen wird. Beispielsweise könnte sich der Benutzer eines PCs durch eine *Chipkarte* gegenüber dem PC ausweisen müssen, bevor er die Verschlüsselungssoftware einsetzen kann. Oder der Zugriff auf das eigene Konto beim Homebanking wird nur gestattet, wenn die entsprechende EC-Karte ebenfalls in den Rechner eingesteckt wurde.



Dabei dienen die Chipkarten als Safe für Geheimschlüssel. Durch "lokale Intelligenz" auf der Karte können Verschlüsselungen (Signaturen) direkt auf der Karte erzeugt werden, ohne daß der gespeicherte Schlüssel transferiert werden muß.
Beispielsweise könnte ein Dialog zwischen einem Bankautomaten und einer Chipkarte so ablaufen, daß die Karte sich zunächst durch ihre Nummer identifiziert, der Automat sich daraufhin den öffentlichen Schlüssel des Kartenbesitzers vom Zentralrechner der Bank besorgt und die Karte dann auffordert, zufällige vom

Bankautomaten gelieferte Daten mit dem geheimen Schlüssel zu codierten. Anschließend entschlüsselt der Automat die Daten mit dem öffentlichen Schlüssel und überzeugt sich auf diese Weise davon, daß die Karte den ursprünglich von der Bank vergebenen geheimen Schlüssel tatsächlich kennt.
Bei diesem Vorgang ist wesentlich, daß der geheime Schlüssel selbst nicht übertragen wird. Hackern sollte es so (praktisch) unmöglich sein, sich den geheimen Schlüssel zu verschaffen. Man beachte, daß dieser geheime Schlüssel selbst der ausgebenden Bank nicht bekannt sein muß, weil die Prüfungen nur mit dem öffentlichen Schlüssel erfolgen.
Die skizzierten Beispiele zeigen, daß es oftmals von Interesse ist, Verschlüsselungsverfahren "abhörsicher" in Spezialhardware zu realisieren. Deshalb soll im folgenden untersucht werden, wie das RSA-Verschlüsselungsverfahren effizient in Hardware realisiert werden kann.

2 Das RSA-Verfahren

Das RSA-Verfahren ist ein asymmetrisches Verschlüsselungsverfahren. Es wird durch zwei Schlüssel e und d, sowie einem zusätzlichen öffentlichen Parameter m definiert. Das Verschlüsseln einer Nachricht x geschieht gemäß der Formel

$$y = x^d \bmod m$$

und das Entschlüsseln geht genauso, indem man einfach den Schlüssel d durch e ersetzt.

$$x = y^e \bmod m$$

In der Praxis sind alle beteiligten Werte sehr große Zahlen. Beispielsweise verwendet man 1000-stellige Binärzahlen.
Die Sicherheit des RSA-Verschlüsselungsverfahrens beruht auf dem Umstand, daß es sehr schwierig ist, große Zahlen in ihre Primfaktoren zu zerlegen. Das Verschlüsselungsverfahren zu knacken bedeutet, aus dem öffentlichen Schlüssel den geheimen Schlüssel zu berechnen. Wann immer dies gelingt, hat man tatsächlich eine aus m ableitbare sehr große Zahl faktorisiert. Im folgenden soll das Verfahren an einem sehr kleinen Beispiel erläutert werden.
Wir benutzen $m = 143$. Dabei gilt $m = 11 \cdot 13$, d.h. m ist das Produkt zweier Primzahlen. Diese Primzahlen werden nur zur Konstruktion der Schlüssel benötigt und sollten danach sofort wieder gelöscht werden. Denn wenn es jemanden gelingt, den öffentlichen Wert m in seine Primfaktoren zu zerlegen, kann er leicht aus dem öffentlichen Schlüssel den geheimen Schlüssel rekonstruieren, der Code wäre also geknackt.
Als öffentlichen Schlüssel benutzen wir (relativ willkürlich) $d = 77$ (diese Zahl muß zu $p - 1$ und $q - 1$ teilerfremd sein). Danach ermitteln wir mit Hilfe des erweiterten Euklidischen Algorithmus einen eindeutig bestimmten Wert e mit

der Eigenschaft $(e \cdot d) \bmod ((p-1)(q-1)) = 1$. Im vorliegenden Beispiel ist dies der Wert $e = 53$, denn $53 \cdot 77 \bmod (10 \cdot 12) = 4081 \bmod 120 = 1$.
Der öffentliche Schüssel lautet also 77 und der geheime Schlüssel ist 53. Da wir modulo 143 rechnen, können wir mit dem Verfahren Zahlen < 143 verschlüsseln. Soll etwa der Wert $x = 50$ codiert werden, so ermitteln wir

$$y = 50^{77} \bmod 143 = 85$$

Die codierte Nachricht lautet also 85. Der Empfänger wird die Nachricht mit seinem geheimen Schlüssel folgendermaßen rekonstruieren:

$$x = 85^{53} \bmod 143 = 50$$

Benutzen wir in der Praxis Schlüssel der Länge 1000 Bit, so zerlegen wir die zu sendende Nachricht jeweils in Blöcke der Größe 1000 Bit, die jeweils einzeln nach obiger Methode codiert werden.
Man beachte, daß man bereits bei dem extrem kleinen Zahlenbeispiel bei naiver Berechnungsmethode zu sehr langen Zwischenergebnissen kommt. So ergibt beispielsweise 85^{53} eine Zahl mit 103 Dezimalstellen (85^{53} = 18163647410256147230426087525449035836107947163994613599421271070366579181154520483687520027160644531 25). Deshalb ist es unbedingt erforderlich, die Modulo-Rechnung so frühzeitig wie nur irgend möglich zu berücksichtigen, um lange Daten zu vermeiden.

2.1 Die Square-and-Multiply Methode

Eine einfache Methode für das Potenzieren besteht darin, den Exponenten bitweise abzuarbeiten. Beispielsweise gilt $53_{\text{dezimal}} = 110101_{\text{binär}}$, so daß a^{53} auch als $a^{32} \cdot a^{16} \cdot a^4 \cdot a^1$ geschrieben werden kann. Die Square-and-Multiply Methode besteht dann darin, den Wert a fortgesetzt zu quadrieren und die jeweils benötigten Potenzen zum Ergebnis y aufzumultiplizieren. In der C-ähnlichen Programmiersprache "bc", die mit beliebig großen Zahlenwerten arbeiten kann, sieht der Algorithmus "pot" folgendermaßen aus:

```
/* Berechnung von (a^b) fuer
   n-stellige Binaerzahlen */
define pot(a,b[]){
 auto i,y,x;
 y=1; x=a;
 for(i=0; i<n; i++) {
  if (b[i]==1) y *= x;
  x *= x;
 }
 return(y);
}
```

```
/* Berechnung von (a^b) mod m
   fuer n-stellige Binaerzahlen */
define rsa(a,b[],m){
 auto i,y,x;
 y=1; x=a;
 for(i=0; i<n; i++) {
  if (b[i]==1) y = (y*x)%m;
  x = (x*x)%m;
 }
 return(y);
}
```

Interessieren wir uns wie bei der RSA-Verschlüsselung nur für die Ergebnisse modulo m, so kann man wie in der Funktion "rsa" dargestellt, sämtliche Multiplikationen modulo m durchführen und so unnötig große Zwischenergebnisse vermeiden. Deshalb ist $(a \cdot b) \bmod m$ die wichtigste bei einer RSA-Implementierung durchzuführende Operation.

3 Multiplikation modulo m

Um geeignete Algorithmen für die Hardwarerealisierung der Grundoperationen auszuwählen, müssen zunächst die einzuhaltenden Randbedingungen festgelegt werden. Soll ein Rechenwerk beispielsweise unabhängig von der Wortlänge n mit einem konstanten Platzbedarf auskommen, ist nur eine sequentielle Abarbeitung wie bei einer Softwarelösung möglich. In diesem Fall bieten sich spezielle Zahldarstellungen, wie etwa die RNS-Darstellung [4] oder eine Fourier-Transformation zur Beschleunigung der Softwarelösung an.
Soll die Hardwarerealisierung aber drastisch schneller sein, als die Software, so muß man die auf dem Chip mögliche Parallelverarbeitung ausnutzen, also eine mit n wachsende Chipfläche investieren. Für die vorliegende Arbeit soll deshalb die Nebenbedingung gemacht werden, daß die zugrundeliegende Operation $a \cdot b \bmod m$ mit linearem Flächenaufwand ($A = \mathrm{O}(n)$) realisiert werden soll.
Unter dieser Nebenbedingung kann die Multiplikation zweier Binärzahlen bestenfalls in Zeit $T = \mathrm{O}(\sqrt{n})$ durchgeführt werden, denn aufgrund der Kommunikationskomplexität gilt für die Binärmultiplikation die Abschätzung $AT^2 = \Omega(n^2)$. Tatsächlich findet man in der Literatur Implementierungen, die eine lineare Laufzeit erreichen. Beispielsweise wird dies in [1] durch die Kombination einer "Montgomery-Multiplikation" mit einer aufwendigen RNS-Zahldarstellung (RNS=Residue Number System) erreicht und in [8] wird die Verwendung eines systolischen Feldes für die Montgomery-Multiplikation vorgeschlagen. Beim ersten Verfahren fällt auf, daß der Vorteil der RNS-Zahldarstellung durch die erforderlichen Datenkonvertierungen wieder zunichte gemacht wird, weil sich RNS-Zahlen nicht gut für die Modulo-Rechnung eignen. Bei der zweiten Arbeit fällt auf, daß das systolische Feld nur relativ langsam mit Daten gefüllt wird, so daß von den vorhandenen Rechenwerken im Durchschnitt nur die Hälfte wirklich arbeiten.
Deshalb soll versucht werden das gleiche Ziel mit einer erheblich einfacheren Schaltungsstruktur zu erreichen. Dabei soll darauf geachtet werden, daß sämtliche Signale eine von n unabhängige kurze Laufzeit haben, so daß sich der Chip auch bei großen Datenlängen mit extrem hohen Taktraten betreiben läßt.

Entwurfsziel: Realisierung von $a \cdot b \bmod m$
mit Zeitbedarf $T = \mathrm{O}(n)$ und Flächenbedarf $A = \mathrm{O}(n)$.

3.1 Multiplikation mit dem Kehrwert

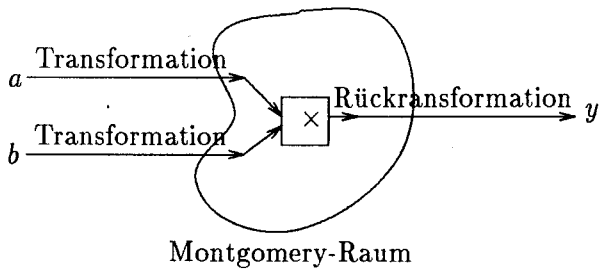
Eine erste Realisierungsidee für die Modulo-Rechnung könnte darin bestehen, eine ganzzahlige Division durch m vorzunehmen und den dabei entstehenden Rest zu betrachten. Damit wäre die Modulorechnung auf eine Division zurückgeführt. Tatsächlich ist dieser Aufwand im vorliegenden Fall nicht erforderlich, weil wir sämtliche Modulo-Rechnungen mit immer dem gleichen Argument m vornehmen und deshalb eine Vorverarbeitung von m möglich ist.
Beispielsweise könnten wir die inverse Größe $\frac{1}{m}$ vorberechnen, so daß sich die Division auf die Multiplikation mit diesem Kehrwert zurückführen läßt. In [6] wird gezeigt, wie man die Aufgabe $(a \cdot b) \bmod m$ mit dieser Technik lösen kann. Dazu bezeichnen wir mit m' den auf $3n$ Stellen genau berechneten Kehrwert von m (in der $3n$-ten Nachkommastelle ist der Wert aufgerundet!).
Dann können wir folgendermaßen vorgehen:

Beispiel:	$a = 124, b = 374, m = 532$
	$m' = 0.001879700$
Methode:	
1. Berechne $a * b * m'$	87.172967200
2. Entferne die Vorkommastellen	0.172967200
3. Multipliziere mit m	92.018550400
4. Runde auf ganze Zahl	92
	Probe: $124 * 374 \bmod 532 = 46374 \bmod 532 = 92$

Auf diese Weise werden, von einfachen Rundungen und dem Wegstreichen von Stellen abgesehen, nur drei Multiplikationen benötigt. Allerdings haben die beteiligten Zahlen nicht nur n Ziffern, sondern bis zu $3n$ Ziffern. Damit ergibt sich bei genauer Betrachtung insgesamt der 6-fache Rechenaufwand einer normalen Multiplikation n-stelliger Zahlen.

3.2 Montgomery-Multiplikation

Eine gänzliche andere Idee besteht darin, die Modulo-Rechnung direkt in den Multiplizierer zu integrieren. Dies geschieht bei der sogenannten Montgomery-Multiplikation, die in einem transformierten Zahlenraum ausgeführt wird. Die Argumente a und b werden zunächst in den Montgomery-Raum transformiert und nach der dort auszuführenden Montgomery-Multiplikation wird das Ergebnis wieder zurücktransformiert.
Da wir beim Square-and-Multiply Verfahren hintereinander sehr viele Rechnungen der Art $a \cdot b \bmod m$ ausführen müssen, bietet es sich an, ganz am Anfang alle Daten in den Montgomery-Raum zu transferieren, dann alle Rechnungen des Square-and-Multiply-Verfahrens dort auszuführen und erst am Schluß das Endergebnis zurückzutransformieren. Bei dieser Vorgehensweise ist der Transformationsaufwand sicher vernachlässigbar.



Die Grundidee der Montgomery-Transformation besteht darin, alle Daten mit einem Faktor r zu multiplizieren. Im Montgomery-Raum sind also alle Werte um diesen Faktor zu groß. Typischerweise benutzt man für r eine Zweierpotenz (allgemein: Potenz der jeweiligen Zahlenbasis), so daß sich die ganzzahlige Division durch r einfach durch Shiften realisieren läßt.

Da die Transformation $(x \rightarrow r \cdot x)$ linear ist, kann man im Transformationsraum normal addieren und subtrahieren. Bei der Montgomery-Multiplikation berechnet man prinzipiell das Produkt $(r \cdot a) \cdot (r \cdot b)$ und sorgt durch Addition geeigneter Vielfache von m dafür, daß das Resultat durch r teilbar ist. Durch Shiften (Division durch r) beseitigt man schließlich den einen überflüssigen Faktor r und erhält als Ergebnis einen Wert, der modulo m zu $r \cdot a \cdot b$ äquivalent ist. Damit liegt das gewünschte Ergebnis als Zahlenwert im Montgomery-Raum vor.

Im folgenden soll dieses Verfahren an einem Zahlenbeispiel verdeutlicht werden.

Beispiel: $a = 144$; $b = 127$; $m = 351$.

Es soll also $(a \cdot b) \bmod m = (144 \cdot 127) \bmod 351 = 36$ berechnet werden. Für die Demonstration rechnen wir im Dezimalsystem und benutzen für r die 10-er Potenz $r = 1000$, weil sämtliche für die Berechnung relevanten Werte mit drei Dezimalziffern darstellbar sind.

Wenn wir a in den Montgomery-Raum transformieren, erhalten wir $(144 \cdot 1000) \bmod m = 90$ und für b ergibt sich $(127 \cdot 1000) \bmod m = 289$. Die Modulorechnung dient dabei lediglich dazu, unnötig große Zahlenwerte zu vermeiden. Das erwartete Ergebnis 36 hat im Montgomery-Raum die Darstellung $(36 \cdot 1000) \bmod m = 198$.

Die Montgomery-Multiplikation von 90 und 289 kann dann beispielsweise folgendermaßen ablaufen: Wir berechnen $90 \cdot 289$, indem wir die 289 gemäß der Schulmethode ziffernweise von rechts nach links abarbeiten. Zusätzlich summieren wir in jedem Schritt Vielfache von m, so daß die drei hinteren Stellen der Ergebnisses zu 0 werden. Streichen wir dann diese drei letzten Stellen, so ergibt sich der erwartete Ergebniswert 198.

9	0	·	2	8	9	
			8	1	0	$90 \cdot 9$
					0	$+0 \cdot 351$
			8	1	**0**	
		7	2	0		$90 \cdot 8$
	3	1	5	9		$+9 \cdot 351$
	3	9	6	**0**	**0**	
	1	8	0			$90 \cdot 2$
1	4	0	4			$+4 \cdot 351$
1	9	8	**0**	**0**	**0**	

Bei der Implementierung des Verfahrens kann man natürlich auf die Speicherung der entstehenden hinteren Nullen verzichten und stattdessen die Zwischenergebnisse nach jedem Schritt um eine Position nach rechts shiften. Damit erreicht man, daß sämtliche auftretenden Zahlen die gleiche Länge haben. Besonders einfach wird die Implementierung, wenn man mit Binärzahlen arbeitet. Dann werden sämtliche auftretenden Multiplikationen mit Ziffern trivial und man braucht nur noch Additionen. Im folgenden ist der entstehende Ablauf durch ein "bc"-Programm dargestellt.

```
/* Vereinfachte Montgomery-Multiplikation zur Basis 2 */
/* a und b sind transformierte Daten                  */
define mp(a, b[], m) {
  auto i;
  y=0;
  for (i=0; i<n; i++) {
    if (b[i] == 1) y += a;
    if (y[0] == 1) y += m;
    y = y/2;
  }
return(y);
}
```

Diese Routine "mp" kann außer für die Montgomery-Multiplikation auch für die Montgomery-Transformation eingesetzt werden. Will man eine Zahl x in den Montgomery-Raum transformieren, so ruft man einfach "mp(x, (r*r)%m, m)" auf und erhält $y = (r \cdot x) \bmod m$. Die Rücktransformation kann mit Hilfe des Aufrufs "mp(y, 1, m)" erfolgen.
Auf diese Weise können also sehr viele für den RSA-Algorithmus benötigte Operationen mit der gleichen relativ kleinen Hardware realisiert werden.

3.2.1 Voraussetzungen der Montgomery-Multiplikation

Eine wichtige Voraussetzung des Montgomery-Verfahrens ist, daß die Argumente a und b nicht wesentlich größer als m sein dürfen. (Die genaue Formulierung

ist in [8] angegeben.) Unter dieser Voraussetzung ist sichergestellt, daß auch das berechnete Ergebnis nicht wesentlich größer als m ist.
Tatsächlich kann es vorkommen, daß das berechnete Ergebnis nicht exakt dem gewünschten Wert $a \cdot b \bmod m$ entspricht, sondern um m zu groß ist. Dann ist das Ergebnis also lediglich modulo m korrekt. Auf die eigentlich erforderliche Fehlerkorrektur durch eine nachträgliche Subtraktion von m kann man aber immer dann verzichten, wenn die Daten wieder als Argumente weiteren "mp"-Operation benutzt werden. Dann ist nämlich garantiert, daß sich der Fehler nicht weiter vergrößert. Erst zum Schluß der gesamten RSA-Berechnung muß gegebenenfalls m noch einmal subtrahiert werden.
Ein Nachteil der Montgomery-Methode besteht darin, daß sie nur dann funktioniert, wenn die Transformationskonstante r und der Modul m zueinander teilerfremd sind. Wenn wir für r eine Zweierpotenz benutzen, dann bedeutet dies, daß die Größe m ungerade sein muß. Diese Einschränkung ist für das RSA-Verfahren nicht gravierend und soll zugunsten der einfachen Implementierung inkauf genommen werden.

3.2.2 Montgomery-Modulo-Rechnung

Der Vollständigkeit halber soll noch kurz auf auf einen Spezialfall der Montgomery-Multiplikation eingegangen werden, der auch zum Verständnis der Verfahrens hilfreich ist.
Wenn wir die Montgomery-Multiplikation zur Berechnung von $(x \cdot 1) \bmod r$ benutzen, dann ist die Multiplikation trivial und es wird lediglich eine Modulorechnung durchgeführt. Als Argumente für "mp" verwenden wir dann die in den Montgomery-Raum transformierte 1, d.h. den Wert $r \bmod m$. Die Größe x transformieren wir ausnahmsweise nicht, so daß im Ergebnis der Faktor r fehlen wird. Dies bedeutet, daß der Aufruf "mp(r%m,x)" als Resultat das gewünschte Endergebnis $x \bmod m$ liefern wird, ohne daß eine Rücktransformierung erforderlich ist.
Im folgenden ist eine bc-Darstellung dieses Sonderfalls für das Zahlenbeispiel $x = 20000$ und $m = 23$ angegeben. Man beachte, daß alle auftretenden Summanden die Länge von m haben. Die Länge des Arguments x geht lediglich in die Berechnung der Konstanten $rm = r \bmod m$ und in die Laufzeit ein.
Man beachte, daß bei dieser Art der Modulo-Rechnung kein Vergleich von Datenwörtern erforderlich ist und selbst das Zeitverhalten der Additionen unkritisch ist, weil nur das niederwertige Bit von y direkt weiterverwendet wird.

```
/* Berechnung von x%m mit Hilfe der Montgomery-Technik */
x=20000          /*  x= grosse Zahl */
m=23             /* kleiner Modul   */
/* Berechnung von Konstanten */
n=log(x,2)+1
r=2^n
rm= r%m;

/* --------- Montgomery-Modulo-Rechnung --- */
convert_vector(x,xf[],n) /* Erzeugt die Binaerdarstellung xf von x */
y=0;
for (i=0; i<n; i++) {
 y= y + rm*xf[i];
 y= y + (y%2)*m;
 y= y/2;
}
"Ergebnis der Montgomery-Modulo-Rechnung="; y
```

4 Hardware-Entwurf

Der RSA-Prozessor soll, wie in Abschnitt 3.2 dargestellt, "mp"-Operationen im Binärsystem durchführen. Dazu braucht man im wesentlichen nur ein Addierwerk für n-stellige Binärzahlen. Erste Prototypen eines solchen Chips wurden für kleine Datenwortlängen bereits mit Hilfe von XILINX-FPGAs realisiert. So liegt ein Demonstrationsboard (siehe [3]) mit 4 Hexadezimalanzeigen und einer Hex-Tastatur vor, mit dem man ähnlich wie auf einem Taschenrechner $a^b \bmod m$ für 8-stellige Binärzahlen berechnen lassen kann. Je nach Programmierung der FPGAs wird die Rechnung mit elementarer Mathematik, oder der Kehrwert-Methode (siehe [6]) oder der Montgomery-Methode (siehe [8]) realisiert.
Da sich die Montgomery-Implementierung als die flächengünstigste Variante herausgestellt hat, wurden weitere FPGA-Prototypen nur noch nach dieser Methode realisiert. Diesmal wurde der Chip mit einer RS232-Schnittstelle und einer ASCII-Hex-Konvertierung ausgestattet, so daß er bequem als Co-Prozessor an einer Workstation betrieben werden kann.
Die folgende Tabelle zeigt, wieviele CLBs (CLB=configurable logic block) dabei jeweils in Abhängigkeit von der Datenwortlänge n für das "mp"-Rechenwerk, die serielle Schnittstelle (mit Datenkonvertierung) und das Steuerwerk (zum Potenzieren) benötigt werden.

n	"mp-Rechenwerk"	Kommunikation	Steuerwerk	Total
8	38	85	28	158
16	72	99	48	234
24	104	110	71	313
32	139	124	91	445

Derzeit entsteht eine neue Variante des RSA-Chips, die sich auch für sehr große Wortlängen eignen soll. Nach ersten Versuchen in [11], mit einem "mp"-Rechenwerk konstanter Größe wird jetzt ein spezielles "mp"-Rechenwerk linearer Größe eingesetzt, bei dem der Akkumulator in k-Blöcke der Länge n/k unterteilt wird, so daß jeweils nur relativ kurze Zahlen pro Takt addiert werden müssen. Auf diese Weise wird eine Addition auf n/k Takte verteilt und es finden n/k verschiedene Additionen gleichzeitig statt.
Ein kleines technisches Problem ist dabei die korrekte Zusammenfassung der zeitlich versetzt anfallenden Zwischenergebnisse. Werden nämlich Teilberechnungen um i Takte verzögert ausgeführt, so müssen die Ergebnisse in einen um inzwischen i Stellen geshifteten Akkumulator geschrieben werden. Inzwischen wurden diese Detailprobleme gelöst und die Korrektheit des neuen Verfahrens wurde mit Hilfe von "bc"-Routinen demonstriert. Die VHDL-Modellierung des neuen RSA-Chips ist noch in Arbeit.
Neben dem "mp"-Rechenwerk soll der neue RSA-Chip auch über ein RAM verfügen, das als Schnittstelle zwischen zwischen dem Rechenwerk und der Chipumgebung fungiert. Die Kommunikation zwischen Rechenwerk und Speicher bzw. zwischen Speicher und Außenwelt soll blockweise erfolgen. Die dafür erforderliche Zeit wird gegenüber der eigentlichen Rechenzeit vernachlässigbar sein.

5 Zusammenfassung und Ausblick

Es wurde gezeigt, wie man mit Hilfe der Montgomery-Multiplikation die Grundoperation eines RSA-Chips ausführen kann. Für kleine Wortlängen wurde diese Technik in VHDL-Schaltungsentwürfen realisiert und anhand von FPGA-Prototypen erprobt. Dabei hat sich insbesondere auch die implementierte Rechnerschnittstelle bewährt mit deren Hilfe die RSA-Chips ähnlich wie Coprozessoren von einer Workstation aus angesprochen werden können.
Derzeit wird ein neuer Schaltungsentwurf erarbeitet, der sich auch für extrem große Datenwortlängen eignet und der auch als ASIC gefertigt werden soll. Bei diesem Chip sollen neben dem Potenzieren auch Teiloperationen, wie Multiplikation, Modulo-Rechnung, Addition,... nach außen verfügbar gemacht werden, so daß der Chip als universeller Coprozessor für die Modulo-Arithmetik großer Zahlen genutzt werden kann. Auf dies Weise ist es auch denkbar, spezielle Softwarelösungen für das RSA-Verfahren, die beispielsweise für die Verwendung spezieller Schlüssel optimiert sind, auf einem Rechner zu implementieren und sämtliche zeitaufwendigen Operationen mit langen Daten auf den Coprozessor auszulagern. Außerdem ist mittelfristig geplant, die bisherige RS232-Schnittstelle des Chips durch eine schnellere PCI-Bus Schnittstelle zu ersetzen.

Literatur

[1] J.C. Bajard, L.S. Didier, P. Kornerup. *An RNS Montgomery Modular Multiplication Algorithm.* LIM-URA CNRS 1787, Université de Provence, France, 1996.

[2] A.Beutelspacher, J. Schwenk, K.D. Wolfenstetter. *Moderne Verfahren der Kryptographie.* Vieweg-Verlag, 1995.

[3] R. Haag. *Modellierung von Verschlüsselungshardware in VHDL.* Diplomarbeit, WS 97/98, Mikrosystemtechnik, FH Furtwangen, 1998.

[4] B. Ljusanin. *ReNaiSsance a RNS based vector co-processor.* Diplomarbeit SS98, Mikrosystemtechnik, FH Furtwangen, 1998.

[5] P.L. Montgomery. *Modular multiplication without trial division.* Mathematics of Computation, 44(170):519-521, 1985.

[6] M.J. Norris, G.J. Simmons. *Algorithms for High-Speed Modular Arithmetic.* Congressus Numerantium, Vol. 31, pp. 153-163, 1981.

[7] R.L. Rivest, A. Shamir, L. Adleman. *A method for obtaining digital signatures and public-key cryptosystems.* Comm. ACM, 21(2):120-126, February 1978.

[8] J. Sauerbrey. *Langzahlmodulo-Arithmetik für kryptographische Verfahren.* DUV (Deutscher Universitäts Verlag), 1993

[9] H. Sedlak. *The RSA Cryptography Processor.* Eurocrypt '87, Amsterdam, LNCS 304, 1987.

[10] K.W. Tse, T.I. Yuk, S.S. Chan. *Implementation of the Data Encryption Standard Algorithm with FPGAs.* In *More FPGAs*, Abingdon EE&CS Books, Oxford, 1994.

[11] O. Winker. *Entwurf eines Kryptographie-Chips.* Diplomarbeit SS98, Mikrosystemtechnik, FH Furtwangen, 1998.

[12] R. Wobst. *Abenteuer Kryptologie.* Addison-Wessley, 1997.

FHOP V2.0

Entwicklung der 2.Generation eines 16Bit Mikroprozessor-Kerns in VHDL auf der Basis des FHO-Prozessors FHOP

Markus Fischer
W. Vollmer, Dirk Jansen, ASIC-Design-Center
Fachhochschule Offenburg, Badstr. 24, 77652 Offenburg
Tel. 0781/205-267, Fax 0781/205-242,
E-Mail: d.jansen@fh-offenburg.de

Einführung

Der Entwurf von Integrierten Schaltungen gestaltet sich immer schwieriger. Es ist notwendig in jeder Applikation eine Steuerbaugruppe zu implementieren, die das Zusammenarbeiten verschiedener Komponenten regelt. Der Entwicklungsaufwand dieser Baugruppe ist zeit- und arbeitsintensiv und birgt zudem ein hohes Fehlerrisiko. Besser wäre es, wenn es eine flexible, universelle Steuereinheit geben würde, die je nach Anwendung konfiguriert werden kann. Diese Steuereinheit stellt nun im weitesten Sinne einen Prozessor-Kern dar, der über ein Programm die Steuerfunktionen übermittelt bekommt.

An der Fachhochschule Offenburg war man Anfang der 90er an einem Prozessor-Kern zu Steuerungszwecken interessiert. Es gab zu dieser Zeit schon einige Mikroprozessor-Zellen auf dem Markt, jedoch waren dies industrielle Typen wie z.B. die 8051-Serie. Diese Zellen waren alle in spezieller Full-Custom-Technologie hergestellt, sehr teuer und für Hochschulen gab es meistens keine Lizenzen. Außerdem waren diese Prozessoren für die Anwendungen meistens überdimensioniert und beanspruchten auch dementsprechend viel Platz auf einem ASIC. Dieser stand nicht in Relation zum Platzverbrauch der eigentlichen Applikation.

Aus diesen Gründen wurde 1994 an der Fachhochschule Offenburg das FHOP-Projekt gestartet. Inerhalb von zwei Diplomarbeiten wurde der erste Prozessor-Kern entwickelt[1,2]. In folgenden Arbeiten wurde der Kern weiterentwickelt und eine Software-Entwicklungsumgebung mit Assembler, Simulator, sowie einem C-Compiler geschaffen [3,4,5]. Der Prozessor-Kern steht heute als Hardmacro zur Verfügung und kann leicht in Standard Zellen ASICs eingefügt werden. Dies wurde auch schon in unterschiedlichen Projekten durchgeführt: einer Chip-Karten-Applikation, einem generellen Mikrokontroller und einem PSK-Modem[6,7,8].

Leistungsmerkmale des FHOP V1.0

- 16Bit Architektur(16Bit ALU)
- 64kB großer Adressbereich
- 6 Register
- Interrupt-, Waitstate- und Hold-Fähigkeit
- Befehle für Arithmetic-, Logik-, Schiebe-, Sprung- und Transfer-Operationen
- Insgesamt 115 Befehle, jeder 8Bit breit mit optionalen Operanden, max. 3Byte
- Taktraten bis zu 50MHz

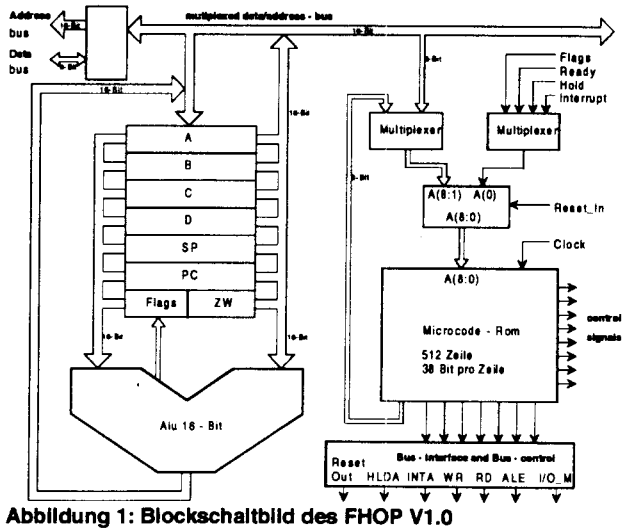


Abbildung 1: Blockschaltbild des FHOP V1.0

Abb.1 zeigt das Blockschaltbild des FHOP-Kerns. Man erkennt die 6 Register(A, B, C, D, SP, PC), sowie die 16Bit breite ALU(Arithmetic Logic Unit). Das Microcode-ROM steuert die ALU, sowie den Datenfluß zwischen ALU, Registern und externen Komponenten.

Der Befehlssatz ist ein Kompromiß zwischen einer RISC und CISC Philosophie. Dies entstand aus der Entscheidung nicht mit anderen Prozessoren kompatibel zu sein, um Lizenz-Probleme zu vermeiden. Man war dadurch gezwungen, eigene Software-Entwicklungstools zu entwickeln. Bis heute sind für den FHOP ein Assembler, ein Simulator und ein C-Compiler zum Programmieren und Debuggen vorhanden. Diese Tools laufen unter Microsoft Windows95/NT.

Im Laufe der letzten Jahre sind einige Verbesserungen immer wieder aufgeschoben worden, so daß es nun an der Zeit ist einen überarbeiteten Kern zu erstellen. Die Änderungen sehen wie folgt aus:

- Separater externer 16Bit Daten-/Adressbus

  Eine konsequente 16Bit-Busarchitektur verbessert die Leistung des bisher im Mulitplex betriebenen Datenbusses und ermöglicht den direkten Anschluß von 16Bit Komponenten.

- Austausch des Microcode-ROMs durch eine kombinatorische/sequentielle Schaltung

  In den Microcode wurde der größte Teil des Steueraufwandes des FHOPs verlagert, so daß der Hardwareaufwand sehr gering gehalten werden konnte. Nachteil des ROMs ist allerdings, daß nicht alle Technologien einen ROM-Generator zur Verfügung stellen und man so auf die bisher benutzte ES2 Technologie beschränkt ist. Eine nur aus State-Machine-Strukturen aufgebaute Steuereinheit ist auf jede Zieltechnologie synthetisierbar.

- Erweiterung des Funktionsumfanges durch Aufnahme neuer Befehle bei Erhaltung der Abwärtskompatibilität

- Optimierund der Befehle ⇒ weniger Takte werden benötigen ⇒ Leistungssteigerung

- Einfügen eines Hardware-Multiplizierers

- Synthesefähig auf beliebige Technologien

  Um dies zu erreichen wird der Design komplett in VHDL beschrieben.

Der FHOP V 2.0

Abb. 2 zeigt das Blockschaltbild des Nachfolge-Prozessors FHOP V 2.0, in dem die wesentlichen Änderungen gegenüber der bisherigen Version dargestellt sind. Der neue Kern besteht nun aus den Teilen Datenpfad, Steuerwerk(hier CTRL_ UNIT), die bisher auch vorhanden waren, und der neu hinzugekommen Busentkoppellungseinheit ‚IQ_ MEM_IO'. Sie regelt den Datenaustausch mit externen Peripherieeinheiten und dem Speicher.

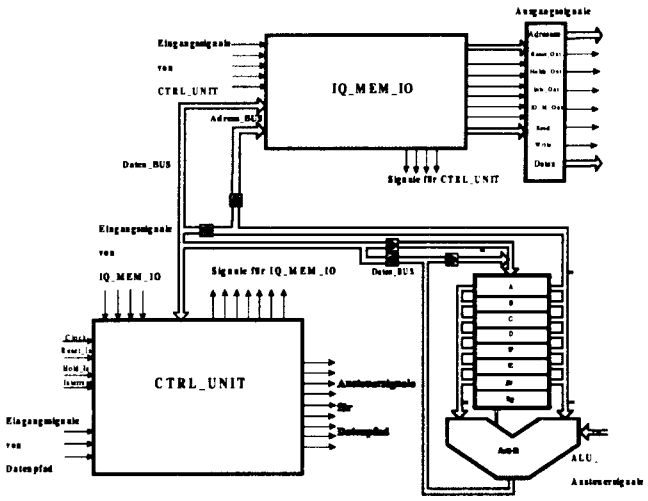


Abbildung 2: Blockschaltbild des FHOP V2.0

Die ‚IQ_MEM_IO' Einheit ist notwendig geworden, da die Abwärtskompatibilität zu der ersten Version des FHOPs gefordert war, so daß der OP-Code mit seinen 8Bit nicht verändert werden durfte. Allerdings ist auch gefordert, daß der externe Daten- und Adreßbus getrennt 16Bit breit sein soll, so daß immer nur 2Byte auf einmal gelesen oder geschrieben werden können. Um nun aus den 16Bit ein Byte OP-Code zu erlangen, wurde die ‚Instruction Queue'(IQ) eingeführt, in der maximal 4Byte in zwei Registern zwischengespeichert werden. In diese zwei Register werden allerdings nur Befehle gespeichert. Zugriffe auf RAM oder PORTs werden über die ‚MEM_IO' Einheit durchgeführt. Aus den Registern der IQ kann nun der OP-Code extrahiert werden, da jedes der vier Byte einzeln ausgegeben werden kann.

Auffälligste Veränderung am Datenpfad ist, daß das Zwischenregister und das Flagregister nun ebenfalls 16Bit breit sind. Dies ist die Folge der konsequenten Auslegung auf 16Bit. Weiterhin wurde ein 8x8 Multiplizierer eingeführt, mit dem über mehrere Takte, zwei 16Bit-Werte multipliziert werden können.

Steuereinheit

Eine geforderte Veränderung gab es auch im Steuerwerk, das nicht mehr aus dem Microcode-ROM besteht, sondern nun aus einer kombinatorischen/sequentiellen Schaltung. Dies hat zur Folge, daß es nun drei Teile im Steuerwerk gibt, den ‚Precoder', die ‚CTRL-State-Machine' und den ‚Decoder'(Abb. 3).

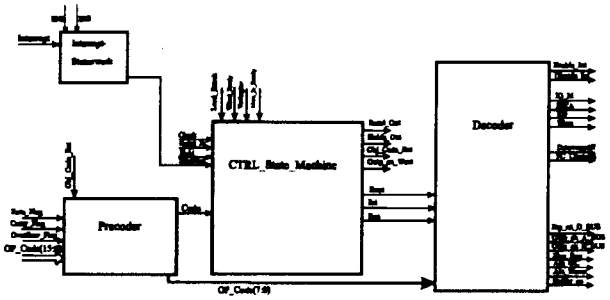


Abbildung 3: Blockschaltbild der CTRL_UNIT

Der ‚Precoder' hat die Aufgabe, aus dem 16Bit breiten Datenbus den 8Bit breiten OP-Code zu übernehmen und zu speichern, zudem kodiert er diesen in einen 4Bit Meta-Code. Diese Kodierung ist im Laufe der Entwicklung der ‚Ctrl_State_ Machine' entstanden, da jeder einzelne der 123 Befehle(115 alte + 8 neue) in den States hätte abgefragt werden müssen. Dies hätte zu einer unüberschaubar großen und undurchsichtigen State_Machine geführt, so daß die Befehle in einzelne Typen eingeteilt wurden. Die Einteilung erfolgt nun nach zwei Kriterien, in 14 Typen. Die beiden Kriterien sind: Anzahl der Bytes und Anzahl der benötigten Takte pro Befehl. Mit der Anzahl der Bytes ist die Größe eines Befehls inklusive der Erweiterung gemeint, d.h. es gibt drei Typen, 1Byte-, 2Byte- und 3Byte-Befehle. Die Anzahl der benötigten Takte, die ein Befehl zur Abarbeitung benötigt, unterteilt die Befehlstypen noch einmal.

Das Herz der Steuereinheit stellt die ‚Ctrl_State_ Machine' dar. Mit ihr werden alle Befehle umgesetzt, so daß der Decoder die Ansteuersignale für die verschiedenen Einheiten generieren kann. Ein Punkt, der die State_Machine sehr komplex gestaltet, ist die Berücksichtigung der Signale Hold, Interrupt und Ready. Dies ist notwendig, um die Abwärtskompatibilität zu gewährleisten. Außerdem ist die Struktur der State_Machine so ausgelegt, daß auf relativ einfache Weise neue Befehle hinzugefügt und alte optimiert werden können.

Der ‚Decoder' erhält den 8Bit breiten OP-Code und ein 4Bit breites Signal aus der ‚Ctrl_State_ Machine'. Aus diesen Signalen generiert dieser dann die Ansteuersignale für den Datenpfad, die ‚IQ_MEM_IO' und die Externen Einheiten.

Die IQ_MEM_IO Einheit

Die Forderung der Abwärtskompatibilität steht im Konflikt mit dem konsequent auf 16Bit ausgelegten Design, da der OP-Code weiterhin 8Bit breit gehalten werden muß. Da nun bei jedem Zugriff auf den Speicher immer 16Bit gelesen werden, müssen diese aufgetrennt werden, um einen OP-Code zu erhalten. D. h. die Daten müssen in einem Register gespeichert werden. Da nun aber Befehle nicht nur aus ihrem OP-Code bestehen, sondern auch eine Erweiterung haben können, die max. zwei Byte groß ist, wurde die ‚Instruction Queue' entwickelt (Abb. 4). In ihr sind zwei 16Bit Register die jeweils 4Byte, OP-Code und Erweiterungen, speichern. Weiterhin ist in ihr Logik enthalten, die es ermöglicht, daß die zwei Register geladen werden können, während ein Befehl abläuft. Im Optimalfall bedeutet dies, daß der neue OP-Code zu dem Zeitpunkt, an dem die Ctrl-Einheit ihn anfordert, schon bereit steht, was zu einer Performance-Steigerung führt.

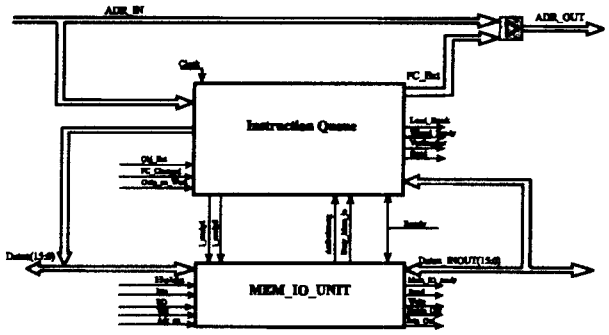


Abbildung 4 Blockschaltbild der IQ_MEM_IO

Diese Leistungs-Verbesserung wird nicht immer ausgeschöpft, da es bei der Abarbeitung eines Befehls ebenfalls einen Zugriff auf die externen Busse geben kann, um Daten zu lesen oder zu schreiben. Um diesen Konflikt zu umgehen mußte eine zweite Einheit entwickelt werden, die diese Zugriffe auf die Busse regelt, die MEM_IO. Hat die ‚Instruction Queue'-Logik ihren letzten Ladevorgang abgeschlossen wird sie gesperrt und die MEM_IO führt ihre Operation aus. Hierbei übergibt sie entweder die Daten vom Datenpfad an den Buscontroller weiter oder umgekehrt. Nach Beendigung der Operation gibt die MEM_IO die IQ wieder frei.

FHOP-Design-Kit

Der FHOP-Design-Kit beinhaltet den Kern in verschiedenen Beschreibungen, einige Peripherie-Module, sowie die benötigten Software-Tools und Dokumentationen, um die Komponenten in einem eigenen Design zu integrieren. Im einzelnen sind dies(Abb. 5):

- Alle Schaltpläne im EDIF-Netzlisten-Format, um diese in andere System zu Importieren
- FHOP-Kern und alle Peripherie-Module als VHDL-Beschreibung
- Geroutete Hardmacros von FHOP-Kern und Peripherie-Module im GDS-II-FORMAT
- Software-Tools: Assembler, Simulator mit einigen Virtual-Project-Komponents(VPC), C-Compiler
- Software-Bibliothek mit oft verwendeten Routinen als FHOP-Assembler-Source
- Dokumentation des FHOP-Kerns, der Peripherie-Module und der Software-Tools

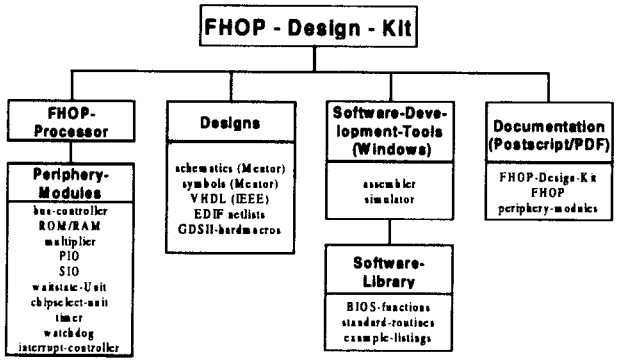


Abbildung 5: Aufbau des FHOP-Design-Kit

Die bisherige FHOP-Version 1.1 bleibt weiterhin verfügbar. Derzeitiges Routing 1.5mm² in Mietec 0.5µm Technologie.

Anwenden des FHOP-Design-Kit

Die Entwicklung eines eigenen Designs, mittels des FHOP-Design-Kits, erfolgt in vier Schritte(Abb. 6):

- Erstellen des Hardware_Designs, als Schaltplan oder in VHDL
- Programmieren der FHOP-Software
- Cosimulation von Hardware und Software
- Chip-Layout fertigen

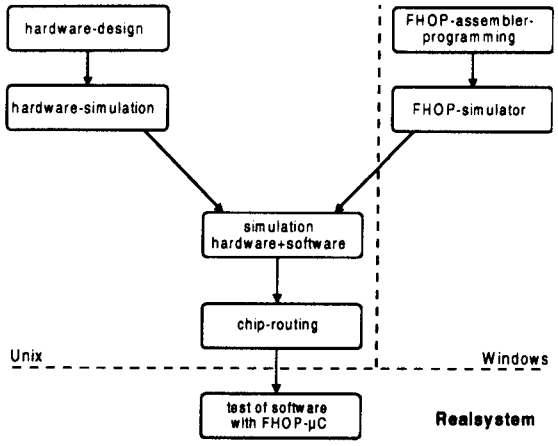


Abbildung 6: Hardware-Software-Codesign mit dem FHOP-Design-Kit

Stand:

- Derzeit Synthese und Optimierung
- Ab Frühjahr 1999 in Design-Kit integriert
- Zunächst in Mietec 0.5µm vorhanden
- Vorraussichtlich ab Herbst 1999 in AMS 0.6µm
- Peripheriemodule werden angepaßt auf 16Bit Architektur, sofern erforderlich

Literatur:

[1] Th. Gieringer, „Entwicklung der Steuerbaugruppe eines 16Bit Mikroprozessor-Chips mit VHDL", FH-Offenburg 1994

[2] F. Zimpfer, „Entwicklung des Datenpfads eines 16Bit Mikroprozessor-Chips mit VHDL", FH-Offenburg 1994

[3] C. Sattler, „Entwicklung eines Assemblers für den Mikroprozessor FHOP unter Windows", FH-Offenburg 1995

[4] J. Heiderich, „Integration eines Simulators in die Entwicklungsumgebung für den an der FHO entwickelten Mikroprozessor FHOP", FH-Offenburg 1996

[5] O. Bischoff, „Entwicklung eines C-Compilers für den FHOP-Prozessor", FH-Offenburg 1997

[6] T. Klumpp, „Thermologger: Entwurf eines ASIC - Bausteins zur komprimierten Aufzeichnung von Meßdaten", FH-Offenburg 1995

[7] W. Vollmer, „Aufbau eines Mikrocontrollersystems auf der Basis des Mikroprozessorkernels FHOP", FH-Offenburg 1996

[8] D. Vogel, „Entwurf eines Controllers für Modelleisenbahnsteuerung mit Netzwerkfunktionen und Realisierung als ASIC", FH-Offenburg 1996

Statische Modelle zur Beschreibung des Großsignalverhaltens von Verstärkern

R. Quell, G. Forster
Fachhochschule Ulm, Prittwitzstraße 10, 89075 Ulm

Zusammenfassung

Ein Ansatz zur weitgehend automatischen Bestimmung von symbolischen Ausdrücken zur Beschreibung des Großsignalverhaltens von Verstärkern wird vorgestellt. Ausgehend vor einer bekannten Übertragungskennlinie werden mit Hilfe eines Mathematica-Notebooks Gleichungen für die Berechnung der harmonischen Verzerrung, des Kompressionspunktes und der Intermodulationskenngrößen abgeleitet. Hierfür wird die Übertragungskennlinie in eine Potenzreihe entwickelt. Die als Antwort auf sinusförmige Eingangssignale entstehenden Spektralanteile werden mit Hilfe trigonometrischer Umformungen gewonnen. Als Beispiel dient ein Differenzverstärker.

Einleitung

Auf dem Gebiet der Mobilkommunikation geht der Trend zu immer kompakteren und verlustleistungsärmeren Systemen bei gleichzeitig erweiterter Funktionalität. Eine Schlüsselfunktion spielt hierbei der rauscharme Eingangsverstärker (Low Noise Amplifier, LNA). Die entscheidende Anforderung an einen solchen Verstärker ist, neben der geforderten hohen Bandbreite, ein möglichst großer Dynamikbereich. Dieser wird bestimmt durch das Rauschen auf der einen Seite und das Großsignalverhalten auf der anderen Seite.

Um den immer kürzer werdenden Entwicklungszyklen gerecht zu werden, gewinnen symbolische Dimensionierungsgleichungen wachsende Bedeutung. Sie erlauben es, Dimensionierungsabläufe (jedenfalls teilweise) zu automatisieren und damit das Schaltungswissen innerhalb kürzester Zeit verfügbar zu machen. Ziel dieser Arbeit ist es deshalb, symbolische (analytische) Gleichungen für die Beschreibung des Großsignalverhaltens von rauscharmen Verstärkern zu entwickeln.

1 Großsignalparameter für rauscharme Verstärker (LNA)

1.1 Harmonische Verzerrungen zweiter und dritter Ordnung

Unter der harmonischen Verzerrung versteht man das Verhältnis aus dem Effektivwert einer Oberwelle zum Effektivwert der Grundwelle einer Signalspannung. Das Verhältnis wird in dBc ausgedrückt (Dezibel below Carrier). Es ist eine Funktion der Eingangssignalleistung. Diese wird daher als Testbedingung angegeben. Anstelle des Effektivwerts kann auch die Amplitude stehen.

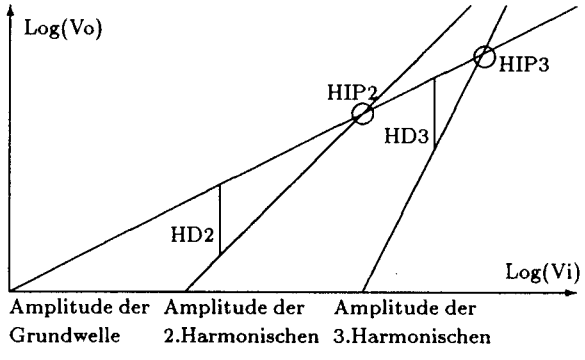


Abbildung 1: Definition der harmonischen Verzerrungen zweiter und dritter Ordnung (HD2, HD3)

In der Abbildung 1 ist die Definition graphisch dargestellt. Es ist die Ausgangsspannung über der Ein-

gangsspannung in doppelt logarithmischem Maßstab aufgetragen. In dem Diagramm sind die Amplitude der Grundwelle sowie die Amplituden der zweiten und dritten Harmonischen gezeigt. Die harmonische Verzerrung zweiter Ordnung HD2 läßt sich am Abstand der beiden linken Linien ablesen, der harmonische Schnittpunkt (Harmonic Intercept Point HIP2) an der Lage des Schnittpunkts. Entsprechendes gilt für die harmonische Verzerrung dritter Ordnung und den harmonischen Schnittpunkt dritter Ordnung.

1.2 Harmonische Verzerrungen zweiter und dritter Ordnung einer Basisschaltung

Bei der Anwendung einer vereinfachten Übertragungskennlinie können die harmonischen Verzerrungen auf einfache Weise berechnet werden. Dies soll am Beispiel einer Basisschaltung (Abbildung 2) gezeigt werden [1].

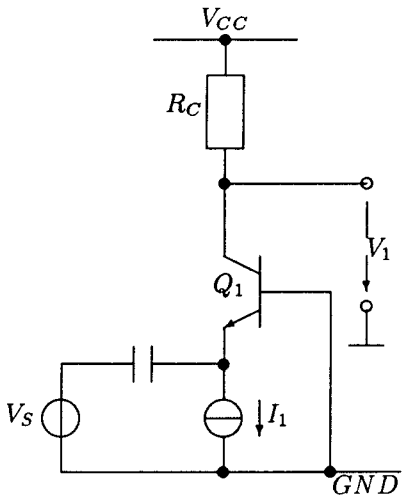


Abbildung 2: Basisschaltung

Ohne Berücksichtigung von Bahnwiderständen gilt für den Kollektorstrom eines Bipolartransistors:

$$I_C = I_S * e^{\frac{V_{BE}}{V_T}} \tag{1}$$

Die Basis-Emitter-Spannung V_{BE} setzt sich aus einer Gleichspannung, resultierend aus dem Emitterstrom I_1 und der Signalspannung V_S (Wechselspannung) zusammen. Wird der Basisstrom vernachlässigt, so gilt:

$$V_{BE} = V_T * \ln \frac{I_1}{I_S} + (-V_S) \tag{2}$$

Eingesetzt in die vorhergehende Formel ergibt dies:

$$I_C = I_S * e^{\ln \frac{I_1}{I_S} + \frac{(-V_S)}{V_T}} = I_1 * e^{\frac{(-V_S)}{V_T}} \tag{3}$$

Für die Ausgangsspannung gilt dann:

$$V_1 = V_{CC} - R_C * I_C \tag{4}$$

$$\boxed{V_1 = V_{CC} - R_C * I_1 * e^{\frac{(-V_S)}{V_T}}} \tag{5}$$

Die Ausgangsspannung, aufgetragen über der Eingangsspannung, ist in Abbildung 3 zu sehen. Die reale Kennlinie weist allerdings eine deutliche Abweichung im negativen Bereich von V_S auf (gestrichelt gezeichnet), weil hier der Transistor in den Zustand der Sättigung gerät und Gleichung 1 nicht mehr gilt. Dieser Bereich entzieht sich jedoch der Handrechnung und soll deshalb, wie anfangs erwähnt, außer Betracht bleiben. Voraussetzung für eine Frequenz-

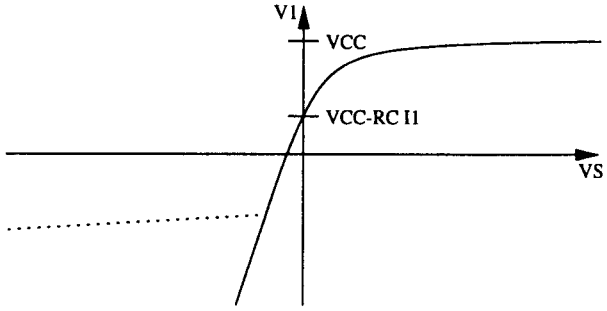


Abbildung 3: Transferkennlinie der Basisschaltung nach Abbildung 2 für Wechselsignal

analyse ist nun, daß die Übertragungskennlinie in eine Potenzreihe entwickelt wird. Wir erhalten:

$$V_1 = V_{CC} - R_C * I_1 (1 + \frac{-V_S}{V_T} + \frac{1}{2}(\frac{-V_S}{V_T})^2 + \frac{1}{6}(\frac{-V_S}{V_T})^3 + \ldots)$$

Dies resultiert in der Ausgangsspannung in Polynomdarstellung:

$$V_1 = V_{1DC} + a_1(-V_S) + a_2(-V_S)^2 + a_3(-V_S)^3 + \ldots$$

mit:

$$V_{1DC} = V_{CC} - R_C I_1$$

$$a_1 = -\frac{R_C I_1}{V_T}$$

$$a_2 = -\frac{R_C I_1}{2V_T^2}$$

$$a_3 = -\frac{R_C I_1}{6V_T^3}$$

Wird V_S sinusförmig angesetzt erhält man mit

$$-V_S = \hat{V}_S \sin \omega t$$

eingesetzt in das Polynom:

$$V_1 = V_{1DC} + a_1 \hat{V}_S \sin \omega t + a_2 \hat{V}_S^2 \sin^2 \omega t + a_3 \hat{V}_S^3 \sin^3 \omega t + \ldots$$

Um die Amplituden der zweiten und dritten Harmonischen und der Grundwelle zu erhalten, werden die Potenzen der Sinusterme trigonometrisch umgeformt:

$$\begin{aligned} V_1 = & \quad V_{1DC} && (DC Anteil) \\ + & \quad a_1 \hat{V}_S \sin(\omega t) && (Grundwelle) \\ + & \quad \tfrac{a_2 \hat{V}_S^2}{2}(1 - \cos(2\omega t)) && (2. Harmonische) \\ + & \quad \tfrac{a_3 \hat{V}_S^3}{4}(3\sin(\omega t) - \sin(3\omega t)) && (3. Harmonische) \\ + & \quad \ldots && (6) \end{aligned}$$

Hierbei wurden die trigonometrischen Beziehungen $\sin^2(\omega t) = \frac{1}{2}(1 - \cos(2\omega t))$ und $\sin^3(\omega t) = \frac{1}{4}(3\sin(\omega t) - \sin(3\omega t))$ angewandt.

Die harmonische Verzerrung zweiter Ordnung HD2 ergibt sich nun durch Vergleich der Amplitude der zweiten Harmonischen mit der Amplitude der Grundwelle:

$$\boxed{\mathrm{HD2} = \frac{\frac{a_2 \hat{V}_S^2}{2}}{a_1 \hat{V}_S} = \frac{1}{2}\frac{a_2}{a_1}\hat{V}_S = \frac{1}{2}\frac{1}{2V_T}\hat{V}_S = \frac{1}{4}\frac{\hat{V}_S}{V_T}} \quad (7)$$

Wie man sieht, wächst die harmonische Verzerrung zweiter Ordnung linear mit der Eingangsspannung. Logarithmisch hat der Abstand zwischen der zweiten Harmonischen und der Grundwelle die Steigung 1.

Die harmonische Verzerrung zweiter Ordnung erreicht 10 % wenn $0.1 = \frac{1}{4}\frac{\hat{V}_S}{V_T}$. Daraus folgt als maximal zulässige Scheitelspannung $\hat{V}_S = 0.4 V_T \approx 10$ mV bei $HD2 = 10$ %.

Die harmonische Verzerrung dritter Ordnung HD3 resultiert aus dem Vergleich der Amplitude der dritten Harmonischen mit der Amplitude der Grundwelle:

$$\boxed{\mathrm{HD3} = \frac{\frac{a_3 \hat{V}_S^3}{4}}{a_1 \hat{V}_S} = \frac{1}{4}\frac{a_3}{a_1}\hat{V}_S^2 = \frac{1}{4}\frac{1}{6V_T^2}\hat{V}_S^2 = \frac{1}{24}\frac{\hat{V}_S^2}{V_T^2}} \quad (8)$$

Die harmonische Verzerrung dritter Ordnung wächst also quadratisch mit der Eingangsspannung. Logarithmisch hat der Abstand zwischen der dritten Harmonischen und der Grundwelle die Steigung 2.

Die harmonische Verzerrung dritter Ordnung erreicht 10 % wenn $0.1 = \frac{1}{24}\frac{\hat{V}_S^2}{V_T^2}$. Daraus folgt als maximal zulässige Scheitelspannung $\hat{V}_S = \sqrt{24} V_T \approx 40$ mV bei $HD2 = 10$ %.

Der harmonische Schnittpunkt HIP2 (bzw. HIP3) ist definiert als diejenige Eingangsleistung, bei welcher die Amplitude der zweiten (bzw. dritten) Harmonischen die Amplitude der Grundwelle erreicht. Es ist also diejenige Eingangsleistung, bei welcher die harmonische Verzerrung zweiter Ordnung HD2 (bzw. dritter Ordnung HD3) den Wert 1 erreicht. Es ist dies ein theoretischer Wert, der ausgehend von dem kleinen Signal V_S durch Extrapolation bestimmt wird.

Es gilt also: $\mathrm{HD2} = \frac{1}{4}\frac{\hat{V}_S}{V_T} \equiv 1 \Rightarrow \hat{V}_S = 4V_T = 104$ mV bei T=300 K.

$$\mathrm{HIP2} = 20 \log \frac{104\ mV}{316\ mV} = -9,7\ dBm \quad (9)$$

Entsprechend gilt: $\mathrm{HD3} = \frac{1}{24}\frac{\hat{V}_S^2}{V_T^2} \equiv 1 \Rightarrow \hat{V}_S = \sqrt{24} V_T = 127$ mV bei T=300 K.

$$\mathrm{HIP3} = 20 \log \frac{127\ mV}{316\ mV} = -7,9\ dBm \quad (10)$$

Für eine Basisschaltung erhält man daher:

$$\boxed{\begin{aligned} \mathrm{HIP2} &= -9,7\ dBm \\ \mathrm{HIP3} &= -7,9\ dBm \end{aligned}} \quad (11)$$

sofern die Ansteuerung V_S direkt am Emitter erfolgt, ohne Vorwiderstand und ohne Berücksichtigung der Bahnwiderstände. Der Ruhestrom I_1 geht in diesem Fall nicht in die Berechnung ein.

Bei Leistungsanpassung am Eingang (Innenwiderstand der Quelle = Eingangswiderstand) führt erst die doppelte Spannung V_S zu der oben angesetzten Spannung am Emitter. Damit erhöhen sich die IP jeweils um 6 Dezibel:

$$\begin{aligned} \mathrm{HIP2} &= -3,7\ dBm \approx -4\ dBm \\ \mathrm{HIP3} &= -1,9\ dBm \approx -2\ dBm \end{aligned} \quad (12)$$

1.3 1-Dezibel-Kompressionspunkt P1dB

Unter dem Kompressionspunkt P1dB versteht man diejenige Eingangssignalleistung, bei welcher am Verstärkerausgang die reale Amplitude der Grundwelle

um ein Dezibel kleiner wird als die ideale Amplitude bei linearer Übertragung.

In der Abbildung 4 ist die Definition graphisch dargestellt. Es ist die Ausgangsspannung über der Eingangsspannung in doppelt logarithmischem Maßstab aufgetragen, und zwar einmal ideal, d. h. entsprechend der Kleinsignalverstärkung A_{dm}, und zusätzlich real, d. h. mit Hilfe der Frequenzanalyse berechnet. Der 1-Dezibel-Kompressionspunkt kennzeichnet diejenige verfügbare Eingangsleistung (bzw. Amplitude), bei der sich die Verstärkung der Baugruppe gegenüber dem Kleinsignalwert um 1 Dezibel reduziert. Üblicherweise wird der 1-Dezibel-Kompressionspunkt auf den Eingang bezogen.

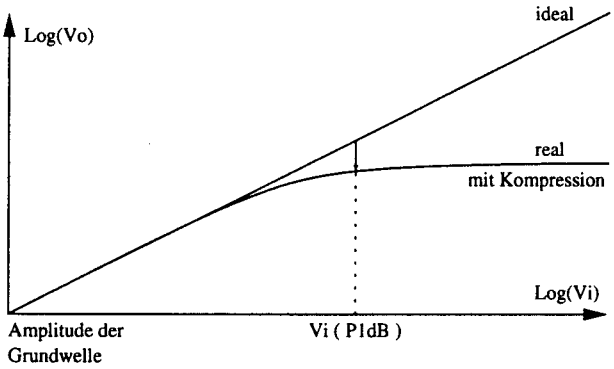


Abbildung 4: Definition des Kompressionspunktes P1dB

Die Berechnung des 1-Dezibel-Kompressionspunktes erfolgt entsprechend der Berechnung der harmonischen Verzerrungen. Die Handrechnung bei der Basisschaltung gäbe hier kein Ergebnis, da sich bei der Übertragungsfunktion mit der einfachen Exponentialfunktion die auf der einen Seite entstehende Kompression mit der auf der anderen Seite existierenden Expansion kompensiert.

1.4 Intermodulation

An den nichtlinearen Kennlinien von Übertragungsgliedern werden aus mehreren anliegenden Signalen neue, unerwünschte Signale erzeugt (siehe Abbildung 5). Der Vorgang wird *Intermodulation* [2] genannt und die entstehenden Signale Intermodulationsprodukte. Die Intermodulationsprodukte werden nach folgendem Gesetz gebildet:

$$\omega_{IM} = m_1\,\omega_1 + m_2\,\omega_2 + \cdots + m_n\omega_n$$
$$mit \quad m_v = 0, \pm 1, \pm 2, \cdots \pm n \quad und \quad n \in \mathrm{N}. \quad (13)$$

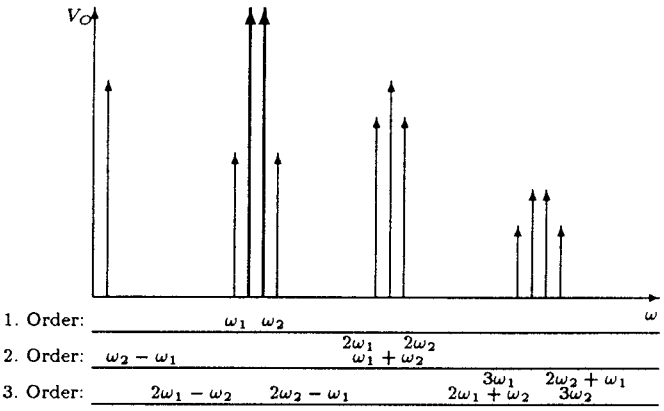


Abbildung 5: Frequenzdiagramm der Intermodulationsprodukte zweiter und dritter Ordnung

Üblicherweise wird bei Verstärkern nur der Fall $n = 2$, d. h. eine Überlagerung von zwei Frequenzen am Eingang mit m_1,m_2; bzw. $|m_1| = 2; |m_2| = 1$ betrachtet:

$$\begin{aligned} \omega_{IM} &= |\quad \omega_1 \pm \omega_2| \qquad 2.Ordnung\ (\mathrm{IM2}) \\ \omega_{IM} &= |2 * \omega_1 \pm \omega_2| \qquad 3.Ordnung\ (\mathrm{IM3}) \end{aligned} \quad (14)$$

1.5 Intermodulationsschnittpunkte zweiter und dritter Ordnung IIP2/IIP3

Unter dem auf den Eingang bezogenen Intercept Point IIP2 (bzw. IIP3) versteht man diejenige Eingangsleistung zweier Sinussignale gleicher Amplitude mit den Frequenzen ω_1 und ω_2, bei welcher am Verstärkerausgang die Amplitude des Mischproduktes mit der Frequenz $(\omega_1+\omega_2)$ bzw. $(\omega_2-\omega_1)$ für IIP2 und $(2*\omega_1-\omega_2)$ bzw. $(2*\omega_2-\omega_1)$ für IIP3 gleich groß wird, wie die ideale Amplitude einer der Grundwellen, wenn diese mit der Kleinsignalverstärkung linear übertragen würden.

Man unterscheidet zwischen Output- und Input-Intercept Point (OIP bzw. IIP), je nachdem die Angabe auf den Ausgang oder den Eingang des betrachteten Übertragungsgliedes bezogen ist [2].

In der Abbildung 6 ist die Definition graphisch dargestellt. Es ist die Ausgangsspannung über der Eingangsspannung in doppelt logarithmischem Maßstab aufgetragen. In dem Diagramm sind die Amplituden der Grundwelle ω_1 oder ω_2, der Intermodulationsprodukte bei der Frequenz $\omega_1 + \omega_2$ oder $\omega_2 - \omega_1$ und der Intermodulationsprodukte bei der Frequenz

$2\omega_1 - \omega_2$ oder $2\omega_2 - \omega_1$ gezeigt. Der Intermodulationsabstand zweiter Ordnung ist der Abstand der beiden linken Linien. Der Intermodulationsabstand dritter Ordnung ist der Abstand der rechten Linie zur linken Linie. Die Schnittpunkte der Linien sind die Intermodulationsschnittpunkte (Input Intercept Point) zweiter und dritter Ordnung. Auf ein Berechnungsbeispiel wird an dieser Stelle verzichtet. Der Ablauf ist gleich wie bei der Berechnung der harmonischen Verzerrung mit dem Unterschied, daß als Eingangssignal die Überlagerung zweier Sinussignale verwendet wird.

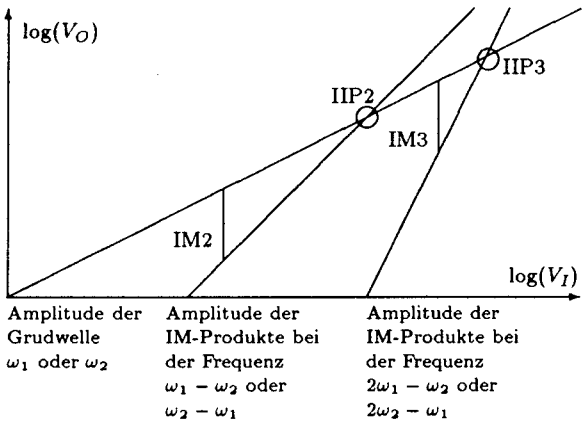


Abbildung 6: Definition der Intermodulationsschnittpunkte zweiter und dritter Ordnung (IIP2, IIP3)

Der Intermodulationsschnittpunkt dritter Ordnung IIP3 kann auch aus dem harmonischen Schnittpunkt dritter Ordnung HIP3 bestimmt werden. Es gilt:

$$\boxed{\text{IIP3} = \text{HIP3} - 4,77dB} \qquad (15)$$

für den Fall, daß kubische Nichtlinearität dominiert. Damit erhält man für die Basisschaltung ohne Vorwiderstand: IIP3 $= -7,9\ dBm - 4,77\ dBm = -12,7\ dBm$ und für die Basisschaltung mit Leistungsanpassung: IIP3 $= -1,9\ dBm - 4,77\ dBm = -6,7\ dBm$.

2 Großsignalparameter mit Mathematica am Beispiel des Differenzverstärkers

Hier wird nun dargestellt, wie die Handrechnung zur Bestimmung der Großsignalparameter durch eine Routine zur automatisierten Berechnung mit Mathematica [3] ersetzt wird. Die Routine erhält eine symbolische Übertragungsfunktion und liefert als Ergebnis symbolische Ausdrücke für die Großsignalparameter wie im obigen Abschnitt skizziert.

Die Extraktion der Großsignalparameter soll nun anhand des Differenzverstärkers in Abbildung 7 gezeigt werden [4].

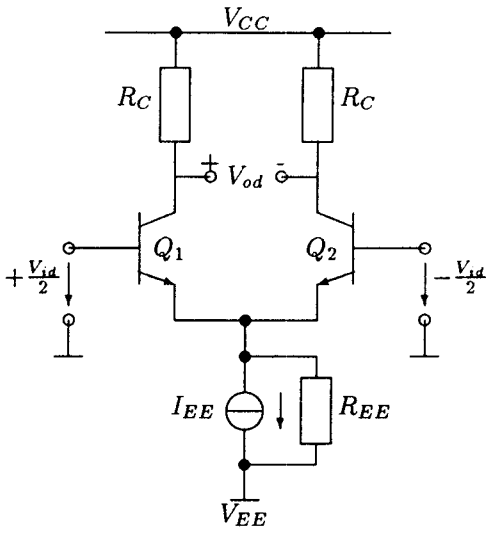


Abbildung 7: Differenzverstärker

Für die Gleichstromcharakteristik der Transistoren wird wieder die vereinfachte Gleichung 1 angesetzt. Die Sättigung wird vernachlässigt. Dies ist keine wirkliche Einschränkung, da im Differenzverstärker die Transistoren normalerweise nicht in der Sättigung betrieben werden. Die begrenzende Transferkennlinie resultiert bereits aus dem aktiven Vorwärtsbetrieb der Transistoren. Die Transferfunktion lautet [5]:

$$\boxed{V_{od} = A * I_{EE} * R_C * \tanh(-\frac{V_{id}}{2V_T})} \qquad (16)$$

wobei $A = \frac{\beta_F}{\beta_F+1}$ die Stromverstärkung zwischen Emitter und Kollektor ist.

Mit den Parametern $V_T = 26\ mV$, $R_C = 1\ k\Omega$, $I_{EE} = 10\ mA$ und $A = 1$ ergibt sich die Transferkennlinie in Abbildung 8.

2.1 Frequenzanalyse

Um die Werte der harmonischen Verzerrung zweiter und dritter Ordnung, sowie die harmonischen Schnittpunkte zweiter und dritter Ordnung und den 1-Dezibel-Kompressionspunkt zu berechnen, ist

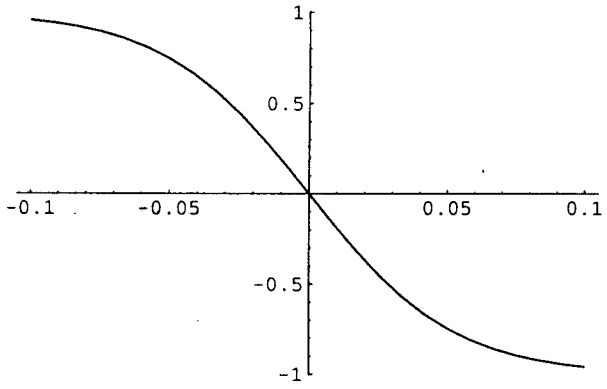


Abbildung 8: Transferkennlinie des Differenzverstärkers

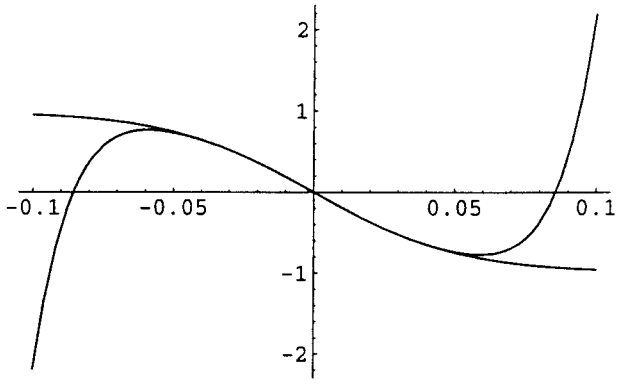


Abbildung 9: Graphischer Vergleich der Approximation mit der tatsächlichen Transferkennlinie.

es notwendig, eine Frequenzanalyse durchzuführen. Hierzu wird die Übertragungsfunktion durch eine Potenzreihe genähert. Man erhält diese zum Beispiel mit Hilfe der Taylor-Entwicklung:

$$f(x) = f(x_0) + f'(x_0)x + \frac{x^2 f''(x_0)}{2} + \\ + \cdots + \frac{x^n f^{(n)}(x_0)}{n!} \tag{17}$$

Die Potenzreihenentwicklung wird von Mathematica berechnet. Hierzu sind folgende Befehle notwendig:

$$Potenzreihe = \text{Series}[\ddot{U}bertragungsfunktion, \\ \{V_{in}, Entwicklungsstelle = 0, Grad\}] \tag{18}$$

Mit Hilfe von

$$\text{Normal}[Potenzreihe] \tag{19}$$

wird der Restterm oder Fehlerterm $O(n)$ entfernt.

Aus der Übertragungsfunktion in Gleichung 16 erhält man die Potenzreihe, wobei nach dem siebten Term abgebrochen wurde.

$$V_{od} = -\frac{A\, I_{EE}\, R_C\, V_{in}}{2\, V_T} + \frac{A\, I_{EE}\, R_C\, {V_{in}}^3}{24\, {V_T}^3} \\ -\frac{A\, I_{EE}\, R_C\, {V_{in}}^5}{240\, {V_T}^5} + \frac{17\, A\, I_{EE}\, R_C\, {V_{in}}^7}{40320\, {V_T}^7} \tag{20}$$

Die Graphen der Gleichung 16 und der Gleichung 20 sind in Abbildung 9 mit den Parametern $V_T = 26\ mV, R_C = 1\ k\Omega,\ I_{EE} = 10\ mA$ und $A = 1$ dargestellt.

Die Genauigkeit der Frequenzanalyse ist abhängig von der Übereinstimmung der Potenzreihenentwicklung mit der Transferfunktion. Liegt einer der Punkte der harmonischen Verzerrung zweiter und dritter Ordnung oder der 1-Dezibel-Kompressionspunkt im Bereich, in dem die Potenzreihe nicht mit der Transferfunktion übereinstimmt, ist das Ergebnis falsch. Die Genauigkeit kann durch Erhöhung des Grades der Taylorreihenentwicklung verbessert werden. Da aber die Potenzreihe hohen Grades an den Randbereichen sehr steil wird, kann man den Konvergenzradius nur unwesentlich steigern. In der Abbildung 9 ist leicht zu erkennen, in welchem Bereich die Potenzreihe mit der Transferfunktion übereinstimmt.

2.2 Harmonische Verzerrung zweiter Ordnung HD2

Nun wird zur Bestimmung der harmonischen Verzerrung zweiter und dritter Ordnung sowie des 1-Dezibel-Kompressionspunktes ein sinusförmiges Eingangssignal $V_{in} = \hat{V}_{in} sin(\omega t)$ eingegeben.

$$\hat{V}_{out,fkt,ser} = A\, I_{EE}\, R_C\, (\\ -\frac{\hat{V}_{in}\, \sin(\omega\, t)}{2\, V_T} + \frac{\hat{V}_{in}^3\, \sin(\omega\, t)^3}{24\, {V_T}^3} \\ -\frac{\hat{V}_{in}^5\, \sin(\omega\, t)^5}{240\, {V_T}^5} + \frac{17\, \hat{V}_{in}^7\, \sin(\omega\, t)^7}{40320\, {V_T}^7}) \tag{21}$$

Jetzt wird diese Reihe trigonometrisch umgeformt. Dies geschieht mit dem Mathematica-Kommando:

$$\hat{V}_{out,IST} = \\ \sum_{n=1}^{max} \text{SeriesCoefficient}[V_{out,fkt,ser}, n] \\ * \frac{\hat{V}_{in}^n * \text{TrigReduce}[V_{in,fkt}^n]}{V_{in,fkt}^n}, \tag{22}$$

wobei $V_{out,fkt,ser}$ Formel (21) und $V_{in,fkt} = \hat{V}_{in}\sin(\omega t)$ ist. Das Ergebnis lautet:

$$\hat{V}_{out,ist} = \frac{-\left(A\,I_{EE}\,R_C\,\hat{V}_{in}\sin(\omega t)\right)}{2\,V_T} + \frac{A\,I_{EE}\,R_C\left(3\,\hat{V}_{in}^3\sin(\omega t) - \hat{V}_{in}^3\sin(3\omega t)\right)}{96\,V_T{}^3} - \frac{A\,I_{EE}\,R_C\left(10\,\hat{V}_{in}^5\sin(\omega t) - 5\,\hat{V}_{in}^5\sin(3\omega t) + \hat{V}_{in}^5\sin(5\omega t)\right)}{3840\,V_T{}^5} + \frac{17\,A\,I_{EE}\,R_C}{2580480\,V_T{}^7}\Big(35\,\hat{V}_{in}^7\sin(\omega t) - 21\,\hat{V}_{in}^7\sin(3\omega t) + 7\,\hat{V}_{in}^7\sin(5\omega t) - \hat{V}_{in}^7\sin(7\omega t)\Big)$$

Werden aus dieser Formel die Vorfaktoren der $sin(\omega\,t)$-Terme aufsummiert, erhält man als Ergebnis die Amplitude der realen Grundwelle:

$$a_1 = -\frac{A\,I_{EE}\,R_C\,\hat{V}_{in}}{2\,V_T} + \frac{A\,I_{EE}\,R_C\,\hat{V}_{in}^3}{32\,V_T{}^3} - \frac{A\,I_{EE}\,R_C\,\hat{V}_{in}^5}{384\,V_T{}^5} + \frac{17\,A\,I_{EE}\,R_C\,\hat{V}_{in}^7}{73728\,V_T{}^7} \tag{23}$$

Dementsprechend ist die Amplitude der zweiten Harmonischen

$$a_2 = 0 \tag{24}$$

Bei der angegebenen Schaltung ergibt sich also eine harmonische Verzerrung zweiter Ordnung von 0. Dies ist damit zu begründen, daß bei symmetrischen Kennlinien keine Verzerrung zweiter Ordnung auftritt.

2.3 Harmonische Verzerrung dritter Ordnung HD3

Die Berechnung der harmonischen Verzerrung dritter Ordnung ist so wie die Berechnung der harmonischen Verzerrung zweiter Ordnung, wobei beim letzten Schritt der Berechnung der Frequenzanalyse an Stelle der zweiten Oberwelle die dritte Oberwelle verwendet wird. Die Funktion für die Amplitude der dritten Harmonischen lautet somit:

$$a_3 = -\frac{A\,I_{EE}\,R_C\,\hat{V}_{in}^3}{96\,V_T{}^3} + \frac{A\,I_{EE}\,R_C\,\hat{V}_{in}^5}{768\,V_T{}^5} + \frac{-17\,A\,I_{EE}\,R_C\,\hat{V}_{in}^7}{122880\,V_T{}^7} \tag{25}$$

Teilt man a_3 durch die Amplitude der Grundwelle a_1, so ergibt sich für die dritte harmonische Verzerrung:

$$\boxed{HD3 = \frac{3840\,\hat{V}_{in}^2\,V_T{}^4 - 480\,\hat{V}_{in}^4\,V_T{}^2 + 51\,\hat{V}_{in}^6}{184320\,V_T{}^6 - 11520\,\hat{V}_{in}^2\,V_T{}^4 + 960\,\hat{V}_{in}^4\,V_T{}^2 - 85\,\hat{V}_{in}^6}}$$

Um den harmonischen Schnittpunkt dritter Ordnung zu berechnen, werden nur die dominanten Terme $a_1 = -\frac{A\,I_{EE}\,R_C\,\hat{V}_{in}}{2\,V_T}$ und $a_3 = -\frac{A\,I_{EE}\,R_C\,\hat{V}_{in}^3}{96\,V_T{}^3}$ benutzt. Anschließend wird das Verhältnis $\frac{a_3}{a_1} = \frac{\hat{V}_{in}^2}{48\,V_T^2}$ gleich 1 gesetzt und nach $\hat{V}_{in}$ aufgelöst. Das Ergebnis für den harmonischen Schnittpunkt dritter Ordnung ist:

$$\boxed{HIP3 = 4\sqrt{3}\;V_T} \tag{26}$$

Bei der angegebenen Schaltung werden die Zahlenwerte eingesetzt $A = 1$; $I_{EE} = 1$ mA; $R_C = 1\ k\Omega$; $T = 27\ °C$ eingesetzt und die Ergebnisse graphisch auftragen (vergleiche Abbildung 10). Es ist im doppelt logarithmischen Maßstab zu sehen, wie die Amplituden der Grundwelle und der dritten Harmonischen verlaufen. Die Geraden stellen die Verläufe der dominanten Terme dar. Der Schnittpunkt der Geraden ist der Harmonische Schnittpunkt dritter Ordnung. Bei der Berechnung ergibt sich ein harmonischer Schnittpunkt dritter Ordnung bei 179 mV. Dies entspricht einer Leistung von -4,9 dBm. Die darunterliegenden Kennlinien stellen den Verlauf der realen Amplituden mit Kompression dar. Der Abstand der Kennlinien ist die harmonische Verzerrung dritter Ordnung.

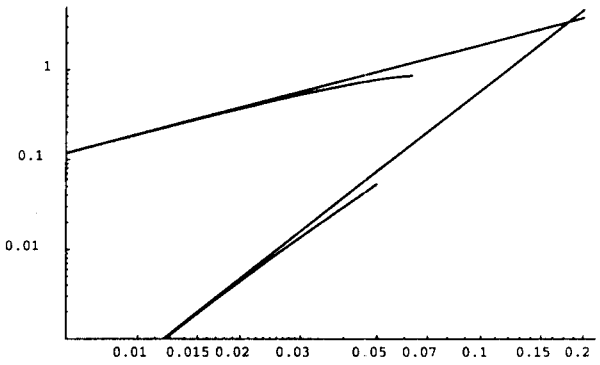


Abbildung 10: Harmonische Verzerrung und harmonischer Schnittpunkt dritter Ordnung

2.4 1-Dezibel-Kompressionspunkt P1dB

Kleinsignalverstärkung A_{dm}

Die Kleinsignalverstärkung A_{dm} im Arbeitspunkt ist die Steigung der Transferkennlinie. Sie ist eine Funk-

tion der Arbeitspunktspannung $V_{in,ap}$.

$$A_{dm} = \frac{\delta}{\delta V_{in}}(A * I_{EE} * R_C * \tanh(\frac{-V_{in}}{2 * V_T})) \quad (27)$$

Das Mathematica Kommando lautet hierzu:

$$D[V_{out}, V_{in}] \quad (28)$$

und ergibt:

$$A_{dm} = \frac{-A * I_{EE} * R_C * \mathrm{sech}(\frac{V_{in}}{2*V_T})^2}{2 * V_T} \quad (29)$$

Die Graphik für $V_T = 26*10^{-3}$, $R_C = 1*10^3$, $I_{EE} = 10*10^{-3}$ und $A = 1$ ist in Abbildung 11 dargestellt.

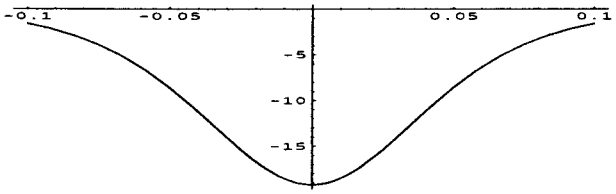


Abbildung 11: Kleinsignalverstärkung als Funktion der Arbeitspunktspannung V_{id}

Der numerische Wert für $V_{in} = 0$ ist:

$$A_{dm} = \frac{-A * I_{EE} * R_C}{2 * V_T} \approx -19.2308 \quad (30)$$

Die ideale Amplitude $\hat{V}_{out,ideal}$ der Grundwelle ist:

$$\begin{aligned} \hat{V}_{out,ideal} &= A_{dm} * \hat{V}_{in} \\ \hat{V}_{out,ideal} &= -\frac{A * I_{EE} * R_C * \hat{V}_{in}}{2 * V_T} \end{aligned} \quad (31)$$

Sie entspricht dem dominanten Term von a_1 (Gleichung 23).

Die reale Amplitude ist durch den gesamten Ausdruck von a_1 (Gleichung 23) gegeben:

$$\hat{V}_{out,real} = a_1 \quad (32)$$

Kompressionsfaktor k

Der Kompressionsfaktor k setzt sich aus dem Bruch des realen Wertes durch den idealen Wert zusammen.

$$k = \frac{\hat{V}_{out,real}}{\hat{V}_{out,ideal}} \quad (33)$$

Mathematica gibt somit folgende Formel aus:

$$k = 1 - \frac{\hat{V}_{in}^2}{16\,{V_T}^2} + \frac{\hat{V}_{in}^4}{192\,{V_T}^4} - \frac{17\,\hat{V}_{in}^6}{36864\,{V_T}^6} \quad (34)$$

Dies ist eine Funktion in Abhängigkeit von $\hat{V}_{in}$

1-Dezibel-Kompressionspunkt

Der Kompressionsfaktor wird bei 1 Dezibel berechnet. Hieraus ergibt sich ein Kompressionsfaktor von:

$$k = 10^{\frac{-dB}{20}} = 10^{\frac{-1}{20}} \approx 0,891251 \quad (35)$$

Anschließend wird dieser mit folgendem Mathematica-Kommando nach $\hat{V}_{in}$ aufgelöst:

$$\hat{V}_{in} = \mathrm{Solve}[10^{\frac{-1}{20}} == \frac{\hat{V}_{out,real}}{\hat{V}_{out,ideal}}, \hat{V}_{in}]; \quad (36)$$

Die symbolische Lösung sieht wie folgt aus:

$$\begin{aligned} \hat{V}_{in} = 2\sqrt{\frac{2}{85}}\sqrt{(40V_T^2 + 2^{\frac{13}{20}}5^{\frac{59}{60}}(-2601 + 2117\ 10^{\frac{1}{20}} + \\ 51\sqrt{2601 - 4234\ 10^{\frac{1}{20}} + 1789\ 10^{\frac{1}{10}}})^{\frac{1}{3}}(V_T^6)^{\frac{1}{3}} - \\ \left(350\ 2^{\frac{7}{20}}5^{\frac{1}{60}}(V_T^6)^{\frac{2}{3}}\right)/((-2601 + 2117\ 10^{\frac{1}{20}} + \\ 51\sqrt{2601 - 4234\ 10^{\frac{1}{20}} + 1789\ 10^{\frac{1}{10}}})^{\frac{1}{3}}V_T^2)) \end{aligned}$$

Es ist zu sehen, daß der Kompressionspunkt des Differenzverstärkers nur von der Temperatur abhängt.

Zur graphischen Darstellung werden wieder die Werte $A = 1$; $I_{EE} = 1$ mA; $R_C = 1$ $k\Omega$; $T = 27$ $^\circ C$ eingesetzt (vergleiche Abbildung 12). Es ist im doppelt logarithmischen Maßstab zu sehen, wie die ideale und reale Amplitude der Grundwelle verläuft. Bei dieser Berechnung ergibt sich ein 1-Dezibel-Kompressionspunkt bei 36,9 mV. Dies entspricht einer Leistung von -18,6 dBm.

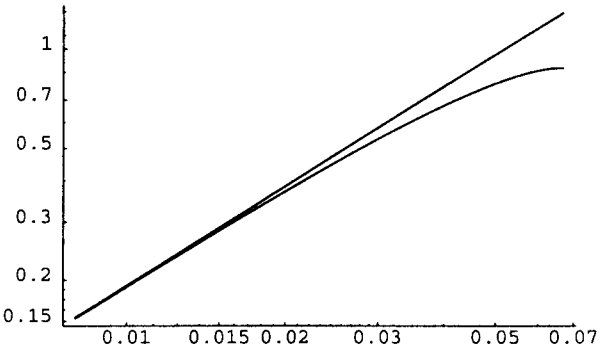


Abbildung 12: Graphik des 1-Dezibel-Kompressionspunktes

Abbildung 13 zeigt die Temperaturabhängigkeit des Kompressionspunktes in dBm.

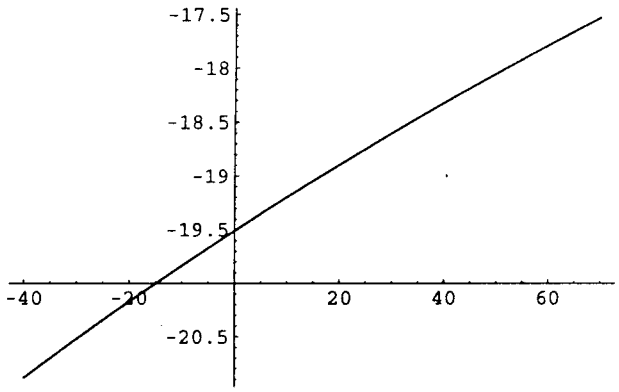


Abbildung 13: Kompressionspunkt in dBm aufgetragen über der Temperatur in Grad Celsius

2.5 Intermodulationsschnittpunkt zweiter Ordnung (auf den Eingang bezogen) IIP2

Das Intermodulationsverhältnis IM2 ist eine Funktion von $\hat{V}_{in}$. In die Übertragungsfunktion wird nun die Überlagerung $V_{in} = \hat{V}_{in}\ (\sin(\omega_1 t) + \sin(\omega_2 t))$ eingegeben. Dies ergibt die Funktion:

$$V_{out} = -A I_{EE} R_C \tanh\left(\frac{\hat{V}_{in}\,(\sin(\omega_1 t) + \sin(\omega_2 t))}{2 V_T}\right)$$

Zuerst wird für die Frequenzanalyse die Funktion wieder durch eine Taylorreihe ersetzt:

$$V_{od,ser} = \text{Series}\ [\ V_{od},$$
$$\{\ \hat{V}_{in},\ Entwicklungsstelle = 0\ , AnzahlTerme\}\ ].$$

Man erhält:

$$V_{od,ser} = -\frac{A\,I_{EE}\,R_C\,(\sin(\omega_1 t) + \sin(\omega_2 t))\,\hat{V}_{in}}{2\,V_T} + \frac{A\,I_{EE}\,R_C\,(\sin(\omega_1 t) + \sin(\omega_2 t))^3\,\hat{V}_{in}^3}{24\,{V_T}^3} - \frac{A\,I_{EE}\,R_C\,(\sin(\omega_1 t) + \sin(\omega_2 t))^5\,\hat{V}_{in}^5}{240\,{V_T}^5} + \frac{17\,A\,I_{EE}\,R_C\,(\sin(\omega_1 t) + \sin(\omega_2 t))^7\,\hat{V}_{in}^7}{40320\,{V_T}^7}$$

Anschließend werden die Sinusterme mit Mathematica trigonometrisch umgeformt:

$$\hat{V}_{out,ist} = \sum_{n=1}^{max} \text{SeriesCoefficient}[V_{out,fkt,ser}, n] * \frac{\hat{V}_{in}^n * \text{TrigReduce}[V_{in,fkt}^n]}{V_{in,fkt}^n}$$

Das Ergebnis lautet:

$$\text{Out[14]} = -\frac{A\,I_{EE}\,R_C\,(\sin(t\omega_1)\,\hat{V}_{in} + \sin(t\omega_2)\,\hat{V}_{in})}{2\,V_T} + \frac{1}{96\,V_T^3}(A\,I_{EE}\,R_C\,(9\sin(t\omega_1)\,\hat{V}_{in}^3 - \sin(3t\omega_1)\,\hat{V}_{in}^3 + 9\sin(t\omega_2)\,\hat{V}_{in}^3 - \sin(3t\omega_2)\,\hat{V}_{in}^3 - 3\sin(t\omega_1 - 2t\omega_2)\,\hat{V}_{in}^3 + 3\sin(2t\omega_1 - t\omega_2)\,\hat{V}_{in}^3 - 3\sin(2t\omega_1 + t\omega_2)\,\hat{V}_{in}^3 + 3\sin(t\omega_1 + 2t\omega_2)\,\hat{V}_{in}^3)) - \frac{1}{3840\,V_T^5}(A\,I_{EE}\,R_C\,(100\sin(t\omega_1)\,\hat{V}_{in}^5 - 25\sin(3t\omega_1)\,\hat{V}_{in}^5 + \sin(5t\omega_1)\,\hat{V}_{in}^5 + 100\sin(t\omega_2)\,\hat{V}_{in}^5 - 25\sin(3t\omega_2)\,\hat{V}_{in}^5 + \sin(5t\omega_2)\,\hat{V}_{in}^5 - 5\sin(t\omega_1 - 4t\omega_2)\,\hat{V}_{in}^5 - 10\sin(2t\omega_1 - 3t\omega_2)\,\hat{V}_{in}^5 - 50\sin(t\omega_1 - 2t\omega_2)\,\hat{V}_{in}^5 + 10\sin(3t\omega_1 - 2t\omega_2)\,\hat{V}_{in}^5 + 50\sin(2t\omega_1 - t\omega_2)\,\hat{V}_{in}^5 - 5\sin(4t\omega_1 - t\omega_2)\,\hat{V}_{in}^5 - 50\sin(2t\omega_1 + t\omega_2)\,\hat{V}_{in}^5 + 5\sin(4t\omega_1 + t\omega_2)\,\hat{V}_{in}^5 - 50\sin(t\omega_1 + 2t\omega_2)\,\hat{V}_{in}^5 + 10\sin(3t\omega_1 + 2t\omega_2)\,\hat{V}_{in}^5 + 10\sin(2t\omega_1 + 3t\omega_2)\,\hat{V}_{in}^5 + 5\sin(t\omega_1 + 4t\omega_2)\,\hat{V}_{in}^5)) + \frac{1}{2580480\,V_T^7}(17\,A\,I_{EE}\,R_C\,(1225\sin(t\omega_1)\,\hat{V}_{in}^7 - 441\sin(3t\omega_1)\,\hat{V}_{in}^7 + 49\sin(5t\omega_1)\,\hat{V}_{in}^7 - \sin(7t\omega_1)\,\hat{V}_{in}^7 + 1225\sin(t\omega_2)\,\hat{V}_{in}^7 - 441\sin(3t\omega_2)\,\hat{V}_{in}^7 + 49\sin(5t\omega_2)\,\hat{V}_{in}^7 - \sin(7t\omega_2)\,\hat{V}_{in}^7 - 7\sin(t\omega_1 - 6t\omega_2)\,\hat{V}_{in}^7 + 21\sin(3t\omega_1 - 5t\omega_2)\,\hat{V}_{in}^7 - 147\sin(5t\omega_1 - 4t\omega_2)\,\hat{V}_{in}^7 + 35\sin(7t\omega_1 - 4t\omega_2)\,\hat{V}_{in}^7 - 245\sin(3t\omega_1 - 3t\omega_2)\,\hat{V}_{in}^7 + 35\sin(4t\omega_1 - 3t\omega_2)\,\hat{V}_{in}^7 - 735\sin(t\omega_1 - 2t\omega_2)\,\hat{V}_{in}^7 + 245\sin(3t\omega_1 - 2t\omega_2)\,\hat{V}_{in}^7 - 21\sin(5t\omega_1 - 2t\omega_2)\,\hat{V}_{in}^7 + 735\sin(2t\omega_1 - t\omega_2)\,\hat{V}_{in}^7 - 147\sin(4t\omega_1 - t\omega_2)\,\hat{V}_{in}^7 + 7\sin(6t\omega_1 - t\omega_2)\,\hat{V}_{in}^7 - 735\sin(2t\omega_1 + t\omega_2)\,\hat{V}_{in}^7 + 147\sin(4t\omega_1 + t\omega_2)\,\hat{V}_{in}^7 - 7\sin(6t\omega_1 + t\omega_2)\,\hat{V}_{in}^7 + 735\sin(t\omega_1 + 2t\omega_2)\,\hat{V}_{in}^7 - 245\sin(3t\omega_1 + 2t\omega_2)\,\hat{V}_{in}^7 + 21\sin(5t\omega_1 + 2t\omega_2)\,\hat{V}_{in}^7 + 245\sin(2t\omega_1 + 3t\omega_2)\,\hat{V}_{in}^7 - 35\sin(4t\omega_1 + 3t\omega_2)\,\hat{V}_{in}^7 + 147\sin(t\omega_1 + 4t\omega_2)\,\hat{V}_{in}^7 - 35\sin(3t\omega_1 + 4t\omega_2)\,\hat{V}_{in}^7 + 21\sin(2t\omega_1 + 5t\omega_2)\,\hat{V}_{in}^7 - 7\sin(t\omega_1 + 6t\omega_2)\,\hat{V}_{in}^7))$$

Für die Amplituden der Grundwellen ω_1 oder ω_2 ist das Ergebnis:

$$b_1 = -\frac{A\,I_{EE}\,R_C\,\hat{V}_{in}}{2\,V_T} + \frac{3\,A\,I_{EE}\,R_C\,\hat{V}_{in}^3}{32\,{V_T}^3} - \frac{5\,A\,I_{EE}\,R_C\,\hat{V}_{in}^5}{192\,{V_T}^5} + \frac{595\,A\,I_{EE}\,R_C\,\hat{V}_{in}^7}{73728\,{V_T}^7} \qquad (37)$$

Für die Amplituden der Mischterme bei den Frequenzen $\omega_1 + \omega_2$ oder $\omega_2 - \omega_1$ ist das Ergebnis:

$$b_2 = 0 \qquad (38)$$

Nun wird noch das Verhältnis gebildet zwischen der Amplitude bei der Frequenz $\omega_1 + \omega_2$ oder $\omega_2 - \omega_1$ und der Amplitude der Grundwelle ω_1 oder ω_2. Das Intermodulationsverhältnis IM2=$\frac{b_2}{b_1}$ wird also zu Null, weshalb auch kein Intermodulationsschnittpunkt existiert. Der Grund liegt wieder in der symmetrischen Transferkennlinie.

2.6 Intermodulationsschnittpunkt dritter Ordnung (auf den Eingang bezogen) IIP3

Der Ablauf der Frequenzanalyse und die Berechnung der Grundwelle entsprechen genau dem Intermodulationsschnittpunkt zweiter Ordnung. Für die Amplitude bei der Frequenz $2\omega_1 - \omega_2$ oder $2\omega_2 - \omega_1$ erhalten

wir:

$$b_3 = -\frac{A\,I_{EE}\,R_C\,\hat{V}_{in}^3}{32\,{V_T}^3} + \frac{5\,A\,I_{EE}\,R_C\,\hat{V}_{in}^5}{384\,{V_T}^5} - \frac{119\,A\,I_{EE}\,R_C\,\hat{V}_{in}^7}{24576\,{V_T}^7} \qquad (39)$$

Als Ergebnis für das Intermodulationsverhältnis dritter Ordnung erhält man:

$$\mathrm{IM3} = \frac{b_3}{b_1} = \frac{2304\,\hat{V}_{in}^2\,{V_T}^4 - 960\,\hat{V}_{in}^4\,{V_T}^2 + 357\,\hat{V}_{in}^6}{36864\,{V_T}^6 - 6912\,\hat{V}_{in}^2\,{V_T}^4 + 1920\,\hat{V}_{in}^4\,{V_T}^2 - 595\,\hat{V}_{in}^6} \qquad (40)$$

Intermodulationsschnittpunkt dritter Ordnung

Um den Intermodulationsschnittpunkt dritter Ordnung zu berechnen, werden nur die dominanten Terme $b_1 = -\frac{A\,I_{EE}\,R_C\,\hat{V}_{in}}{2\,V_T}$ und $b_3 = -\frac{A\,I_{EE}\,R_C\,\hat{V}_{in}^3}{32\,{V_T}^3}$ benutzt. Anschließend wird das Verhältnis $\frac{b_3}{b_1} = \frac{\hat{V}_{in}^2}{16 V_T^2}$ gleich 1 gesetzt und nach $\hat{V}_{in}$ aufgelöst. Es ergibt sich das Ergebnis mit folgendem Kommando:

$$\mathrm{IIP3}_{lsg} = \mathrm{Solve}[\frac{\hat{V}_{in}^2}{16 V_T^2} == 1, \hat{V}_{in}]; \qquad (41)$$

Das Ergebnis für den Intermodulationsschnittpunkt dritter Ordnung ist:

$$\boxed{\mathrm{IIP3} = 4\ V_T} \qquad (42)$$

Der Intermodulationsschnittpunkt dritter Ordnung des hier verwendeten Differenzverstärkers hängt nur von der Temperatur ab.

Mit $A = 1$; $I_{EE} = 1$ mA; $R_C = 1\ k\Omega$; $T = 27\ °C$ werden die Ergebnisse in Abbildung 14 graphisch dargestellt. Im doppelt logarithmischen Maßstab sind die Amplituden der Grundwelle und der Intermodulationsprodukte dritter Ordnung bei der Frequenz $2\omega_1 - \omega_2$ oder $2\omega_2 - \omega_2$ aufgetragen. Die Geraden stellen die Verläufe der dominanten Terme dar. Der Schnittpunkt der Geraden ist der Intermodulationsschnittpunkt dritter Ordnung. Rechnerisch ergibt sich ein Intermodulationsschnittpunkt dritter Ordnung (auf den Eingang bezogen) von 103,5 mV. Dies entspricht einer Leistung von -9,7 dBm. Die darunterliegenden Kennlinien stellen den Verlauf der realen Amplituden mit Kompression dar. Der Abstand dieser Kennlinien ist der Intermodulationsabstand dritter Ordnung.

Abbildung 15 zeigt die Temperaturabhängigkeit des Intermodulationsschnittpunkts dritter Ordnung.

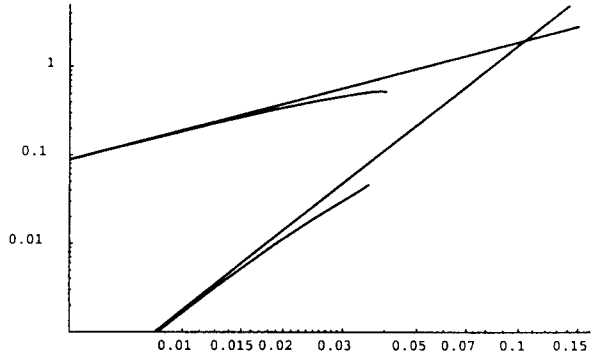


Abbildung 14: Intermodulationsabstand und Intermodulationsschnittpunkt dritter Ordnung

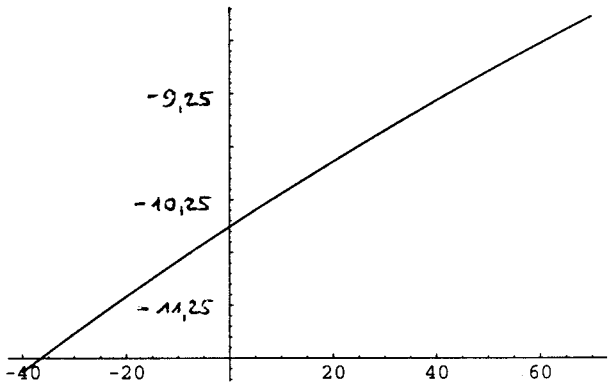


Abbildung 15: IIP3 in dBm aufgetragen über der Temperatur in Grad Celsius

3 Resumee

Mit dieser Arbeit wurde gezeigt, daß es mit Hilfe des Computeralgebrasystems Mathematica möglich ist, Großsignalparameter für Verstärker auf automatischem Wege symbolisch zu bestimmen. Sollten Bauteile in die Gleichungen eingehen, so könnte nach diesen aufgelöst werden, wodurch eine unmittelbare Dimensionierung dieser Bauteile möglich wäre. Voraussetzung hierfür ist, daß die Übertragungskennlinie in symbolischer Form darstellbar ist. Mit Bipolar-Transistoren stößt man jedoch in der Regel auf transzendente Gleichungen, wenn Emitterwiderstände oder Bahnwiderstände berücksichtigt werden sollen. Außerdem müssen für Anwendungen im HF-Bereich auch die dynamischen Einflüsse auf das nichtlineare Verhalten berücksichtigt werden. Große Bedeutung wird daher zukünftig weitergehenden Approximationsverfahren wie die Padé- oder Volterra-Approximation zukommen.

Literatur

[1] G. Forster, R. Quell, A. Stürmer: *MEDEA SADE Milestone Report R 3.2.2*, TEMIC Semiconductor GmbH, Fachhochschule Ulm, Dec. 1998

[2] Meinke, Gundlach: *Taschenbuch der Hochfrequenztechnik*, Studienausgabe, Springer-Verlag (1996)

[3] Wolfram, Stephen: *Das Mathematica Buch*, Die offizielle Dokumentation, 3. Auflage Mathematica Version 3, Addison-Wesley

[4] V. Mutlu, G. Forster: *Dimensionierung einer Differenzverstärker-Stufe mit ANALOG INSYDES und MATHEMATICA*, Fachhochschule Ulm, Prittwitzstraße 10, 89075 Ulm

[5] P. R. Gray, R. G. Meyer: *Analysis and Design of Analog Integrated Circuits*, Wiley, New York 1993

Diese Arbeit wird gefördert vom Bundesministerium für Bildung und Forschung (Förderkennzeichen 01M3037C). Die Autoren sind für den Inhalt verantwortlich.

Mikrosystemtechnik an der FH-Furtwangen

Prof. Dr. Ulrich Mescheder
Geschäftsführender Leiter Institut für Angewandte Forschung
Fachbereich Mechatronik und Mikrosysteme

Abstract

Die Arbeiten auf dem Gebiet der Mikrosystemtechnik an der FH Furtwangen beziehen sich auf die Bereiche Lehre und Forschung (Folie 1 und 2).

Lehre
Bereits seit 1991 gibt es einen Studiengang Mikrosystemtechnik an der FH Furtwangen. Dieser neuartige Studiengang wurde aus Aktivitäten zum Thema Mikroelektronik entwickelt. Schwerpunkte dieses Studiengangs Mikrosystemtechnik sind neben den allgemeinen elektrotechnischen Grundlagen der Entwurf digitaler und analoger Schaltungen, mikroelektronische Bauelemente und Technologie der hochintegrierter Schaltungen und der Mikromechanik. Im Herbst 1999 wird auch ein Masterprogramm „Mikrosystemtechnik" mit einer internationalen Ausrichtung gestartet (Folie 3).

Forschung
Seit über zehn Jahren werden an der FH Furtwangen Forschungsarbeiten im Bereich Mikrosystemtechnik durchgeführt. Die Mikrosystemtechnik ist ein starker Schwerpunkt im Institut für Angewandte Forschung der FH Furtwangen (Folien 4-7). Die Forschungsarbeiten profitieren dabei insbesondere von dem hervorragend ausgestatteten Mikroelektronik-Laboratorium, das über Fertigungsmöglichkeiten für einfache mikroelektronischen Bauelemente und vor allem für mikromechanische Bauelemente verfügt (Folie 8, 9). Dementsprechend beschäftigen sich viele Projekte mit der Entwicklung von mikromechanischen Sensoren (Folie 10). Beispiele für an der FHF entwickelte Sensoren sind:

miniaturisierte Mikrofone
mikromechanische Zwei-Achsen-Neigungssensoren (Folie 11-13)

Neben der Sensorentwicklung ist die Verbesserung der Herstellungstechnologie für Mikrosysteme ein zentrales Forschungsthema.
Ein Beispiel für Entwicklungen in diesem Bereich ist die optische Prozeßkontrolle. Es wurde ein Verfahren entwickelt, das die optische Messung des dünngeätzten Siliziums während des naßchemischen Ätzens gestattet. Auf diese Weise können auch relativ dicke Membranen (>10 µm) reproduzierbar hergestellt werden (Folie 14 und 15). Durch Licht läßt sich aber auch die Ätzrate beim anisotropen Ätzen von Silizium mit KOH steuern, was neue Fertigungsmöglichkeiten erschließt (Folie 16 u. 17).
In einem weiteren Projekt wird die Anwendung von Lasern in der Mikrotechnik untersucht. Es wurde eine Laserlithografieanlage mit einem kurzwelligen Argon-Laser aufgebaut. Mit einem Nd-Yag-Laser wurden Untersuchungen zum Trimmen von mikromechanischen Sensoren und zur Verbindung mit verschiedenen Substraten durchgeführt (Folie 18).

Ein Beispiel für eine Entwicklung im Bereich der Aktorik ist der Piezoschrittmotor (Folie 19). Mit diesem Motor, der Schritt- und Linearbetrieb in einem Funktionsblock enthält, können große Verfahrwege (>100 mm) mit einer Auflösung von 4 nm (Meßauflösung) realisiert werden. Das Kernstück dieses Motors ist ein Antriebsblock, der alle notwendigen Elemente in

einem monolithischen Block enthält. Dieser Aufbau vereinfacht die Montage (Folie 20 u. 21). Die Wiederholgenauigkeit liegt bei großen Verfahrwegen um $\pm$ 200 nm (Folie 22), im Nahbereich bei etwa + 50 nm (Folie 23).

Viele der laufenden Drittmittelprojekte werden in Kooperation mit der Industrie durchgeführt.

Da es in den letzten Jahren auch in der Mikrosystemtechnik einen Trend zu Foundries gibt, auf die z.B. innerhalb von Europractice zugegriffen werden kann, gibt es durchaus Übereinstimmungen zu den Arbeiten des MPCFH-Verbundes. Insbesondere die Vereinheitlichung und Verbesserung von Entwurfstools ist in den nächsten Jahren eine wichtige Fragestellung innerhalb der Mikrosystemtechnik. Eine Zusammenarbeit mit dem MPCFH-Verbund ist daher sinnvoll.

In der Anlage sind einige ausgewählte Folien zum auf der MPCFH-Tagung gehaltenen Vortrag zu finden. Im einzelnen beziehen sich die Folien auf die Projekte „Mikromechanischer Zwei-Achsen-Neigungssensor", „Optische Prozeßkontrolle", „Laserbearbeitung in der Mikrotechnik" und „Piezoschrittmotor".

Mikrosystemtechnik an der FH Furtwangen

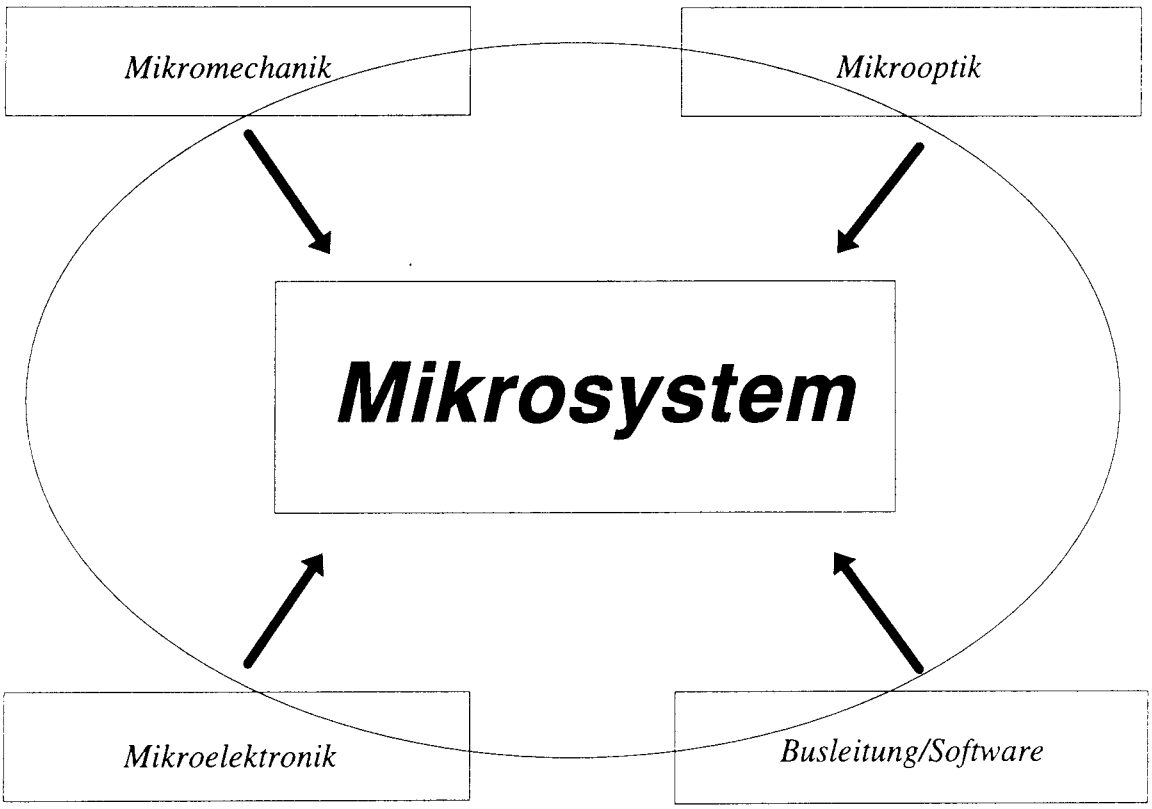
Prof. Dr. Ulrich Mescheder

Fachbereich Mechatronik und Mikrosysteme
Institut für Angewandte Forschung (IAF)

- **Studium der Mikrosystemtechnik an der FHF**
- **IAF-Forschungsschwerpunkte**
- **F+E-Projekte im Schwerpunkt Mikrosystemtechnik**

Begriffsdefinition Mikrosystemtechnik

- **funktionsbestimmende Strukturelemente in der Größenordnung Mikrometer**
- **Verknüpfung elektrischer und nichtelektrischer Funktionen (Sensoren, Aktoren, Elektronik)**
- **Herstellungsverfahren der IC-Technik**



MPCFH-22-1-99_4

Studium der Mikrosystemtechnik an der FHF

- **Beginn: 1991 (davor als Schwerpunkt Mikroelektronik)**

- **spezifische Themengebiete:**

  ⇒ Technologie (Si-Linie)

  ⇒ Entwurf (analog/digital)

  ⇒ Halbleitercharakterisierung

- **Weiterentwicklung**

  ⇒ Entwicklung von Bachelor- und Masterprogrammen (BLK-Programm mit Hochschule Bremen)

  ⇒ Start des Masterprogramms: WS 99/00

MPCFH-22-1-99_2

Leitung: Prof. Dr. U. Mescheder

Mikrosystemtechnik

- Sensoren
- Mikroelektronik
- Positioniersysteme

Prof. Dr. W. Kuntz
Prof. Dr. U. Mescheder
Prof. Dr. B. Richards
Prof. Dr. W. Rülling
Prof. Dr. A. Stoffel

Oberflächentechnik

- Hartstoffbeschichten
- Spritzgießen
- Hochgeschwindigkeits-schleifen
- Zahnimplantate

Prof. Dr. J. Ebberink
Prof. Dr. G. Haimerl
Prof. Dr. H. Schiefer
Prof. Dr. T. Tawakoli

Umweltsystemtechnik

- Umweltverfahrenstechnik
- Umweltbildung
- Umweltmanagement
- Umweltinformationssysteme

Prof. Dr. F. Bigge
Prof. Dr. H. Krinn
Prof. Dr. S. Mahling-Ennaoui
Prof. Dr. H. Meinholz
Prof. Dr. T. Oppenländer

Übersicht über die am IAF der FH Furtwangen durchgeführten Forschungsprojekte

IAF-Schwerpunkt	Diplom- und Studienarbeiten in Projekten	Veröffentlichungen in Fachliteratur	wissenschaftl. Vorträge	Messebeteiligungen, Ausstellungen	Patente
Mikrosystemtechnik	8	10	4	4	2
Oberflächentechnik	0	10	5	4	1
Umweltsystemtechnik	4	10	4	3	0
Summe	12	29	13	11	3

MPCFH-22-1-99_3

Drittmittel für IAF-Forschungsprojekte 1998

IAF-Schwerpunkt	Anzahl der Forschungsprojekte	akquirierte Drittmittel
Mikrosystem-technik	8	535.060 DM
Oberflächen-technik	11	90.000 DM
Umweltsystem-technik	5	440.000 DM
Summe	24	1.065.060 DM

Infrastruktur Mikrosystemtechnik

- Reinräume - MOS- und CMOS-Techniken

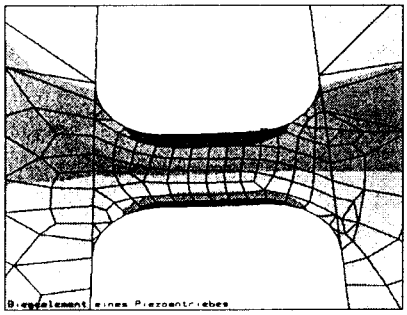
- Sondertechniken für Si-Mikromechanik
- moderne Entwurfs- und Simulationstools (CADENCE, MENTOR, Xilinx, SYNOPSYS u.a.)
- Meß- und Charakterisierungsgeräte

Mikroelektronik

- analog
- digital

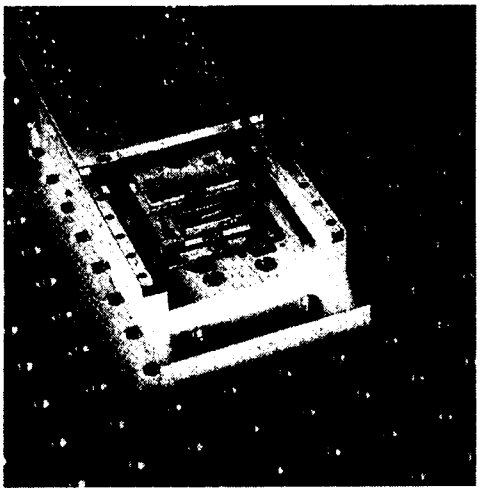
Entwurf und Simulation

- Design und Simulation elektronischer Schaltungen
- Finite-Elemente-Simulation (ANSYS)



Positioniersysteme

- hochpräzise Piezoschrittmotoren

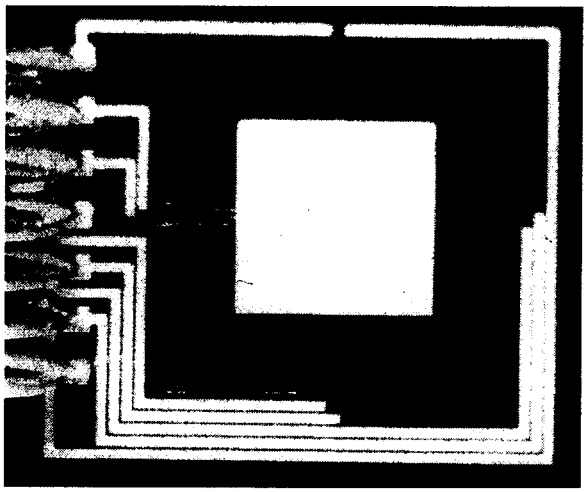


Zuverlässigkeit von Mikrosystemen

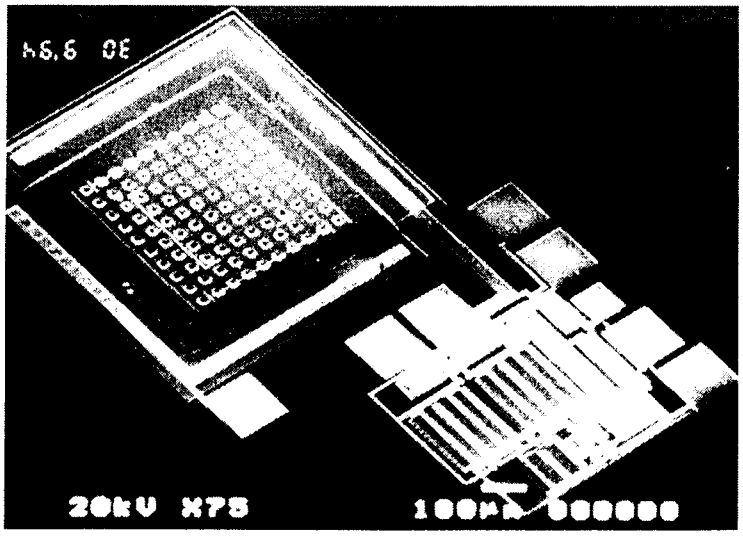
- Fehlertoleranter Entwurf von Mikrosystemen

mikromechanische Sensoren

- Neigung - Zwei-Achsen-Neigungssensor



- Bewegung
- Schall - mikromechanisches Si-Mikrofon



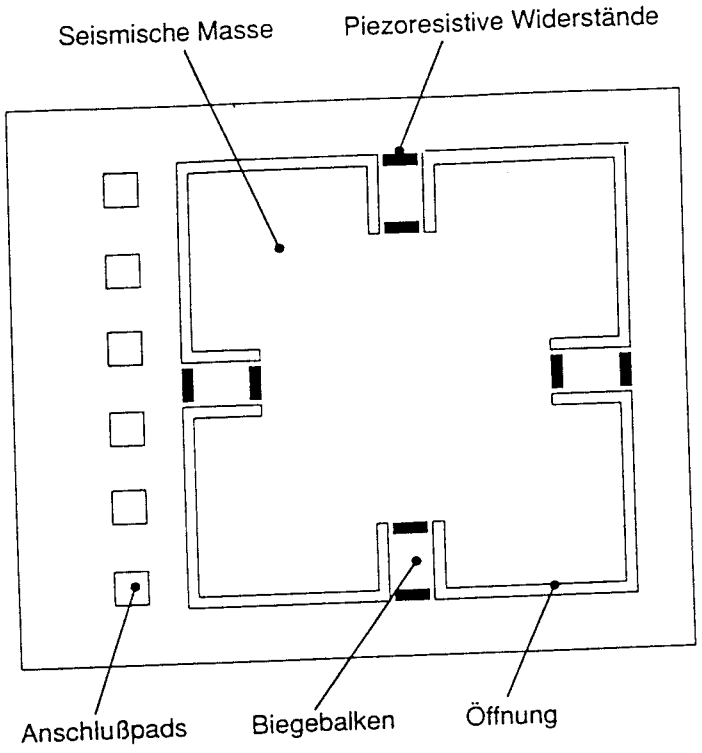
- Druck
- Feuchte

Mikromechanischer Zwei-Achsen-Neigungssensor

Anwendungsgebiete

- **Kfz**
  - ⇒ betriebsabhängige Steuerung in Automatikgetrieben
  - ⇒ Fahrwerksregelung
  - ⇒ Überschlagssensor
  - ⇒ Diebstahlsicherungssysteme
- **Medizintechnik**
  - ⇒ Herzschrittmacher
  - ⇒ computerunterstützte Operation
- **Consumerbereich**
  - ⇒ elektronische Wasserwaage
  - ⇒ elektronische Kompasse
- **Sonstige Anwendungen**
  - ⇒ handgehaltene optische Geräte
  - ⇒ Ersatz von Quecksilberschaltern

Feinlayout aus Simulation



- Gesamtfläche 5x5 mm2
- Balkenlänge ca. 500 µm
- Balkenbreite 70 µm
- Balkendicke 5 µm
- mechanische Endanschläge: 5 µm

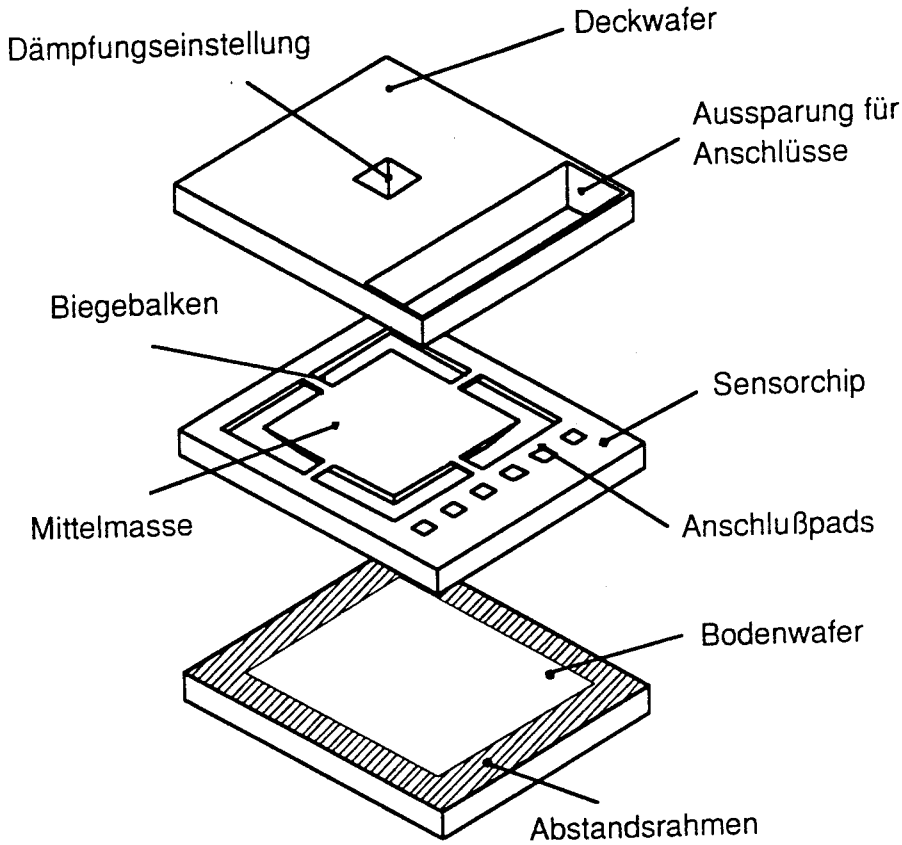
⟶ Schockfestigeit ca. 100g

bn1139610

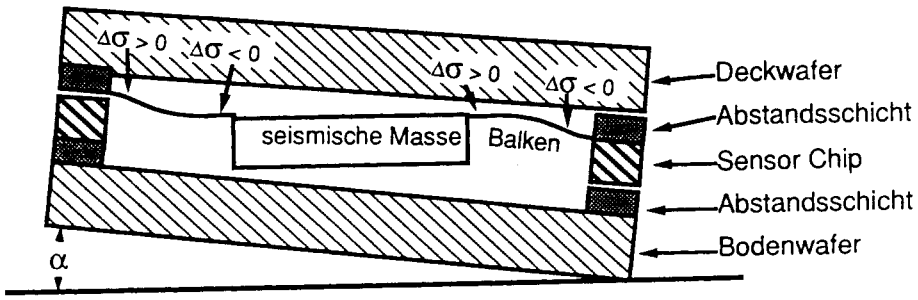
Institut für
Angewandte
Forschung

Hochschule für Technik
Furtwangen (Germany)
FB Mechatronik u. Mikrosysteme

Gesamtansicht des Sensors



Explosionszeichung des Gesamtsensors mit Boden- und Deck-Chip zum Schutz



Querschnitt durch Gesamtsensor (geneigt)

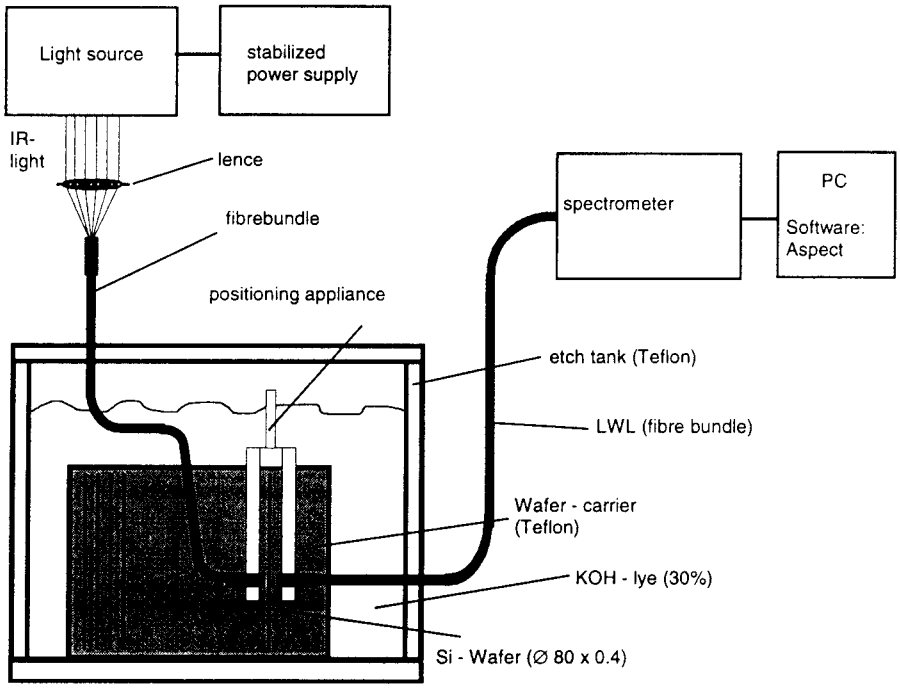
bn1139615

Institut für
Angewandte
Forschung

Hochschule für Technik
Furtwangen (Germany)
FB Mechatronik u. Mikrosysteme

Optical process control during anisotropic etching
Experimentell set-up
of c-Si

Experimental set-up

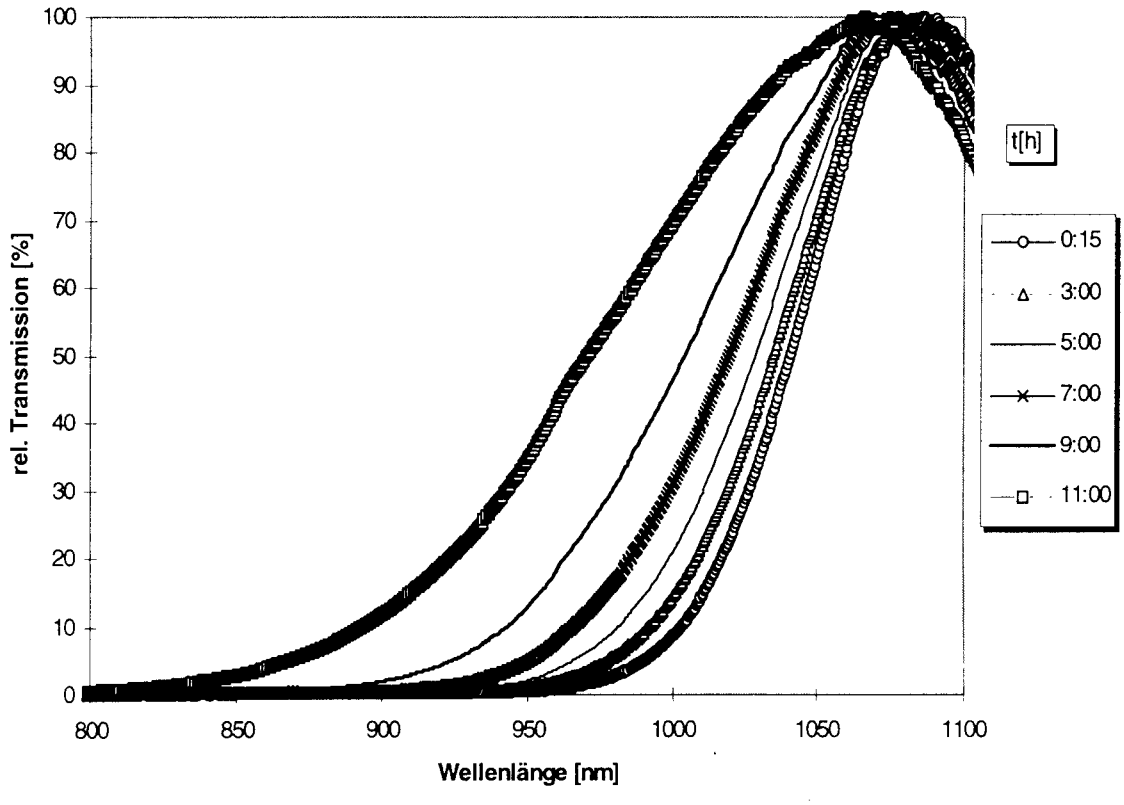


Process monitoring: algorithmn

$$d(t_1) = \frac{1}{\alpha_A - \alpha_B} \cdot \ln\left(\frac{T_B(t_1)}{T_A(t_1)}\right) = \frac{1}{\alpha_A - \alpha_B} \cdot \ln\left(\frac{I_{TB}(t_1)}{I_{TA}(t_1)}\right)$$

α_A , α_B variabel

$$\frac{d(t_1)}{d(t_0)} = \ln\left(\frac{I_{TB}(t_1)}{I_{TA}(t_1)}\right) / \ln\left(\frac{I_{TB}(t_0)}{I_{TA}(t_0)}\right) \mid \alpha_A , \alpha_B \text{ fixed}$$



MPCFH-22-1-99_9

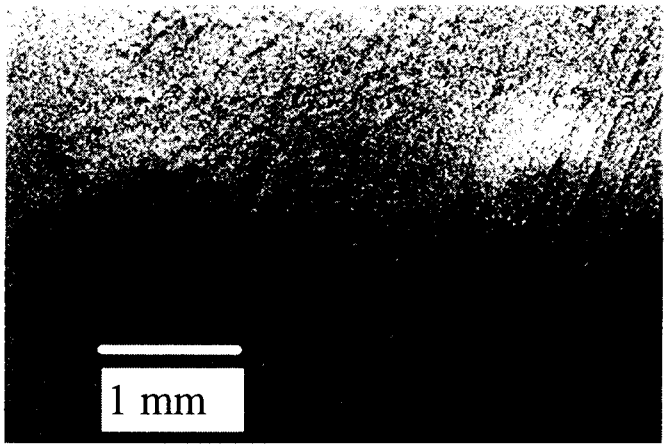
Control of etch rate during anisotropic etching of Si in KOH by light illumination

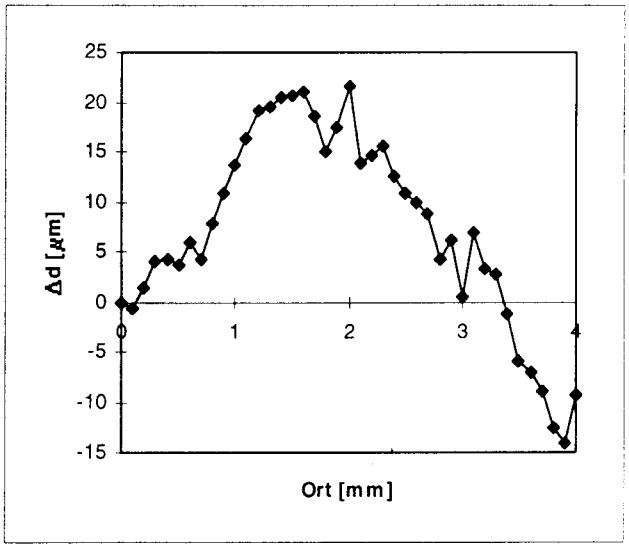
⇒ Reduction of etch rate (λ:700-900 nm)

⇒ etching of circular structures within an anisotropic process

⇒ compensation of konvex corners

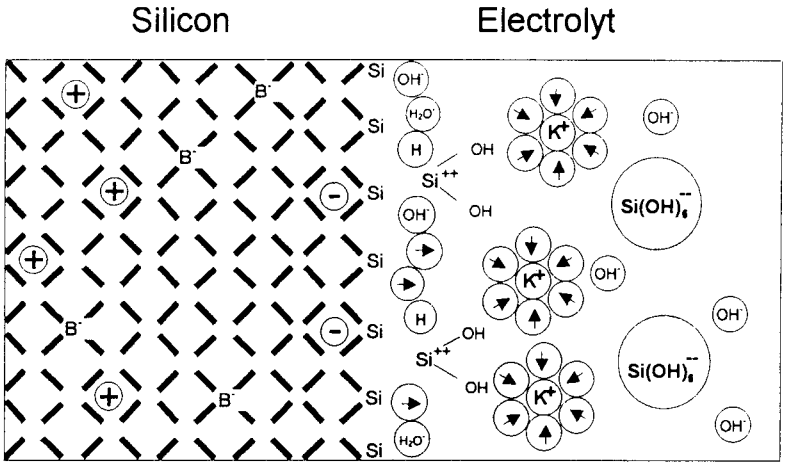
⇒ maskless structuring of simple features



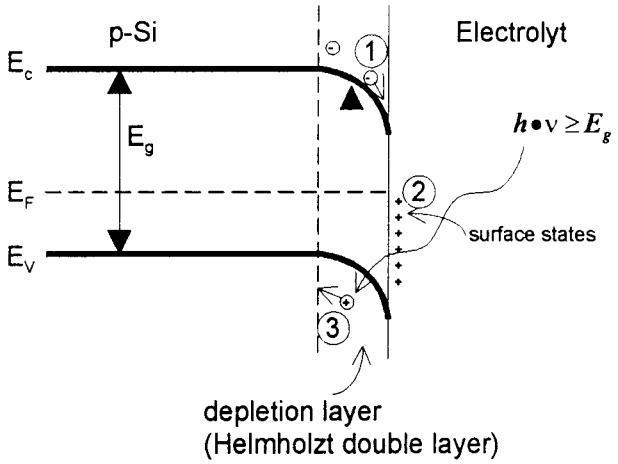


MPCFH-22-1-99_10

Mechanismn for light control in Si-etching process with KOH



Mechanismn



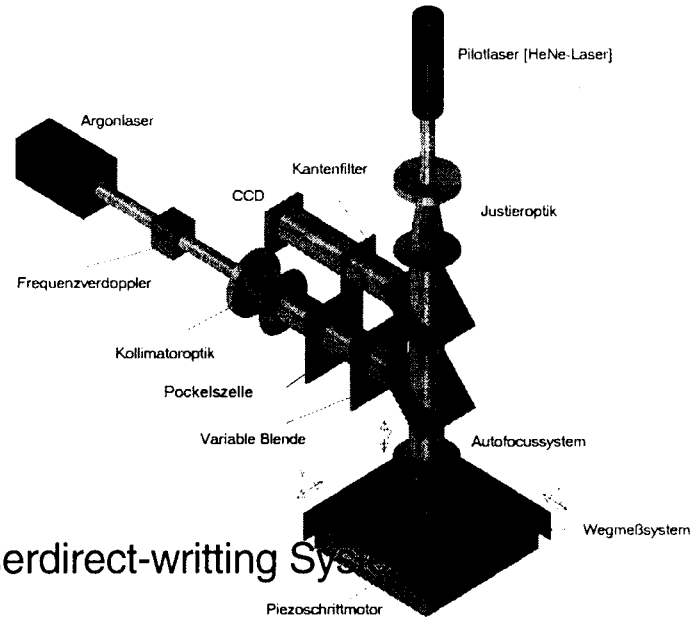
MPCFH-22-1-99_11

Laser-micromaching for micro-components

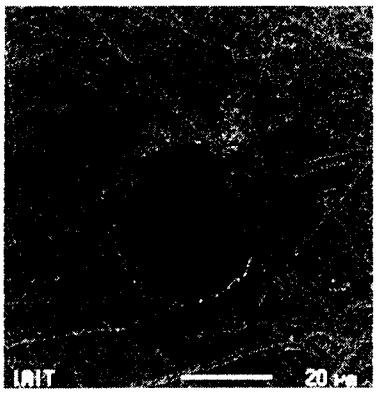
⇒ **Laserlithography (Laser-directwritting)**

⇒ **Personalisation of micromechanical Sensors (Trimming)**

⇒ **Laserinduced Bonding**



Set-up for Laserdirect-writting Sys



mit Nd-Yag-Laser gebohrtes Loch in Si

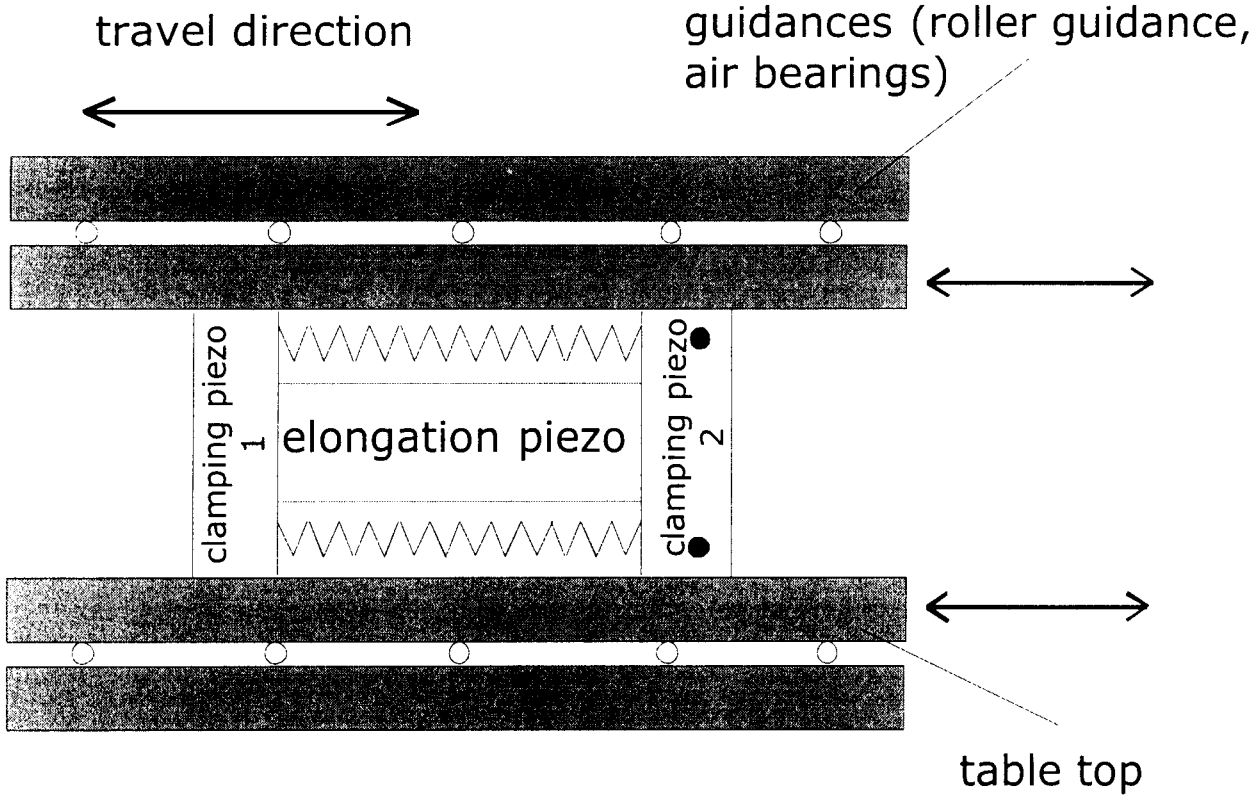
MPCFH-22-1-99_12

Piezoschrittmotor

Ziele

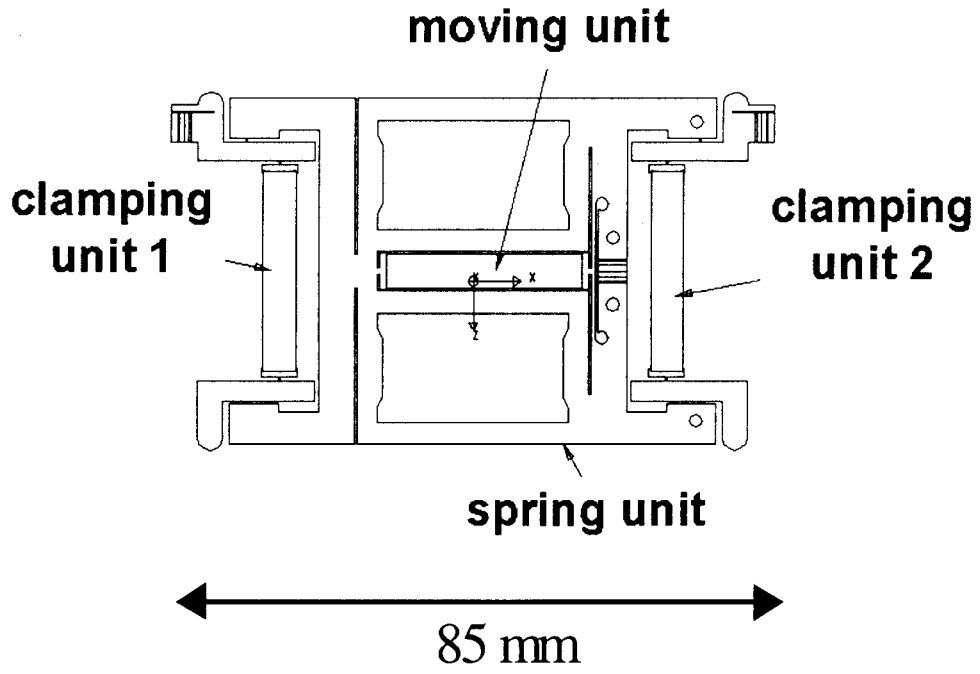
- Aufbau einer Laserbearbeitungsanlage zur Hochpräzisionsbearbeitung
- Einfacher, preiswerter Aufbau
- einfache Ansteuerung
- hohe Genauigkeit: 100 nm = 10^{-7} m
- Verfahrweg $\geq$ 100 mm
- konfigurierbar für unterschiedliche Anwendungen

Principle of Drive (University of Applied Sciences Furtwangen)

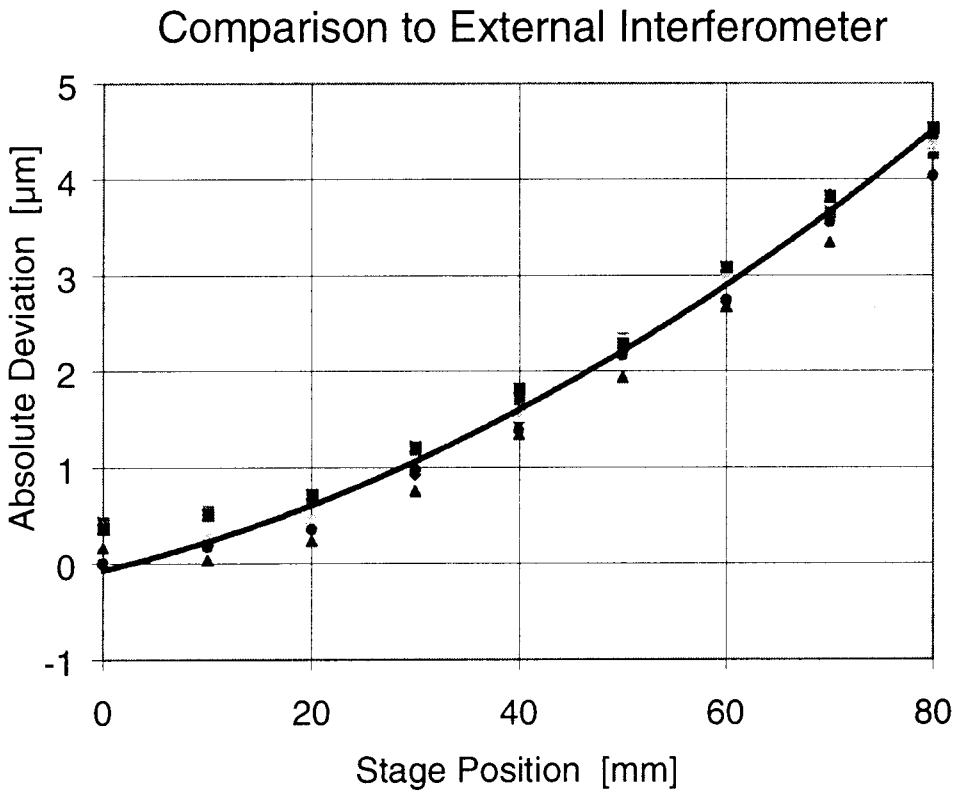


- plain structure
- friction-based clamping
- use of conventional piezostacks
- monolithically integrated driving block
- simple integration of length measuring systems (optical linear decoder, interferometer)

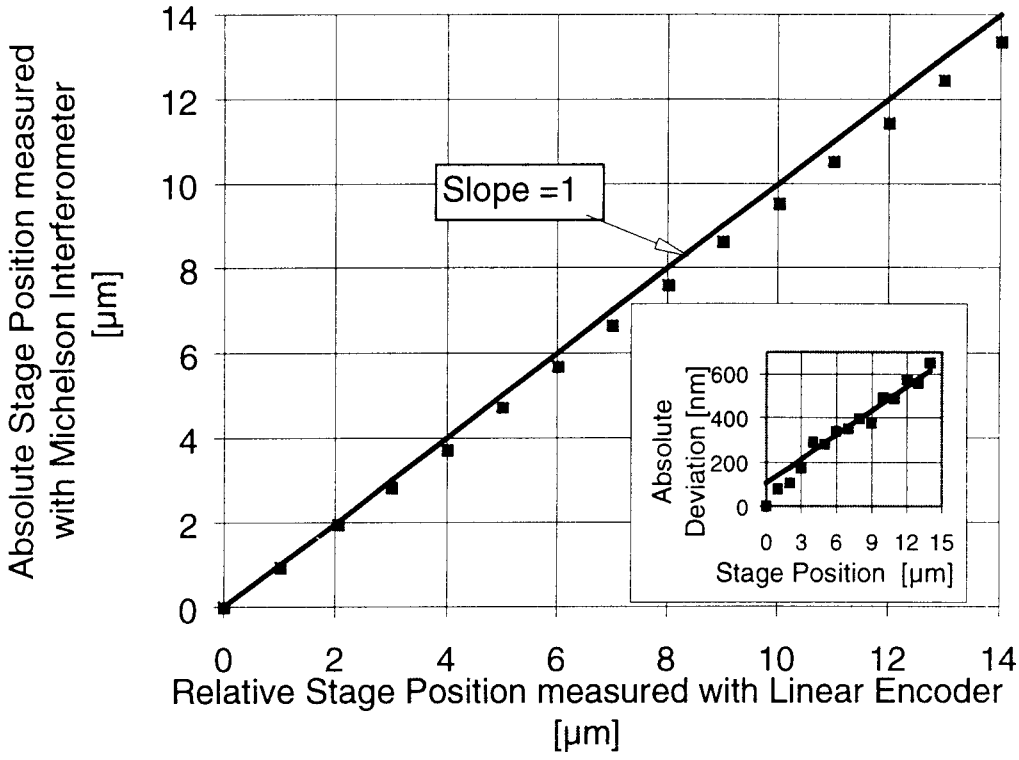
Driving Element



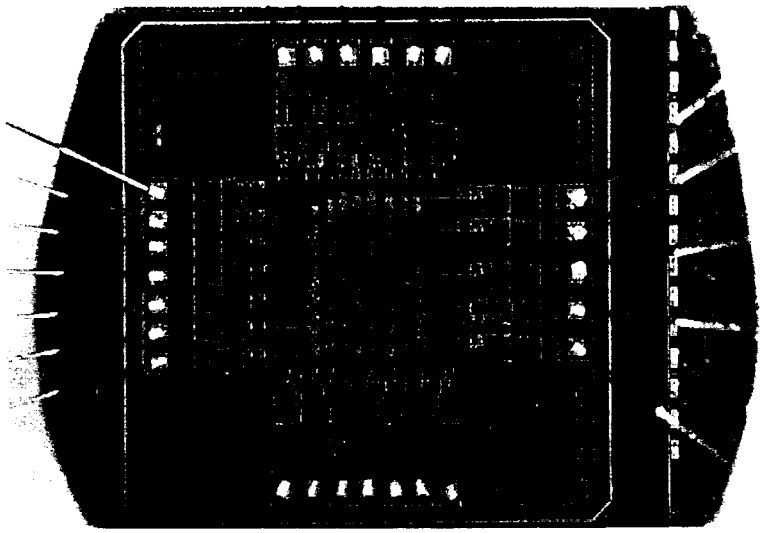
- monolithic block (clamping elements, springs, levers, gears)
- manufactured by wire erosion (laser cutting)
- suitable for mass production


Comparison to External Interferometer
Absolute Deviation [µm]
5
4
3
2
1
0
-1
0
20
40
60
80
Stage Position [mm]

Deviation of Relative and Absolute Stage Position

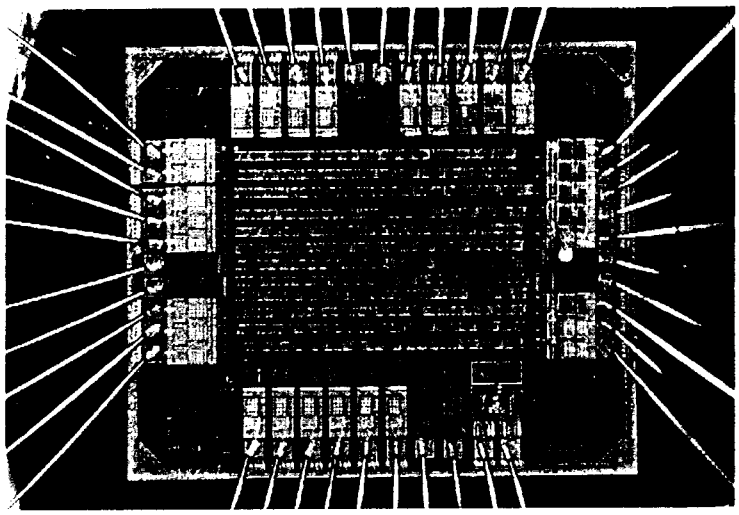


Würfel V4



Entwurf:	Fachhochschule Offenburg Bearbeiter: Oliver Bischoff Betreuer: Prof. Dr.-Ing. Dirk Jansen
Layouterstellung:	Fachhochschule Offenburg (Standardzellenentwurf)
Technologie:	Alcatel Mietec 0,7 µm CMOS (C07M-D)
Chipfertigung:	Europractice, Run 219
Herstelldatum:	März 1998
Kostenträger:	MPC-Mittel FH-Verbund Baden-Württemberg
Chipdaten:	Chipgröße: 2,5 x 2,4 mm^2 Gehäuse: JLCC 44 Komplexität: ca. 5000 Transistoren
Funktion:	Elektronischer Würfel mit Anzeige der Augenzahl über 7 LEDs. Der Würfel "rollt" langsam aus, d.h. die Augenzahl wird nicht sofort, sondern erst nach einer Ausrollzeit von etwa 2 Sekunden angezeigt. Mit dem Rollen wird ein "Klick" generiert, der durch einen Piezo-Summer hörbar gemacht wird. Das Ergebnis wird mit einem aus 2 Klängen bestehenden Ton signalisiert. Bei den Ziffern "6" und "1" wird jeweils eine kurze Melodie gespielt. Der Chip arbeitet mit 3 V aus einer Lithiumbatterie. Daneben werden nur ein Taster, der Uhrenquarz, der Piezo-Speaker und die LEDs mit Vorwiderständen als externe Bauteile benötigt.
Testergebnisse:	Der Chip war bis auf einige kleine Anpassungen eine 1:1 Umsetzung (Remapping) des vorhandenen Designs von ES2-Technologie auf die Mietec 0.7 µm Technologie. Ziel war es, erste Erfahrungen mit er neuen Technologie zu sammeln und die eingesetzte Oszillator-Padzelle zu verifizieren. Der Test ergab, daß der Chip voll funktionsfähig ist. Somit ist der Weg frei, um neue, vor allem komplexere Entwürfe für diese Technologie zu entwerfen und fertigen zu lassen.

Lottozahlengenerator V2



Entwurf: Fachhochschule Offenburg
Bearbeiter: Patrick Erb
Betreuer: Prof. Dr.-Ing. Dirk Jansen

Layouterstellung: Fachhochschule Offenburg (Standardzellenentwurf)

Technologie: AMS CMOS 0,8 µm (CYB)

Chipfertigung: Europractice, Run 250

Herstelldatum: Juli 1998

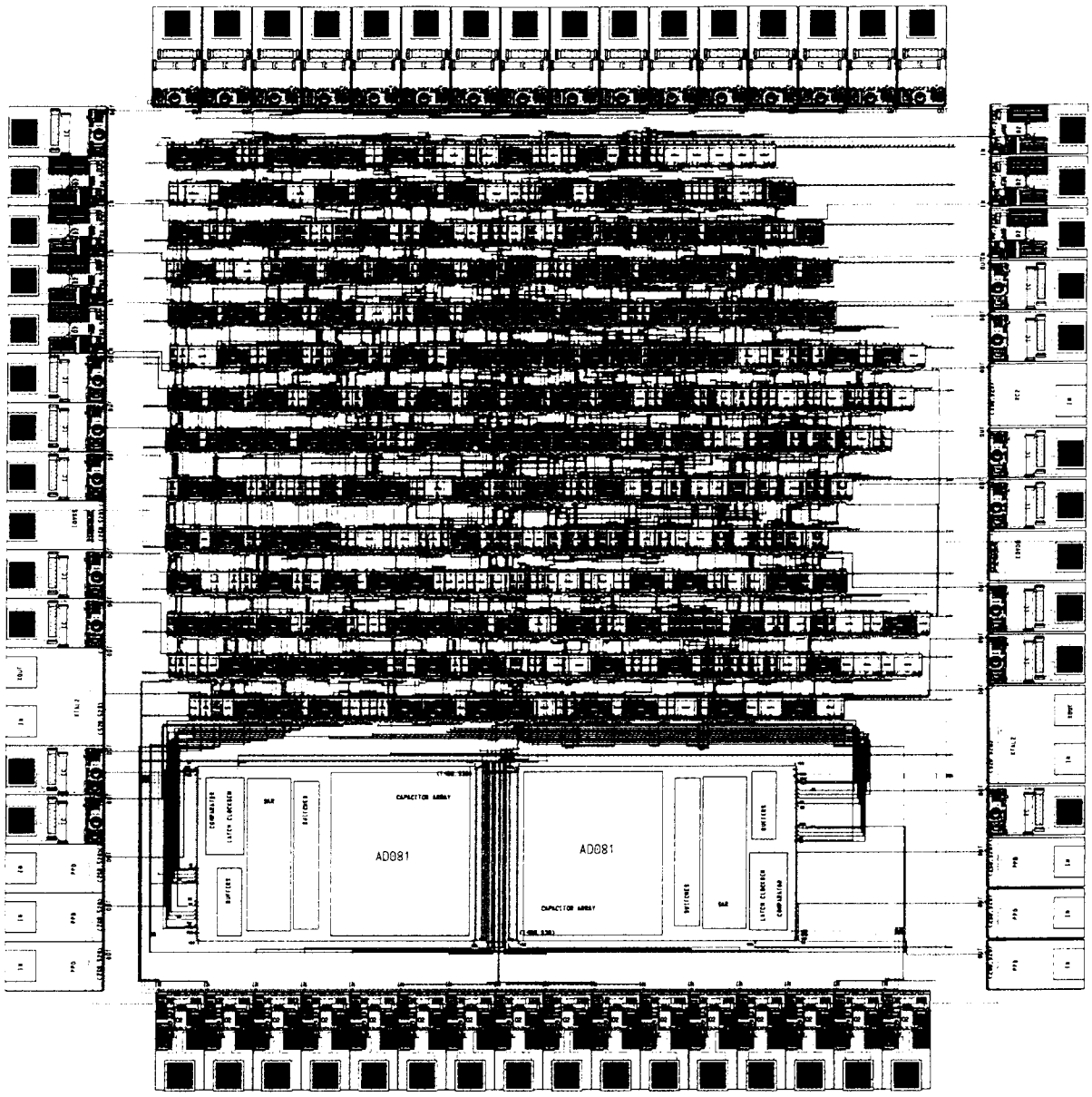
Kostenträger: MPC-Mittel FH-Verbund Baden-Württemberg

Chipdaten: Chipgröße: 2,5 x 2,1 mm^2
Gehäuse: JLCC 44
Komplexität: ca. 6000 Transistoren

Funktion: Lottozahlengenerator "6 aus 49": Die Zahlen werden nacheinander mittels Tastendruck gezogen und mittels LEDs angezeigt. Die Ziehung erfolgt mit einer Ausrollfunktion, dabei wird ein kurzer Ton erzeugt. Sind alle Zahlen gezogen, wird das Badner-Lied gespielt. Desweiteren verfügt der Chip über ein Standby-Modus, so daß er sich automatisch nach einer gewissen Zeit abschaltet, bis erneut eine Taste gedrückt wird.

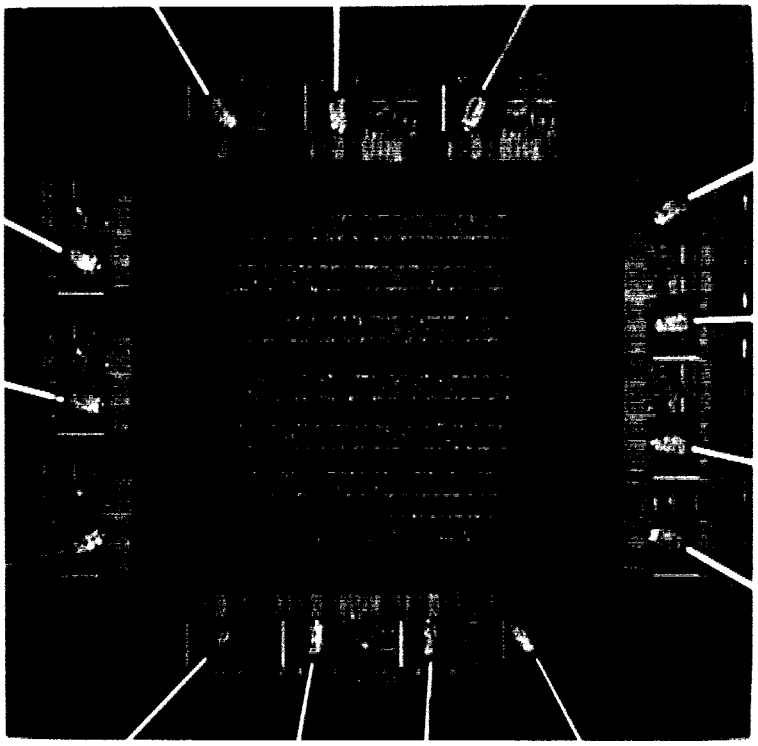
Testergebnisse: Der Chip ist eine 1:1 Umsetzung (Remapping) des vorhandenen Designs von ES2-Technologie auf die AMS 0.8 µm Technologie. Gleichzeitig wurden einige Verbesserungen und Optimierungen vorgenommen. Ziel dieses Entwurfs war es, erste Erfahrungen mit er neuen Technologie zu sammeln und die eingesetzte Oszillator-Padzelle zu verifizieren. Der Test ergab, daß der Entwurf selbst voll funktionsfähig ist, lediglich die eingesetzte Oszillator-Padzelle arbeitet bei 3 V nicht einwandfrei. Grundsätzlich sind wir nun aber in der Lage, um neue, vor allem komplexere Entwürfe für diese Technologie zu entwerfen und fertigen zu lassen.

Microchip "Solar"



Entwurf:	FH- Aalen, EDA- Zentrum
	Bearbeiter: Dipl.-Ing. Andreas Herb
	Betreuer: Prof. Dr. B. Kohlhammer
Chipgröße:	5,7mm x 5,9mm
Transistoren:	4314
Chiphersteller:	Mietec-Alcatel, Belgien
Eigenschaft:	Redesign des Chips zur Optimierung von Photovoltaik-Anlagen; mittels MPP-Tracking und Pulsweitenmodulation wird eine maximale Leistungsabgabe der Photovoltaikanlagen erreicht. Der Chip soll in jedes Modul eingebaut werden, dessen Leistung optimieren und per getrennter Steuerung eine bessere Anpassung an die Bedürfnisse des Verbrauchers ermöglichen.

Steuerung für Fußgänger-Ampel

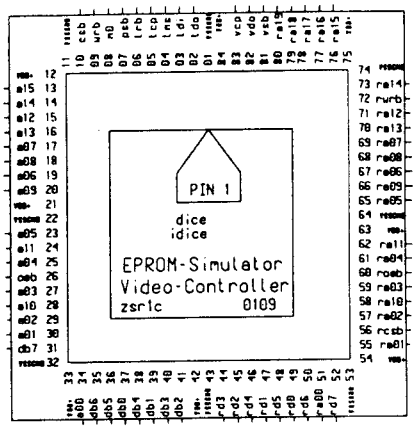


Entwurf:	Fachhochschule Ulm Bearbeiter: J. Müller, W. Schwenk Betreuer: Prof. Dipl.-Phys. Gerhard Forster
Layouterstellung:	Digitaler Standardzellenentwurf
Technologie:	CAE 1,2 µm CMOS, Fa. AMS
Chipfertigung:	Fa. AMS, Österreich, über Europractice, Run 226
Herstelldatum:	2. Quartal 1998
Kostenträger:	MPC-Mittel, FH-Verbund Baden-Württemberg
Chipdaten:	Chipfläche: 1,7 x 1,6 mm^2 Gehäuse LCC 44 Komplexität ca. 500 Gatter, interne Takterzeugung
Funktion:	Das Digital-ASIC, Ergebnis einer Studienarbeit, enthält eine Fußgänger-Ampel-Steuerung. Die Schaltung wurde, ausgehend von einer Verhaltensbeschreibung in VHDL, mit Autologic II synthetisiert. Der Chip ist Bestandteil einer kleinen Demo-Platine in SMD-Technik.

ZSR1 - EPROM-Simulator/CRT-Controller

Funktionsblöcke entworfen, simuliert
und verifiziert im technischen Wahlfach
"CAE Elektronischer Schaltungen",

7. Semester ELEKTRONIK im SS98
durch die Herren
Büchele,Siegfried, Bulach,Eric
Elbs,Alexander, Fischer,Stephan
Genzel,Peter, Hecker,Robert,
Körber,Martin, Kramer,Andreas
Müller,Wolfgang, Schmid,Klaus
Stoppel,Simon, Sassano,Mario
Waitschies,Eldor



Chipfertigung:	IMS Stuttgart
EDIF-Tape-out:	Ende SS98
Komplexität:	ca. 22000 Transistoren
Technologie:	0.8um CMOS Gate-Forest

zsr1 enthält einen EPROM-Simulator und ein Datensichtgerät. Die Baugruppen können parallel genutzt werden. Es werden zwei Betriebsarten - "Simulator" und "VIDEO" - unterschieden.

Als EPROM-Simulator nimmt zsr1 einen Adressraum von 64k-Byte entgegen und bildet ihn auf ein MegaByte (RAM oder EPROM) ab. Das Umschalten der Speicherbereiche erfolgt über interne Register, auf die über eine Test-Schnittstelle, die aus den Signalen trb, tcp, tms, tdi und tdo besteht oder über den PC-seitigen Datenbus zugegriffen wird. Der RAM- oder EPROM-Bereich ist durch externe Bausteine realisiert und wird über die Test-Schnittstelle geladen.

In der Betriebsart "VIDEO" greifen CRT- und Prozessor auf den RAM-Bereich zu, wobei der Prozessor die höhere Priorität besitzt. Dies beeinträchtigt bei ausgedehnten EPROM-Lese-Zyklen eines Single-Chip-Prozessors die Video-Ausgabe. Ein zusätzliches EPROM für den CRT-Treiber "vor" dem zsr1 kann diesen Effekt aber beseitigen, Selektionsleitungen für diesen Anwendungsfall sind bereitgestellt.

Als VIDEO-Controller hat zsr1 uneingeschränkten Zugriff auf den gesamten Speicherbereich, um Video-Daten abzulegen und den Zeichen-Generator anzusprechen. Es wird ein Bildformat von 16 Zeilen mit je 48 Zeichen aufgebaut, welches problemlos auf ein einfaches, handelsübliches Fernsehgerät (video-walkman) passt. Ein Pixel-, und ein Synchronsignal stehen an Pins zur Verfügung, ein Austastsignal kann auf tdo konfiguriert werden. Die drei Signale werden durch vier Widerstände extern zum BAS-Signal zusammengemischt.

Auf den Speicherbereich kann auch über die Test-Schnittstelle zugegriffen werden. Die Zahl der dazu notwendigen Ports ist relativ klein: Funktionalität ist bereits mit drei Pins gegeben. Über die Test-Schnittstelle kann der zsr1 unabhängig von seiner Betriebsart voll kontrolliert werden, auf diesem Wege gelangen Baustein- und RAM-Daten - gesteuert oder beobachtet durch die Test-Pins - in die adressierten Speicherbereiche. Dieser Weg ist notwendig, da ein EPROM-Sockel normalerweise kein Schreibsignal führt. Einen schnelleren Schreib-Zugriff erlaubt ein Signal wrb, als Gegenstück zum EPROM-Lese-Signal oeb. Dieses Signal kann genutzt werden, wenn zsr1 z.B. als "Memory-Expander" auf einen 32-k-RAM-Sockel gesetzt wird. Natürlich besteht auch die Möglichkeit, ggf. wrb in "fliegender Verdrahtung" vom Prozessor zum zsr1 zu führen - was dem Single-Chip-Prozessor dann ein zusätzliches MegaByte statischen Speicher bzw. ein Ausgabe-Medium zur Verfügung stellt...

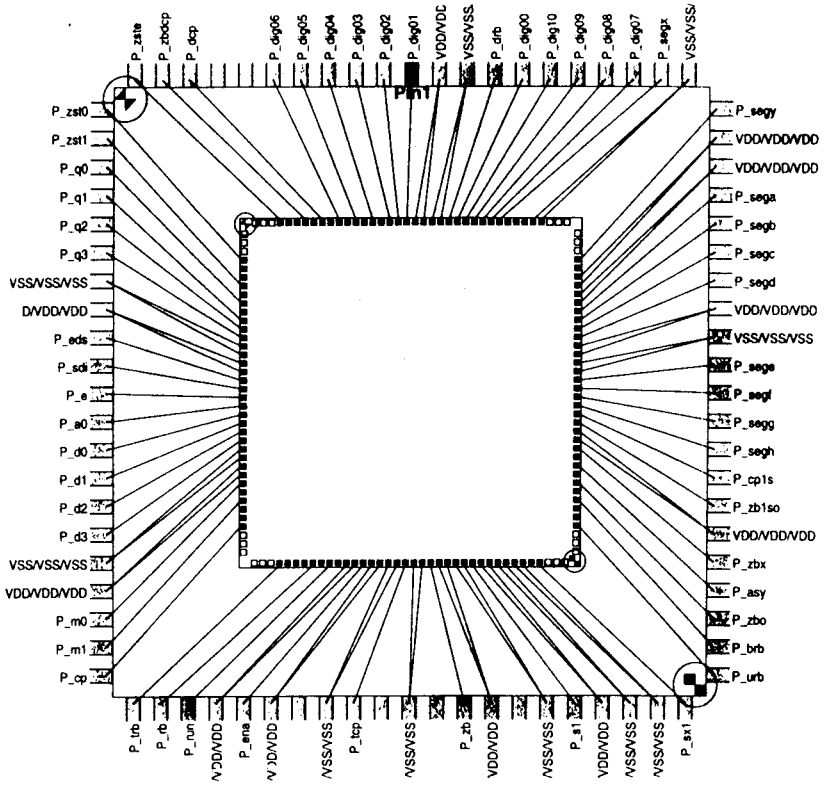
W.Ludescher, 980909

Zählerbaustein ZAE3

Der anwendungsspezifische Schaltkreis ZAE3 wurde im Rahmen der CAE-Vorlesung im WS97/98 vom 5. Semester Elektronik an der Fachhochschule Ravensburg/Weingarten entworfen.
Ziel: Messung von Frequenz, Periode, Ereignis und Zeit im Bereich von DC bis ca 200MHz.

Entwurfs-Team:

Bottlinger, Martin
Buck, Christina
Büchele, Siegfried
Deckert, Yann
Ehrmann, Dieter
Erne, Stefan
Gröllich, Stefan
Häfele, Christof
Keckeisen, Michael
Körber, Martin
Lenk, Armin
Lenker, Oliver
Liebherr, Timo
Prasser, Florian
Reinhardt, Olaf
Schmid, Klaus
Spina, Marco
Straub, Torsten
Strobel, Gero
Waitschies, Eldor
Weiss, Daniel



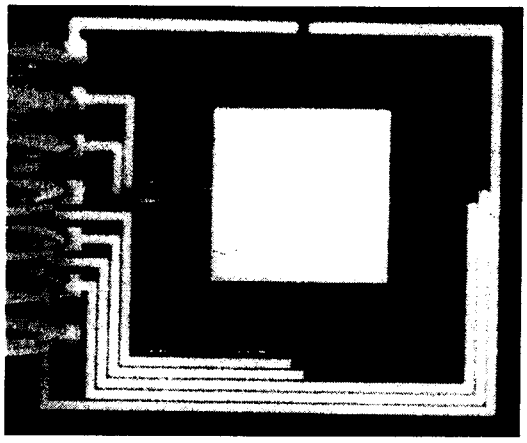
ZAE3 besteht aus den Baugruppen "Vorteiler", "Zeitbasis", "Torsteuerung", "Zählerdekaden", "Summierer" und aus einer "Ausgabe-Einheit". Die Ausgabe-Einheit wird auch zur Anzeige der Uhrzeit genutzt, eine Uhrenschaltung befindet sich mit auf dem Chip. Zur Ankopplung eines externen Mikroprozessors ist eine Prozessor-Schnittstelle vorhanden. Ein Betrieb ohne Mikroprozessor ist ebenfalls möglich - dafür sorgt eine einfache Bedien-Schnittstelle, die 16 Input-Ports bereitstellt. Die Ports können auch vom Prozessor beobachtet werden.

Anwendungen und Funktionen:
Eingangssignale von DC bis ca. 200MHz
Messung der Frequenz, Periodendauer, Multiperiodendauer
Zeitbasis 1MHz, Auflösung 1Hz bei Torzeit 1s
Binäre oder dekadische Vorteiler zuschaltbar
Anzeige über 10 Dekaden, Unterdrückung führender Nullen,
Addition/Subtraktion einer Zwischenfrequenz zur Anzeige der aktuellen Empfangsfrequenz (lokaler Oszillator) eines Empfängers.

Entwurf:	FH Ravensburg-Weingarten CAE-Projekt 5.Semester Elektronik, WS97/98
betreut durch:	Dipl.-Ing.(FH) F.Förster, Prof.Dr.-Ing. W.Ludescher
CAE-Software:	Mentor V8, VHDL, Design-Architekt,...
Chipfertigung:	IMS Stuttgart, im Rahmen des MPC-Verbundes Baden-Württembergs
Komplexität:	ca. 43000 Transistoren
Technologie:	0.8um CMOS Gate-Array

W.Ludescher, 27.Jan.98

mikromechanischer Zweiachsen-Neigungssensor



Projektleitung: Prof. Dr. Ulrich Mescheder

Projektbearbeitung: Dipl.-Ing. (FH) Dirk Efferenn

Entwurf: Institut für Angewandte Forschung der Fachhochschule Furtwangen

Layouterstellung: Fachhochschule Furtwangen

Technologie: 5 µm-Mikromechanik Fachhochschule Furtwangen

Chipfertigung: Mikrolaboratorium Fachhochschule Furtwangen

Herstellung: IV. Quartal 1998

Kostenträger: Ministerium für Wissenschaft und Forschung Baden-Württemberg, Fa. Gottlieb Nestle GmbH & Co. KG, Fa. Genesys Elektronik GmbH

Chipdaten:

Sensorfläche:	6,2 x 5,8 mm²
Auflösung:	0,1°

Funktion: Dieser Sensor besteht aus einer über vier dünne Stege mit dem Chiprahmen verbundenen Masse. Der Massenschwerpunkt liegt dabei unterhalb der Stege. Bei Neigung des Chips versucht die Masse in ihrer ursprünglichen zur Gravitation senkrechten Lage zu verweilen. Dabei werden die dünnen Stege mechanisch verspannt. Auf die Stege aufgebrachte piezoresistive Widerstände werden durch die Verspannung in ihrem Widerstandswert verändert. Diese Widerstandsänderung wird ausgelesen und damit die Neigung des Sensors bestimmt.

Testergebnisse: Bei dieser Sensorversion handelt es sich lediglich um einen Prototypen. Die endgültige Version ist bis Ende 1999 verfügbar.

Die Eckdaten sind dabei:

Sensorgröße:	7 x 7 mm²
Auflösung:	10^{-4} °
absolute Genauigkeit:	$5x10^{-3}$ °