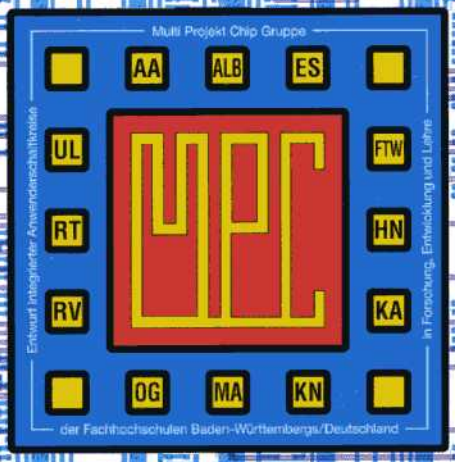
MULTIPROJEKT CHIP-GRUPPE

BADEN - WÜRTTEMBERG

Workshop Januar 2000

Mannheim



MULTIPROJEKT CHIP-GRUPPE

BADEN - WÜRTTEMBERG

Workshop Januar 2000

Mannheim

Herausgeber: Fachhochschule Ulm

Adressen der

MULTIPROJEKT-CHIP-GRUPPE (MPC-Gruppe)

BADEN - WÜRTTEMBERG

http://www.mpc.belwue.de

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

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

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

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

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

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

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

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

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

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

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

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

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

Inhaltsverzeichnis

Workshop-Vorträge

Weiterer Beitrag

Gefertigte Bausteine

Mixed–Signal–Spezifikation und Optimierung des dynamischen Verhaltens von Digital–Analog–Wandlern in Deep–Submicron CMOS–Technologie

Prof. Dr.–Ing. W. Bonath
FH Gießen–Friedberg
Wiesenstraße 12
D–35290 Gießen

Kurzfassung

Der Beitrag beschreibt die Optimierung von Digital–Analog–Wandlern in Hinblick auf ihr dynamisches Verhalten. Dabei wird ein mixed–Signal–Modell des Wandlers vorgestellt, in welchem strukturelle Analogmodelle durch analoge Verhaltensbeschreibungen ergänzt werden, so daß eine effiziente mixed Signal–Simulation möglich ist.

1. Mixed Signal–Simulation und Verhaltensbeschreibungen

Ausgangspunkt der Mixed–Signal–Simulation ist die Kopplung von klassischen analogen Schaltungs– mit digitalen Gattersimulatoren, wobei das Hauptziel darin besteht, die maximale Größe simulierbarer analog–digitaler Schaltungen zu erhöhen, für die digitalen Teile auf Logikebene mit grobem Zeitraster zu simulieren und analoge Signale mit sehr kleinem Zeitschritt genau zu erfassen. Die Grenzen socher Simulatoren liegen aber auch wieder im begrenzten Aufwand möglicher Schaltungsbeschreibungen, d.h. bei praktisch realisierbaren Simulationszeiten und –speicherbedarf. Zu den erforderlichen Ressourcen für die Einzelsimulatoren kommt der Overhead für die Kommunikation zwischen ihnen hinzu.

Aus der Digitaltechnik stammt der Ansatz, neben strukturellen Beschreibungen auch das Verhalten von Hardware zu spezifizieren und zu simulieren. Neben der damit erreichten enormen Steigerung der Komplexität simulierbarer Schaltungen ist es aber isbesondere auch möglich, zusätzliche Funktionalität zu erfassen, welche "on chip" nicht realisierbar und mit klassischen Simulatoren nicht simulierbar ist.

Während auf der digitalen Seite seit längerer Zeit die standardisierten Sprachen VHDL und Verilog existieren, entwickelten sich analoge Sprachen zur Verhaltensbeschreibung nur zögernd als Eigenentwicklungen der Simulatorhersteller (z.B. MAST und HDLA). Erst in jüngster Zeit entstand auch für analoge Verhaltenspezifikationen der VHDL–AMS–Standard.

Hochwertige Mixed–Signal–Simulatoren erlauben mittlerweile die gemischte Simulation von analogen und digitalen Systemen sowohl auf struktureller als auch auf Verhaltensebene. Die mixed–Signal–Simulation erlaubt damit nicht nur, sehr komplexe Schaltungen zu spezifizieren und zu simulieren, sondern auch die Beschreibung und Optimierung neuer Problemstellungen, speziell an der Schnittstelle zwischen analoger und digitaler Schaltungstechnik.

Das in diesem Beitrag vorgestellte Optimierungsbeispiel konzentriert sich auf die Optimierung von Digital–Analog–Wandlern für die Mobilkommunikation. Charakteristikum solcher Systeme sind zum einen die durch die extrem hohen Stückzahlen bedingten Optimierungspotentiale, durch das kritische dynamische Verhalten sowie durch die zur Anwendung kommenden "Deep–Submicron–CMOS"–Prozesse.

2. Wandlerarchitektur und Optimierungsverfahren

Wie die folgende Abbildung 1 zeigt, wird ein mit der Taktrate f_{sample} = 15 MHz ankommendes Signal gewandelt und anschließend gefiltert. Die wichtigsten Randbedingungen für den DAC liegen zum einen in einem möglichst kompakten Aufbau (=minimale Siliziumfläche), zum anderen in einem optimierten dynamischen Verhalten. Das optimale Systemverhalten ergibt sich durch Quantisierungseffekte des Wandlers bzw. des vorgegebenen digitalen Signals; die dynamischen Effekte des Wandlers sollen soweit als möglich unterdrückt werden.

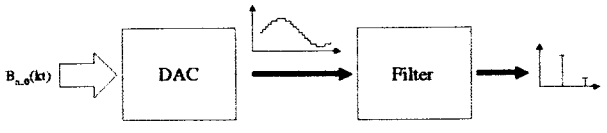


Abb. 1

Eine besonders kompakte und sehr reguläre

Struktur ist der spannungsgesteuerte Wandler (Widerstandskette, "Rstring").

Für einen n-bit-DAC werden 2^n-1 Widerstände in Reihe geschaltet und der so entstandene Spannungsteiler mit einer positiven und negativen Referenzspannung + V_{ref}/$-V_{ref}$ gespeist. Für die hier untersuchten Strukturen wurde $-V_{ref}=0V$ gesetzt. Es entstehen 2^n Spannungen mit einer Auflösung von $V_{ref}/(2^n-1)$. Durch Schalter wird in Abhängigkeit vom zu wandelnden Binärwort b_{n-1} b_{n-2} ... b_1 b_0 genau eine Zwischenspannung durchgeschaltet und ergibt die Ausgangsspannung V_{out} nach der folgenden Beziehung:

$$V_{out} = V_{ref}/(2^n-1) \cdot [b_0 + b_1 \cdot 2^1 + \ldots b_{n-1} \cdot 2^{n-1}]$$

Beispiel: n=2 ergibt 3 Widerstände. Bei $V_{ref}=1.5V$ beträgt die Auflösung 500mV und es ergibt sich die in Abb.2 gezeigt Übertragungskennlinie:

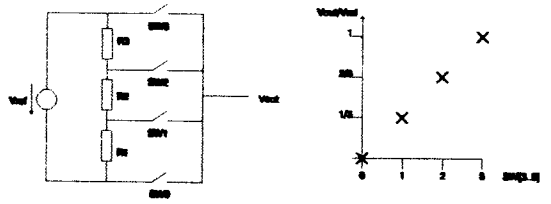


Abb.2: Schaltung und Kennlinie eines 2-bit DACs

Die Schalter werden durch n-Kanal-MOSFETs realisiert. Um die Ansteuersignale für eine Schalterstruktur nach Abb. 1 zu erzeugen, muß das Eingangssignal b_{n-1} b_{n-2} ... b_1 b_0 vollständig dekodiert werden, wodurch sich eine relativ aufwendige Ansteuerschaltung ergibt.

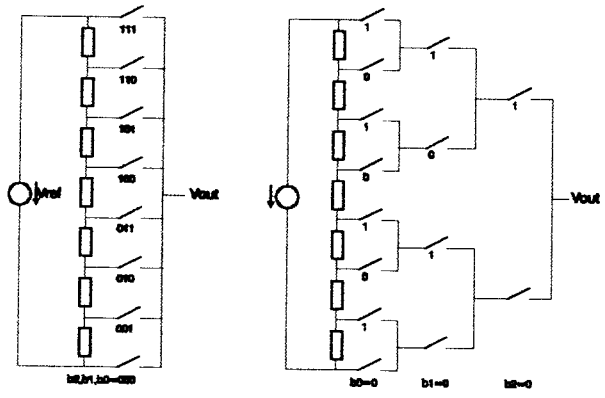


Abb.3: Schalterstrukturen für einen 3-bit-Wandler (vollständige Dekodierung bzw. Baumstruktur)

Mit einer baumartigen Schalterstruktur kann man allerdings auf die Dekodierung vollständig verzichten. Abb. 3 zeigt dies am Beispiel eines 3-bit Wandlers.

Aus diesen beiden Realisierungsformen können weitere Varianten abgeleitet werden, indem jeweils Teile des Ansteuerwortes dekodiert werden. Die folgende Abbildung 3 zeigt die zwei Varianten, die sich noch für den 3-bit-Beispielwandler ergeben:

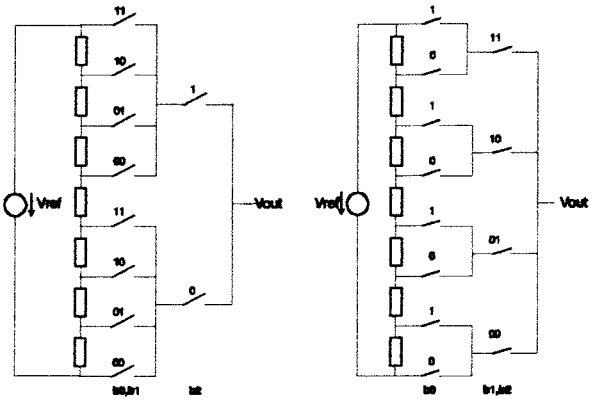


Abb.4: Teilweise dekodierte Schalterstrukturen

Die Anzahl der möglichen Varianten steigt exponentiell mit der Bitzahl n, für n=8 ist eine vollständige Übersicht kaum noch möglich.

Zum Vergleich des Aufwandes sollen die beiden Grundformen „vollständige Dekodierung" und „Binärbaumstruktur" untersucht werden:

„Vollständige Dekodierung": Es werden 2^n Schaltertransistoren und pro Schalter je 1 Ansteuergatter mit n Eingängen benötigt, sowie zusätzlich Inverter und Treiber für die Ansteuerschaltung. Insgesamt folgt eine Minimalanzahl von $2^n + n \times 2^{n+1}$ Transistoren.

Für die binärbaumartige Schalterstruktur werden dagegen nur die Schaltertransistoren benötigt, d.h. für den gesamten Wandler reichen $2^n + 2^{n-1} + \ldots + 2^1 = 2^{n+1} - 2$ Transistoren aus.

Zahlenbeispiel: Für einen 8-bit-Wandler benötigt die vollständige Dekodierung 4352 und die Binärbaumrealisierung 510 Transistoren.

Unberücksichtigt bei dieser einfachen Aufwandsabschätzung bleiben zunächst:

- die Latches für die Ansteuerung
- Inverter, Treiberschaltungen und Transistorgrößen (W/L)
- Layoutstrukturen und damit erreichbare Flächenverhältnisse

Ein großes Problem bei der Binärbaumrealisierung ist das dynamische Verhalten, da die Zahl der Schalter, die bei einem Wechsel des Eingangswortes gleichzeitig bewegt werden, sehr hoch sein kann (maximal alle $2^{n+1} - 2$ Schalter, minimal nur einer). Wenn b0 wechselt, werden z.B. alle Schalter der ersten Stufe gleichzeitig bewegt. Gerade diese Situation wird bei einem langsamen Eingangssignal häufig auftreten. Eine theoretische Abschätzung ist schwierig, da die Abhängigkeit vom Eingangssignal sehr komplex ist.

Um die verschiedenen DAC–Architekturen hinsichtlich ihres dynamischen Verhaltens zu bewerten, ist die vollständige Simulation von typischen Eingangssignalen erforderlich. Im einfachsten Fall mit einer gemischt digital–analogen Simulation (Eingangssignal und Ansteuerlogik digital, Ausgangssignal, Rstring und Schalter analog). Eine solche Simulation wurde wegen des sehr hohen CAD–Aufwandes nicht durchgeführt.

Um mit realistischen Rechenzeiten auszukommen, wurde die Möglichkeit des SABER–Simulationssystems ausgenutzt, gemischt auf analoger Verhaltens– und Transistorebene zu modellieren und zu simulieren. Abb. 5 verdeutlicht das Simulationskonzept:

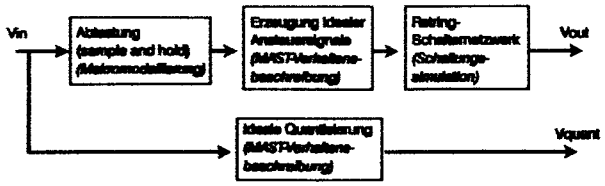


Abb. 5: Simulationskonzept der Rstring–Simulation

Das analoge Eingangssignal V_{in} wird zunächst in einer *Sample and Hold*–Schaltung zeitdiskretisiert, um die reale Umgebung des Wandlers nachzubilden. Diese *Sample and Hold* Schaltung wurde mit idealen Bauelementen in einem Makromodell beschrieben. Aus der zeitdiskretisierten Spannung werden dann in Abhängigkeit der zu simulierenden Wandlerstruktur die Ansteuersignale für die Schalter erzeugt. Der Algorithmus dazu wurde vollständig in ein MAST–Templates beschrieben. Die Ausgangsspannung V_{out} ergibt sich dann aus einer klassischen Schaltungssimulation der Widerstände und Transistorschalter. Die parallel ablaufende ideale Quantisierung —ebenfalls in Form eines MAST–Templates —liefert ein ideales Ausgangssignal V_{quant} zum Vergleich.

Der Vorteil dieser Simulationsmethodik liegt darin, daß insgesamt eine reine Schaltungssimulation durchgeführt wird, es gibt keine Kopplung zu einem Digitalsimulator, und die MAST–Beschreibungen werden für die Simulation direkt in Elemente der zu simulierenden Knotenadmittanzmatrix überführt. Insgesamt ist der Simulationsaufwand damit sehr gering.

Da die zunächst untersuchten 6–bit–Strukturen bereits Rechenszeiten im Stundenbereich benötigten, wäre eine „klassische" mixed Signal Simulation mit den heute verfügbaren Werkzeugen wahrscheinlich nicht realisierbar gewesen.

3. Übersicht über die entwickelten Modelle

3.1. Sample and Hold

Die Sample and Hold–Schaltung besteht aus einem Makromodell, d.h. einer idealen Schalter–Kondensatorkombination nach Abb. 6.

Die hier verwendete Realisierung der Sample and Hold Schaltung enthält ideale Bauelemente aus den SABER–Simulationsbibliotheken in einem Stromlaufplan; identische Stromlaufpläne sind in allen parallel untersuchten Wandlerstrukturen enthalten.

Die Eingangsspannung V_{in} (Spannungsquelle V1) liefert ein Sinussignal mit $f=125kHz$, $V_{pp}=500mV$, $V_{SS}=1V$. Der ideale spannungsgesteuerte Schalter besitzt die folgenden Eigenschaften: R_{ON}=100mW, R_{OFF}=10GW, $V_1=V_2$=250mV.
Simulationsergebnisse und der Vergleich der Spektren ergaben, daß die Modellbildung ausreichend genau ist.

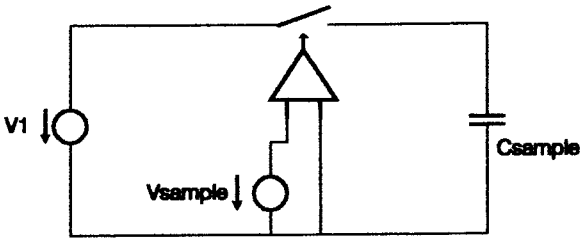


Abb.5: Sample– and Hold Schaltung mit idealen Bauelementen

Eine Rechteckspannung (Spannungsquelle Vsample) realisiert das Abtastsignal: $t_{PULSE}=10ps$, $t_R=t_F=1ps$, $f=16MHz$, $t_P=62.5ns$.

Die Kapazität von C_{sample} und die Pulsquelle V_{sample} wurden so dimensioniert, daß das Ausgangssignal einerseits mit ausreichender Genauigkeit den Eingangswert erreicht, andererseits die resultierende Zeitkonstante $t=R_{ON}\cdot C_{sample}$ möglichst groß wird (Rechengenauigkeit vs. –aufwand!)

Um die Spektren mit Simulatoren auswerten zu können, wurde ein geradzahliges Verhältnis zwischen Signal– und Abtastfrequenz gewählt („spectral leakage"–Effekt, [1]).

3.2. Ideale Quantisierung

Als "Testbench" wurde ein idealer Quantisierer als MAST–Template realisiert, welche eine analoge Eingangsspannung (Knoten „inp") in eine analoge Ausgangsspannung (Knoten „outp") umsetzt.

Für n=6 wird der Ausgangsspannungsbereich von

0 bis V_{REF} in 2^6 Spannungswerte unterteilt und eine Quantisierung mit einer Auflösung von $V_{ref}/(2^n-1) = V_{ref}/63$ durchgeführt. Die Realisierung in MAST erfolgt über die Implementierung der *modulo* –Funktion, in MAST mit Multiplikation und *int* –Statement:

```
template quantizer8 vin vout vgnd = i
#
#!!!! voraussetzung: Vref=1.02V, 8bit, Aufloesung 4mV !!!!
#
electrical vin, vout, vgnd
number i

{
val v vc
var i ivs

values {
      vc=0.004*int((v(vin)+0.002)/0.004)
    }v

equations {
        i(vout->vgnd)+=ivs
        ivs: v(vout,vgnd)=vc
       }

}
#
#
#
quantizer8.call vsin vsinq gnda = 0
```

3.3 Rstring–Schalternetzwerk

Es wurden verschiedene 6–bit und 8–bit Strukturen untersucht.

Jede einzelne Wandlerstruktur besteht aus einem Stromlaufplan, d.h. einem klassichen strukturellen Simulationsmodell sowie MAST–Templates, d.h. der Verhaltensbeschreibung für die Ansteuerschaltungen.

Es wurden ideale Widerstände, Kapzitäten und NMOS–Transistoren im BSIM3V3.1–Modell mit zunächst minimalem W/L verwendet. Alle Ansteuersignale werden durch MAST–Templates erzeugt.

Eine Kapazität $C_L=100fF$ stellt die angenommene Belastung des Wandlers dar; der Anschluß realer Oprationsverstärker war aufgrund der hohen Rechenzeiten nicht sinnvoll möglich.

Sämtliche Schaltungen arbeiten mit einer Versorgungsspannung von V_{DD}=2.5V, die Referenzspannung V_{ref} betrug bei den 6–bit–Schaltungen 1V, bei den 8–bit Wandlern 1.024V (Vermeidung von Rundungsfehlern bei der numerischen Auswertung).

3.4. Erzeugung der Ansteuersignale

Der Algorithmus für die Erzeugung der

Ansteuersignale ist abhängig von der jeweils untersuchten Wandlerstruktur, demzufolge unterscheiden sich auch die MAST–Templates aller implementierten Wandler.

Im einfachsten Fall, bei der vollständigen Ausdekodierung, wird genau ein Schalter aktiviert.

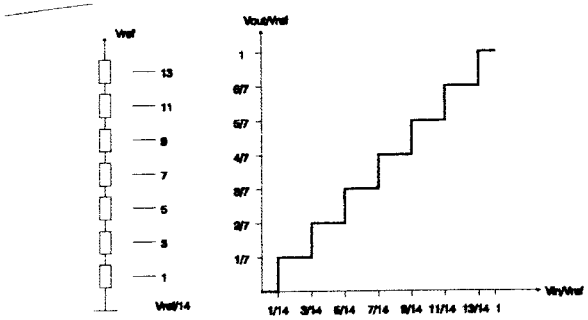


Abb. 7: Kennlinie zur Ansteuerung der Schalter (idealer ADC)

Abb. 7 zeigt am Beispiel von n=3 die ideale Kennlinie, anhand derer sich aus einer analogen Eingangsspannung V_{in} die entsprechenden Schalterstellungen ergeben. Entsprechende Kennlinien ergeben sich für die untersuchten 6– bzw. 8–bit Wandler.

Aus Symmetriegründen sind jeweils der erste und letzte Eingangsspannungsbereich jeweils auf den halben Wert reduziert. Durch den symmetrischen Aufbau sind alle Spannungssprünge durch einen Index adressier– und berechenbar; im o.g. 3–bit Beispiel ergeben sich die Grenzen V_1 und V_2 für den Schalter A_i, $i=0..7$ durch die Formel:

$$V_1 = (2i-1) \cdot V_{ref} / 14, \qquad V_2 = (2i+1) \cdot V_{ref} / 14$$

In der MAST–Beschreibungssprache wurde zur Realisierung eine Funktion „*dac_mux_...*“ spezifiziert, die einen Schaltertransistor durchsteuert, wenn die Eingangsspannung in einem vorgegebenen Fenster liegt. Zur Parametrisierung dieser Funktion werden der Gateknoten der Eingangsspannung „*out1*“ sowie ein Index „*i*“ für die Berechnung des Spannungsfensters vorgegeben. Die entsprechende MAST–Funktion muß dann für jeden einzelnen Schalttransistor mit den jeweiligen Parametern versorgt aufgerufen werden.

Bei den zwei– und dreistufigen Schalterstrukturen wird die Ansteuerungsfunktion komplexer, da ja eine Ansteuermatrix vorliegt, bei der die niedrigwertigen bits die erste(n) Stufe(n) direkt versorgen. Die Ansteuerfunktion für die erste bzw. bei einer dreistufigen Struktur auch die zweite Stufe muß den gesamten Bereich der Eingangsspannung nun transformieren auf den Spannungsbereich, der der jeweiligen Bitkombination entspricht.

Die Struktur der jeweiligen Ansteuerfunktionen „*dac_mux_...*" ist – bei unterschiedlichen Zahlenwerten – für alle untersuchten Beispiele gleich:

In einem ersten Schritt ergibt sich eine reduzierte Eingangsspannung $V_{in}' = V_{in} \bmod (2/2^{n+1}-2)$, wobei j die Anzahl der in der Stufe ausdekodierten bits darstellt.

Nach der Transformation kann dann durch Selektion der entsprechende Schalter geschlossen werden. Auch hier sind die MAST–Funktionen parametrisiert und werden mehrfach aufgerufen.

Alle Ansteuerschaltungen wurden durch Simulation (DC– und Transientensimulation) verifiziert.

Zur Ansteuerung der Schalttransistoren wurden verschiedene Schaltungsvarianten untersucht:

A. Ansteuerung mit einer idealen Spannungsquelle

Bei der Ansteuerung mit einer idealen Spannungsquelle bedingt die MAST–Realisierung eine Flankensteilheit, die im Bereich der Simulator–Zeitschrittweite liegt. Damit treten auch in den mehrstufigen Wandlerstrukturen keine realistischen dynamischen Effekte auf.

B. Ansteuerung durch Stromquellen mit endlichen Innenwiderständen

Abb. 8 zeigt eine Schaltung aus idealen Bauelementen, welche den Ein– bzw. Ausschaltvorgang mit minimalem Simulationsaufwand, aber ausreichend genau modelliert:

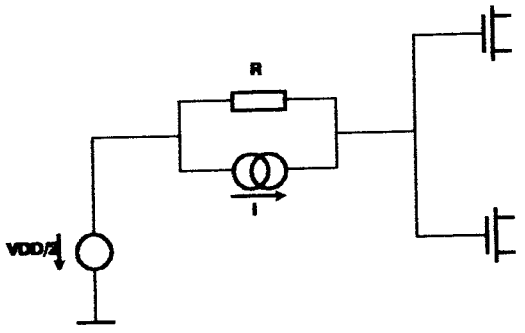


Abb. 8: Ansteuerschaltung für die Rstring–Schaltermatrix

Die Stromquelle I und der Widerstand R werden so dimensioniert, daß zum einen eine vorgegebene Zeitkonstante erreicht wird und zum anderen die Spannungsendwerte von 0V bzw. V_{DD} erreicht werden.

Bei unterschiedlicher Zahl der anzusteuernden Schalttransistoren wurden die

Ansteuerschaltungen für einheitliche und zunächst symmetrische Fall– und Anstiegszeiten $t_R = t_f =$ *1ns* dimensioniert.

Durch die endlichen Flankensteilheiten ergeben sich neben den erwarteteten *clock–feed–through* –Effekten auch zusätzliche dynamische Effekte, da Ein– und Ausschaltflanken sich überlappen (müssen).

Zur Verringerung dieser Taktüberlappung wurden auch unsymmetrisch schaltende Treiber implementiert (unsymmetrische Werte für V_{DD}, unterschiedliche Werte für I und R jeweils für Ein– und Ausschaltvorgang).

Die MAST–Beschreibung der Ansteuerschaltung ist sehr einfach und enthält nur die Parallelschaltung aus Stromquelle und Widerstand.

C. Ansteuerschaltung durch Treiberschaltung

Eine sehr genaue Modellierung ist durch die Implementierung eines CMOS–Treibers möglich, allerdings steigt hierbei der Simulationsaufwand an. Vergleiche ergaben, daß der beste Aufwands–/Genauigkeitskompromiss durch Lösung B, d.h. die gesteuerte Stromquelle, erreicht werden kann.

4. Simulations– und Optimierungsergebnisse

Da die Modelle bereits für n=6 bit recht umfangreich werden, und dann für n=8 und 9bit sehr unübersichtlich sind, ist zur Vorbereitung vergleichender Simulationen ein relativ hoher Verifikationsaufwand erforderlich. Die Funktionalität aller Modelle wurde anhand von Gleichspannungs– und Transientenanalysen nachgewiesen.

Ein Vergleich der verschiedenen Wandlerarchitekturen fand für 6– und 8–bit–Strukturen durch Simulationen im Zeitbereich und Fouriertransformationen statt. Durch die "Testbench" ist es möglich, die Einschwingvorgänge von den restlichen Quantisierungseffekten zu trennen und dann einzeln, beispielsweise durch Effektivwertbildung oder die Betrachtung der Maxima, zu bewerten.

Die abgebildeten Simulationsergebnisse zeigen bereits bei einfachen Sinussignalen eine Vielzahl von dynamischen Störungen, Abbildungen auf einzelne Schalterübergänge gestalten sich sehr schwierig.

Die geringsten dynamischen Effekte ergeben sich bei zweistufigen Wandlern mit möglichst vielen Dekodierbits in den Eingangsstufen.

Die implementierten Modelle erlauben auch, die Dimensionierung von Schalt- und Treibertransistoren zu untersuchen; es ergeben sich jeweils minimale Störungen bei "mittleren" W/L-Verhältnissen, d.h. ein Optimum ist vorhanden.

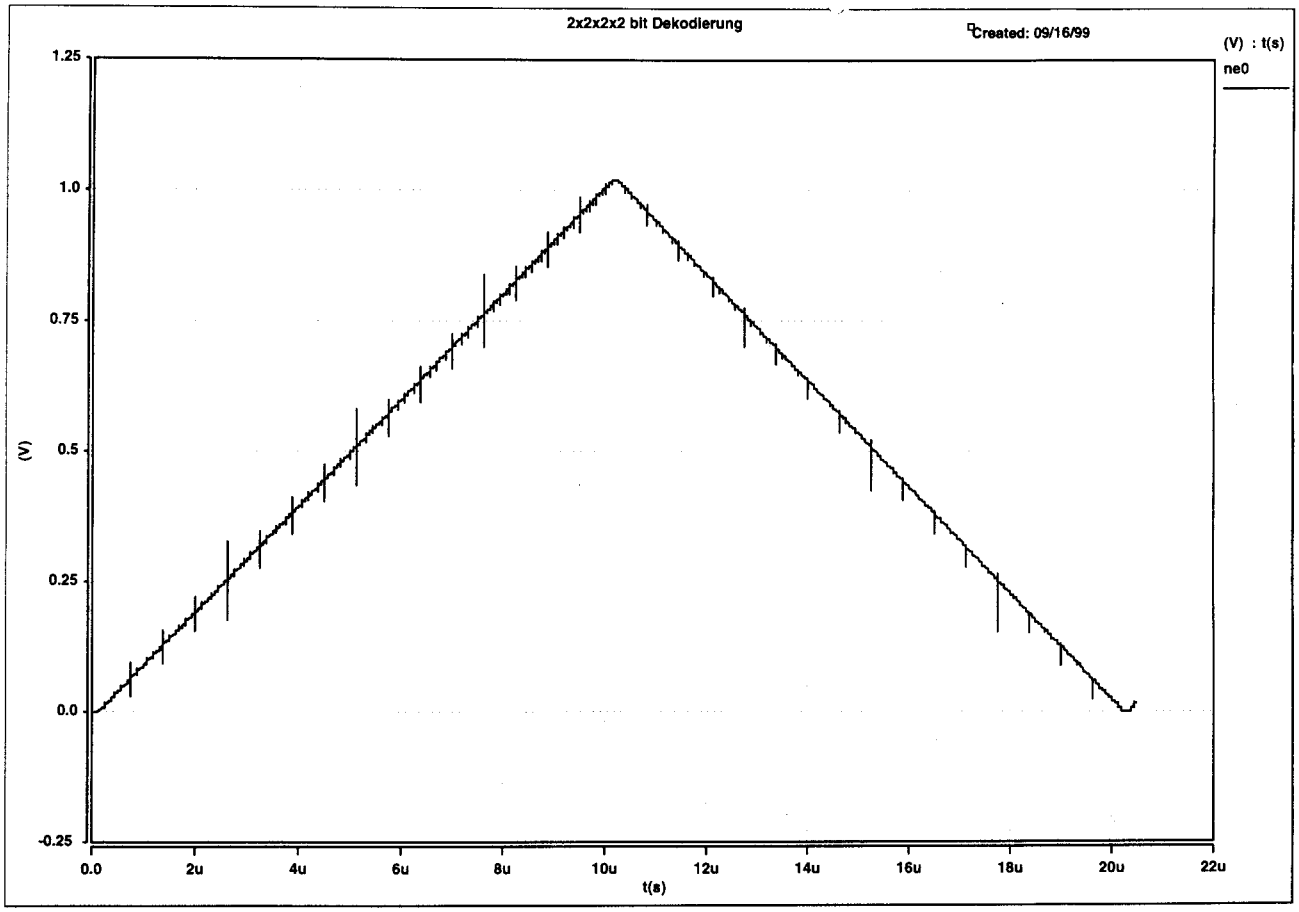
Als sehr viel schwieriger erweist sich die Bewertung der simulierten Spektren, insbesondere wegen der hohen Empfindlichkeit gegenüber Nichtlinearitäten. Neben reinen Sinussignalen wurden auch Mehrtonsignale untersucht; die Ergebnisse bestätigen im wesentlichen die Ergebnisse der Transientensimulationen.

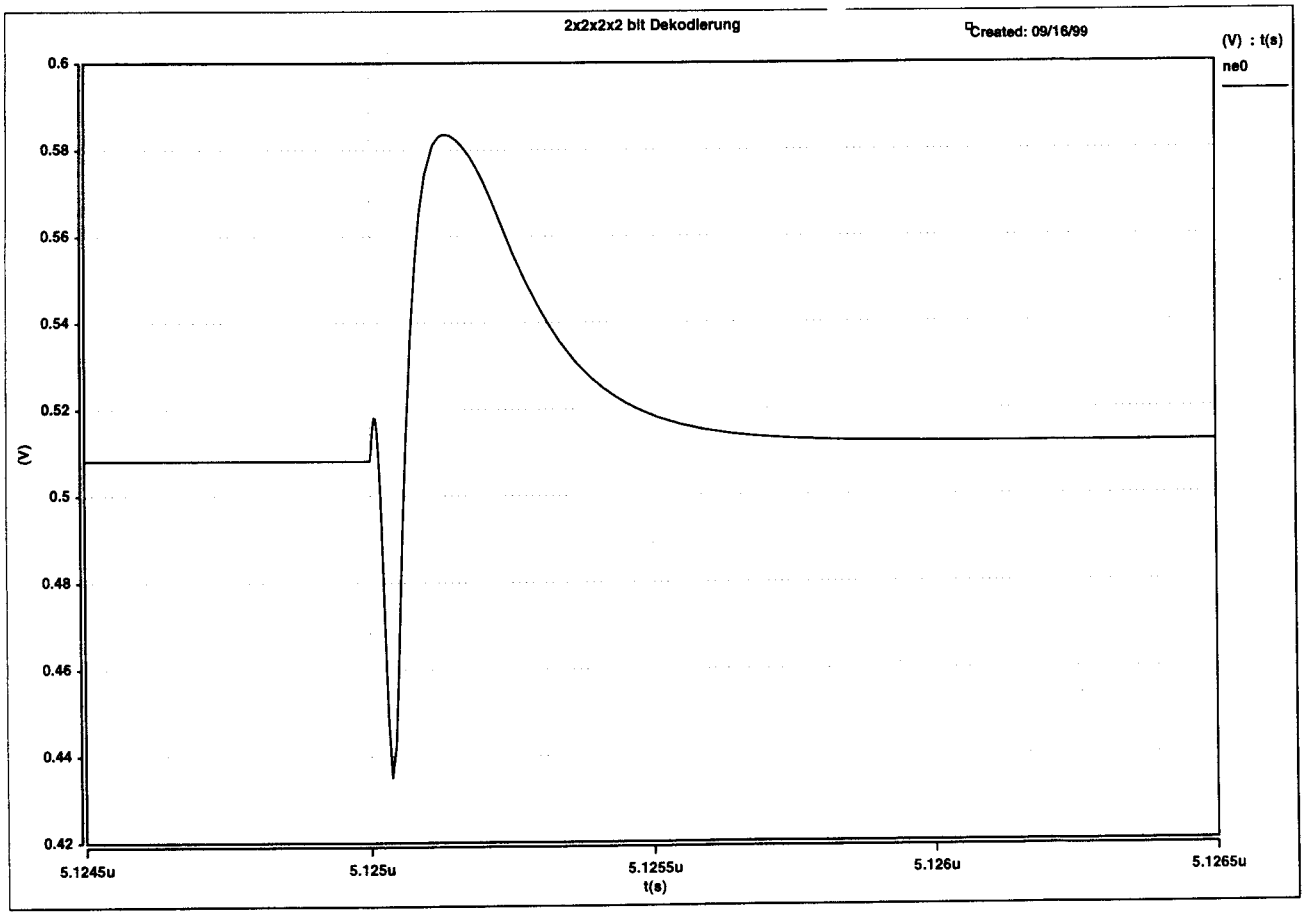
5. Zusammenfassung

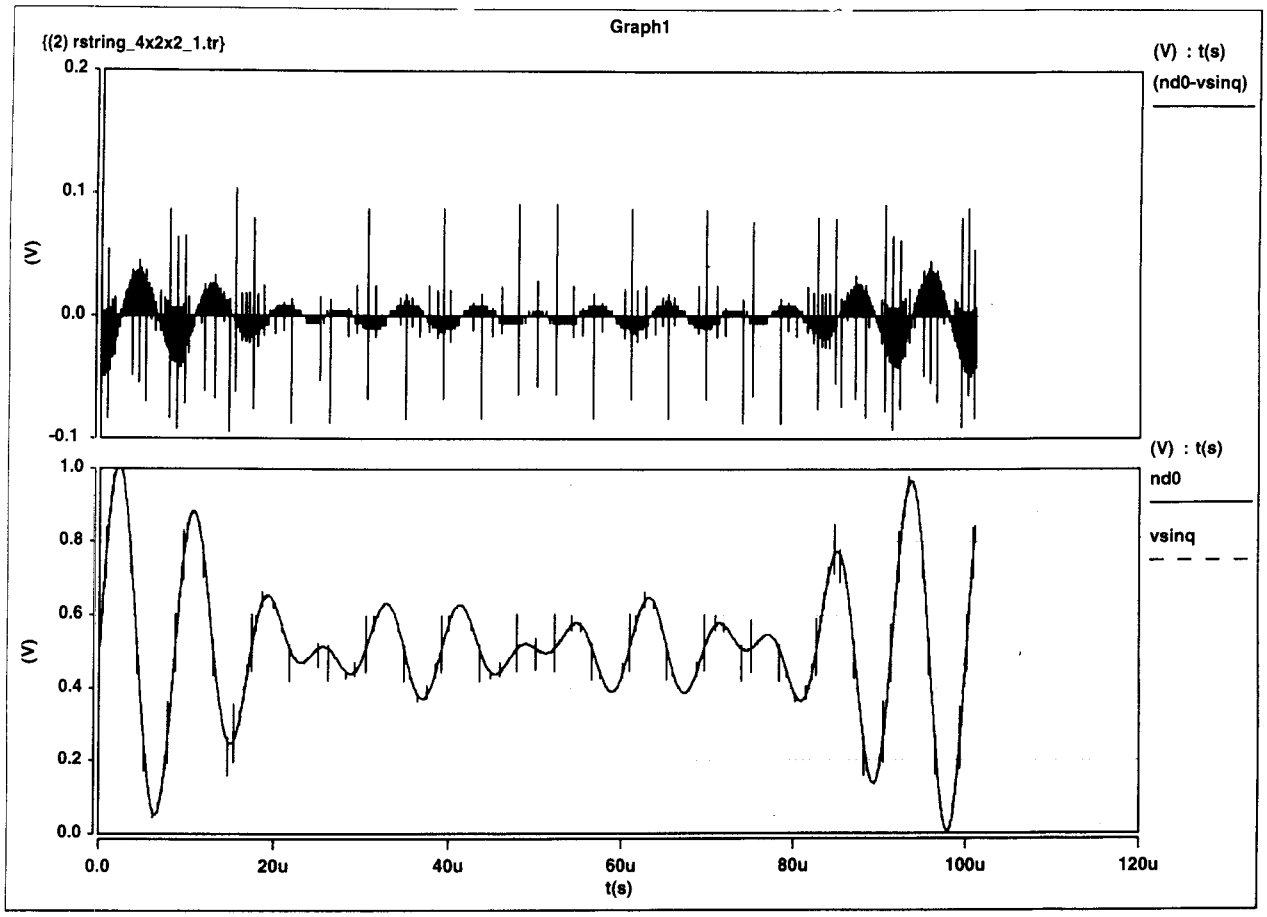
Der Einsatz von analogen Verhaltenspezifikationen hat sich bei der Optimierung von DAC-Strukturen insbesondere deshalb bewährt, da durch die einfache Modellierung nun auch komplexe Signale an 8- und 9-Bit-Wandlern über mehrere Perioden simulierbar sind; eine Simulation mit klassischen strukturellen Modellen versagte hier aufgrund des hohen Aufwandes.

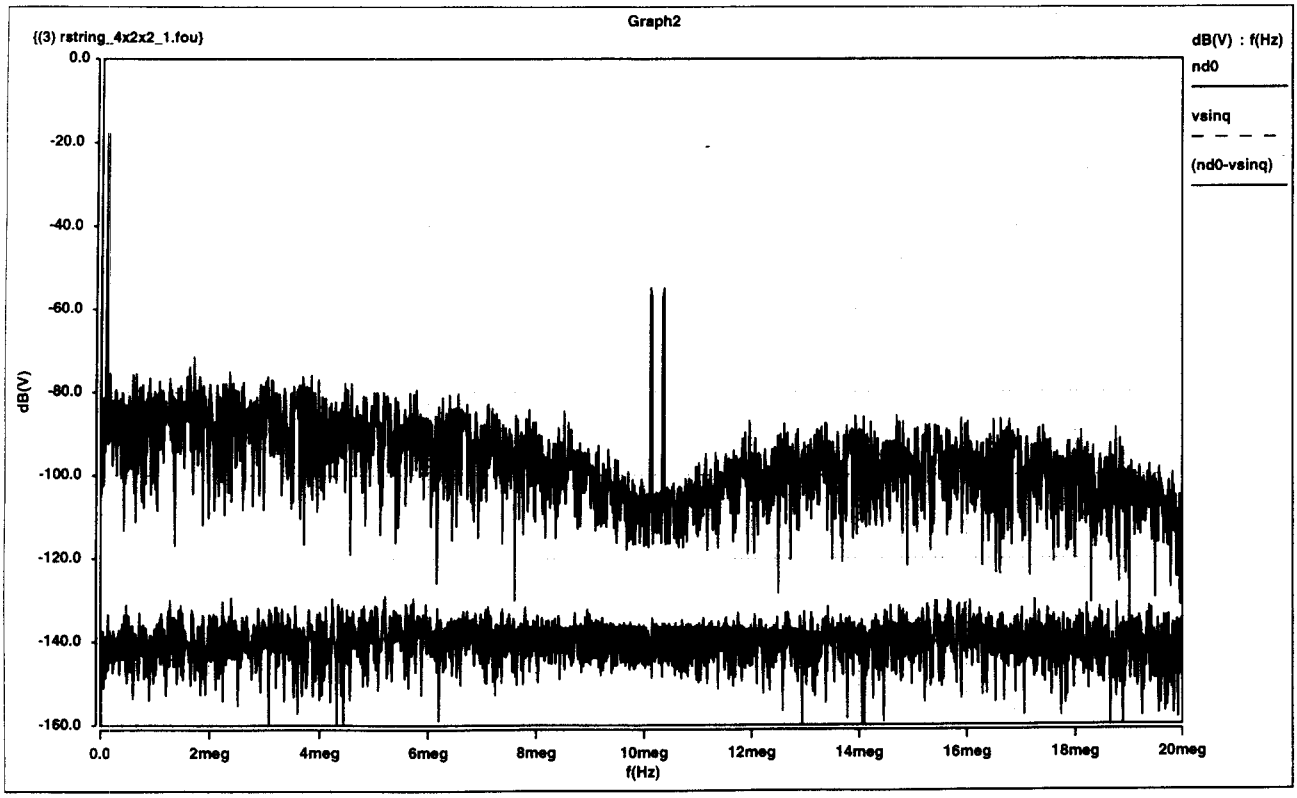
Die verwendete MAST-Modellsprache zeichnete sich in diesem Projekt durch Einfachheit und Effizienz aus, insbesondere der direkte Zugriff auf die Simulator-Datenstrukturen erwies sich als wertvoll. Auf der anderen Seite mußten komplexere Funktionen in MAST teilweise sehr umständlich beschrieben werden; hier wird der neue Standard VHDL-AMS eine interessante Alternative bieten.

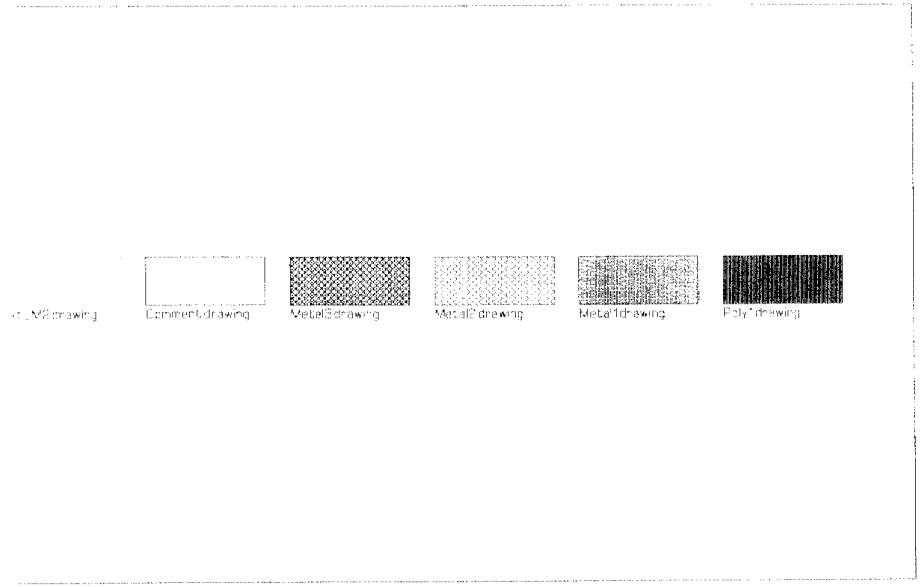
10fF - Lasthap.

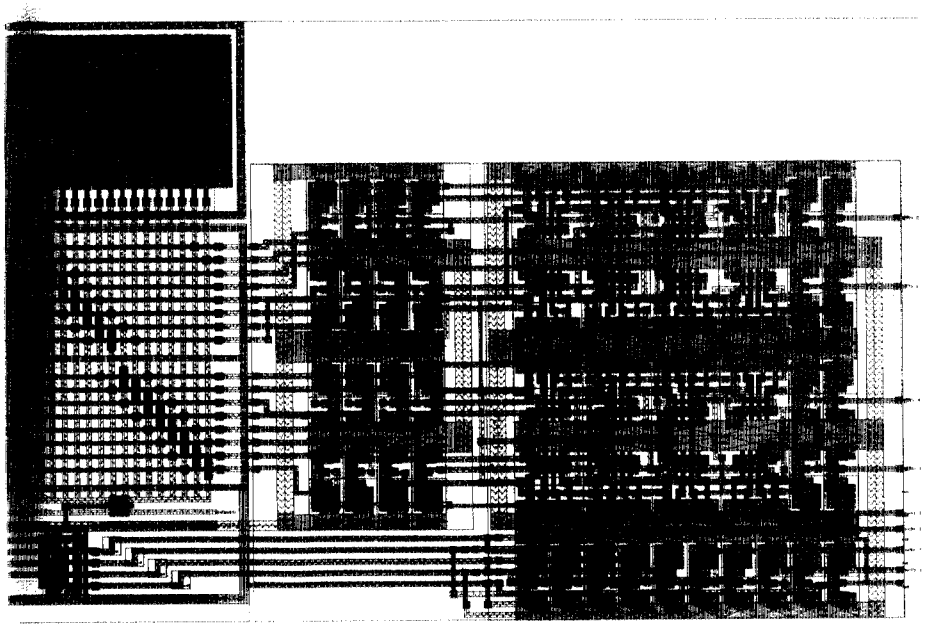





Graph1
{(2) rstring_4x2x2_1.tr}
(V) : t(s)
(nd0-vsinq)
0.2
0.1
0.0
-0.1
(V)
(V) : t(s)
nd0
vsinq
1.0
0.8
0.6
0.4
0.2
0.0
(V)
0.0
20u
40u
60u
80u
100u
120u
t(s)


Graph2
{(3) rstring_4x2x2_1.fou}
dB(V) : f(Hz)
nd0
vsinq
(nd0-vsinq)
0.0
-20.0
-40.0
-60.0
-80.0
-100.0
-120.0
-140.0
-160.0
dB(V)
0.0
2meg
4meg
6meg
8meg
10meg
12meg
14meg
16meg
18meg
20meg
f(Hz)


Contact1drawing
Metal2drawing
Metal1drawing
Poly1drawing



Halbleitertechnologie bei Philips SMST GmbH, Böblingen.

Dipl. Ing. Wolfgang Leicht, Leiter Technologiezentrum, Philips SMST GmbH.

Die Philips SMST GmbH, Böblingen ist ein Hersteller von Halbleitern in Submicron CMOS Technologie. Gebaut in Jahre 1976 als Teil der einstigen IBM Produktionsstätte in Sindelfingen, wurden bis 1995 hauptsächlich Dynamische Speicher (DRAM's) für IBM hergestellt. Nach 5 Jahren als Joint Venture zwischen Philips und IBM ging SMST 1999 vollständig an Philips Semiconductors über und produziert seither nahezu ausschliesslich für Philips Semiconductors.
Seit einigen Jahren werden neben der Standard CMOS Technologie weitere Prozessvarianten entwickelt, die höhere Spannungen im Bereich zwischen 9V und 40V an speziellen I/O Strukturen unterstützen können. Typische Anwendungen sind Treiberschaltungen für LCD's und für Active Matrix Displays.
Die Produktionskapazität von SMST beträgt ca. 225,000 Wafern mit 200mm Durchmesser.

Philips SMST (SubMicron Semiconductor Technologies), ein Teil von Philips Semiconductors gehört zu Philips Royal Electronics; einem weltweit operierenden Elektronik Konzern mit ca. 240000 Mitarbeitern in 60 Ländern und einem Umsatz von 33,9 Mrd. US$ (1998). Die Product Devision S/C gehört zu den grössten Halbleiterherstellern der Welt (Nr.8, 1998) und von Europa mit ca 27000 Mitarbeitern und 17 Produktionsstätten in 11 Ländern. Täglich werden ca. 12 Mill. IC's und 55 Mill. diskrete Bauelemente hergestellt. Über zwei Drittel der Produkte gehen in die Marktsegmente Consumer Electronics und Communications. Das andere Drittel teilt sich auf in Automotive, Industrial und Data processing.
SMST GmbH, Böblingen, liegt ca. 25km südlich von Stuttgart.Die Produktionsstätte wurde 1977 von IBM zur Herstellung von Dynamischen Speicherchips in Betrieb genommen. Im Jahre 1989 wurde hier als erste europäische Halbleiterfabrik auf die Produktion von 200 mm Wafern umgestellt. SMST war von 1995 bis 1998 ein Joint Venture von Philips und IBM zur Produktion von DRAM's, Logik und Embedded DRAM Produkten und ist seit 1999 Teil von Philips Semiconductors. Die steigende Nachfrage nach Halbleitern macht einen Ausbau der heutigen Produktionskapazität notwendig. Zum Jahresende 2000 wird diese 225000 Wafers pro Jahr bei 750 Mitarbeitern betragen. Dies entspricht einem Ausstoss von ca. 300 Mill. chips pro Jahr.
Heute werden an diesem Standort Logik Produkte, Embedded Memories und LCD Treiber Schaltungen hergestellt, in standard CMOS Technologie und High Voltage Prozessvarianten. Ca. 11000 m^2 Produktionsfläche sind in Betrieb, davon 2100 m^2 in Reinraumklasse 1. Die minimal darstellbaren Strukturen sind bei 0,35µm. Planarisierung durch Chemical Mechanical Polish gehört seit vielen Jahren zum Prozess Portfolio. Funktioneller Test, Laser Repair und ein sehr gut ausgestattetes Fehleranalyse Labor unterstützen die Fertigung und werden als Service angeboten.
Parallel zur Weiterentwicklung der Standard CMOS Technologie hat sich SMST auf Prozessvarianten spezialisiert, mit denen Medium bzw. High Voltage tolerante I/O's realisiert werden können. Dies sind Spannungen von grösser 5V bis zu 40V. Die typischen Anwendungen für diese Produkte sind Treiberschaltungen für Liquid Crystal Displays und Treiber für Active Matrix Displays. Eine weitere spezielle Anwendung wird Reflective LCD oder LCoS (Liquid Crystal on Silicon) darstellen. Die höhere Spannungsverträglichkeit wird durch verschiedene Ausführungen der Isolation unterm Transfergate und durch spezielle Formung der N-Well Strukturen erreicht.

Im weiteren sollen einige typische Halbleiterprozesse aus SMST beschrieben werden.
Einer der Standard CMOS Prozesse stellt der 0,5µ Prozess dar. Typische Merkmale sind LOCOS Device Isolierung, 10nm Oxiddicke des Transfergates, Polysilicon Gates, Double Polysilicon Strukturen für analoge Anwendungen in 3V und 5V Ausführung.
Der Prozess hat insgesamt 17 Fotomaskenlagen incl. der 3 Metallisieungslagen, mit CMP planarisiert. Die Verbindung zwischen den Metalllagen wird durch Wolfram ‚plugs' hergestellt. Die ersten Metallage ist 590nm dick und besteht aus einem Sandwich aus TiN / Ti / ALCu (0,5%) / Ti..
Eine weitere Technologie im SMST Portfolio dient der Herstellung von Embedded DRAM's. Es handelt sich dabei um eine Kombination aus ‚Standard DRAM' und ‚Standard Logic'. Die DRAM Basis ist einer der 16Mb Prozesse von IBM erweitert in der Metallisierung um eine Shrink Version des vorne beschriebenen 0,5µ Prozesses. Die Charakteristischen Merkmale sind 0,45µ Technologie, Device Isolierung durch STI (Shallow Trench Isolation Technologie) , DRAM Speicherkapazität durch Deep Trench. Die Fläche einer Speicherzelle beträgt 3,6µm^2 Es werden 19 Fotomaskenlagen bei 3 Metallisieungslagen verwendet. DRAM's sind in Einheiten von 512kb einsetzbar, bis zu einer Gesamtgrösse von 24Mb.

Als letztes Beispiel soll eine Technologie vorgestellt werden, die sich derzeit noch in Entwicklung befindet in SMST. Die dahinter stehende Anwendung ist LCoS oder Liquid Crystal on Silicon. Diese Technologie soll in naher Zukunft die üblichen Transmissive LCD's durch Reflective LCD's ersetzen. Beim Transmissive LCD wird die Pixelstruktur von hinten ‚durchscheint'. Da jedoch die Transistoren zur Ansteuerung der Pixels das Licht abschirmen, wird der transmissive Teil des Pixels mit zunehmender Auflösung immer kleiner, was die Pixelstruktur immer deutlicher sichtbar werden lässt. Aus diesem Grunde bietet sich ein Reflektives Verfahren an. Die Pixel sind Metallstrukturen, die von darunterliegenden Devices angesteuert werden. Kritische Punkte dabei sind die Planarität der ‚Spiegel' und die Medium Voltage Verträglichkeit der Ansteuerung (16V). Die Charakteristik der Technologie ist unterschiedliches Transfergate Oxide für die normale Logik und die Pixel Transistoren, und deep N-Well für MV. Als Abstandhalter des LCD Glases zur Pixelfläche werden Kegelstrukturen aus Nitrid aufgebracht.

Wie wird sich nun die weitere Technologie Entwicklung in SMST gestalten.
Der Weg entlang der Standard CMOS Prozessentwicklung ist über die weitere Reduzierung der min. Fotoabbildungen sehr genau vorgegeben. Die ersten Versuche in 0,3µ Technologie sind dabei bereits angelaufen.
Bei den MV/HV Technologien wird natürlich aus Produktivitätsgründen die Notwendigkeit hin zu kleineren Dimensionen auch bestehen. Die Anforderungen bezüglich höheren Spannungen jedoch werden eher noch höher werden. Als Beispiel sei der Automotive Bereich genannt, wo mit der Einführung das Car Area Networks (CAN) eine breite Tür für 40 – 50V Anwendungen aufgestossen wurde.

Weltweit operierender Electronik Konzern
gegründet 1881 als Lampenfabrik

240000 Mitarbeiter
in 60 Ländern

Headquarters in
Amsterdam, Niederlande

Sales und service Offices in
150 Ländern

Aktien an 16
Aktienmärkten
in 9 Ländern

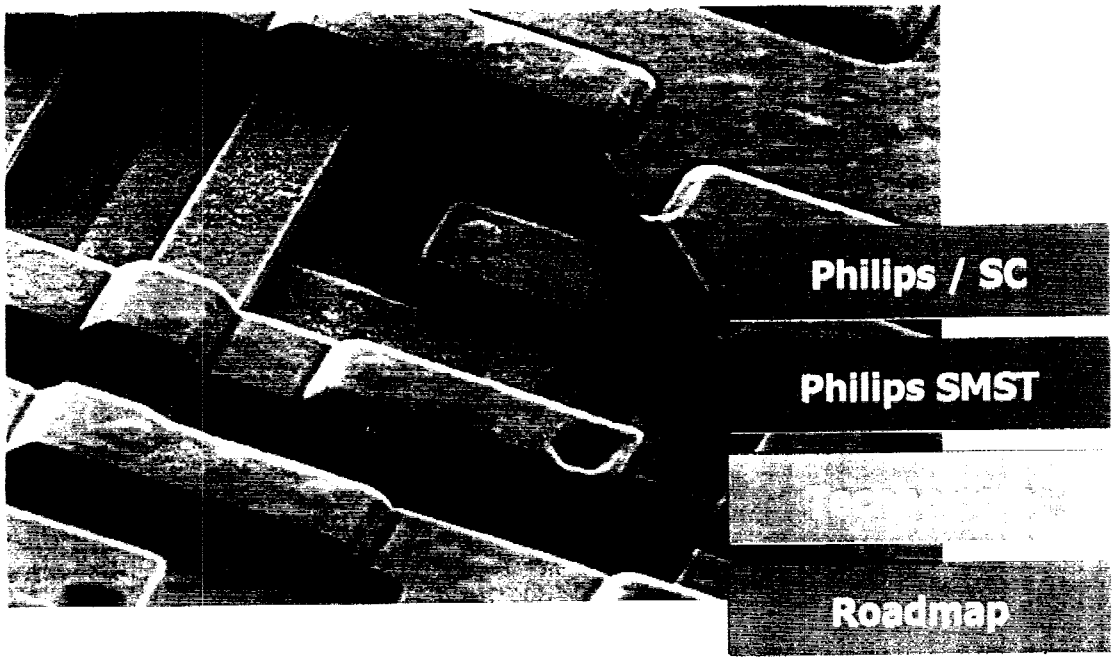
Ergebnisse 1998 (In millions)

	NLG	USD
Sales	67,122	33,900
Net income	13,339	6,876

let's make things better. PHILIPS

Philips Semiconductors SMST GmbH

22. MPC-Workshop in Mannheim am 28.1.2000



let's make things better. PHILIPS

Eine Sparte von Philips Electronics, des achtgrössten Electrionik Konzerns weltweit

1998 achtgrösster Halbleiterhersteller weltweit hinter Intel, NEC, Motorola,...

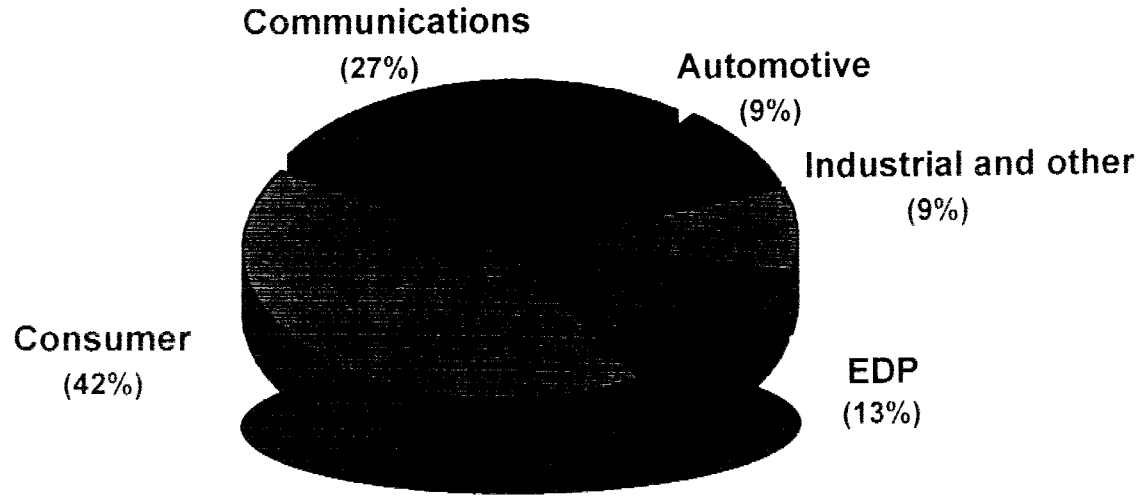
Einer der grössten europäischen Halbleiterhersteller zusammen mit ST, Siemens

27,000 Mitarbeiter in 16 Produktionsstätten in 11 Ländern

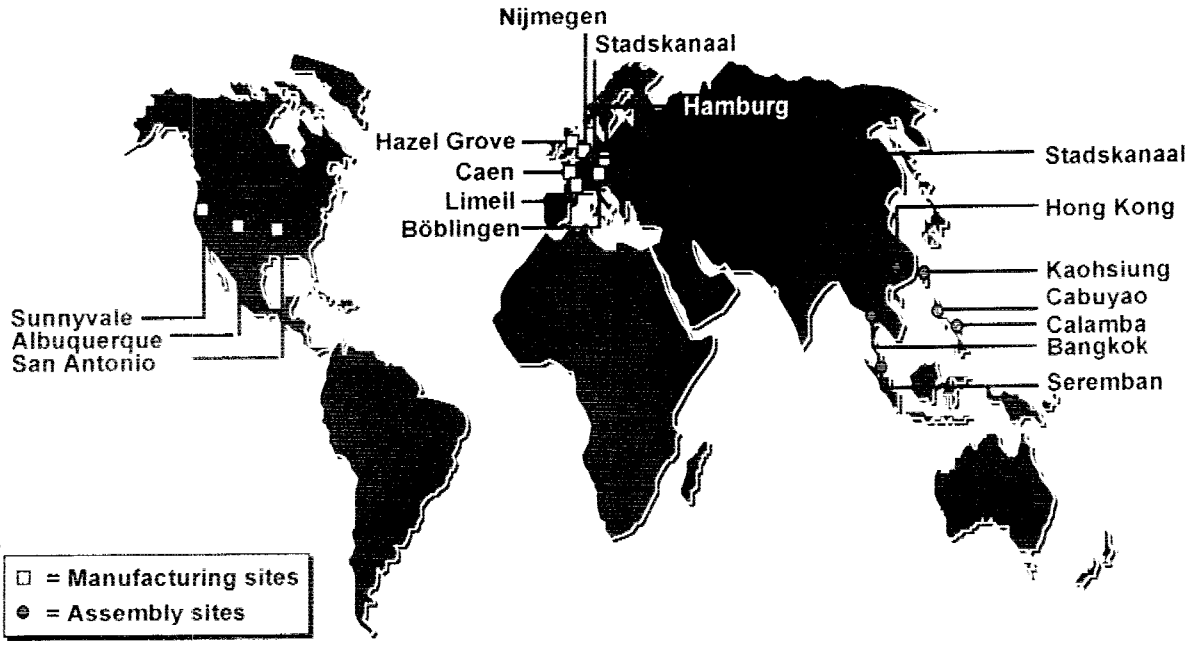
Produktion von 12 Mill. IC's und 55 Mill. descretes täglich

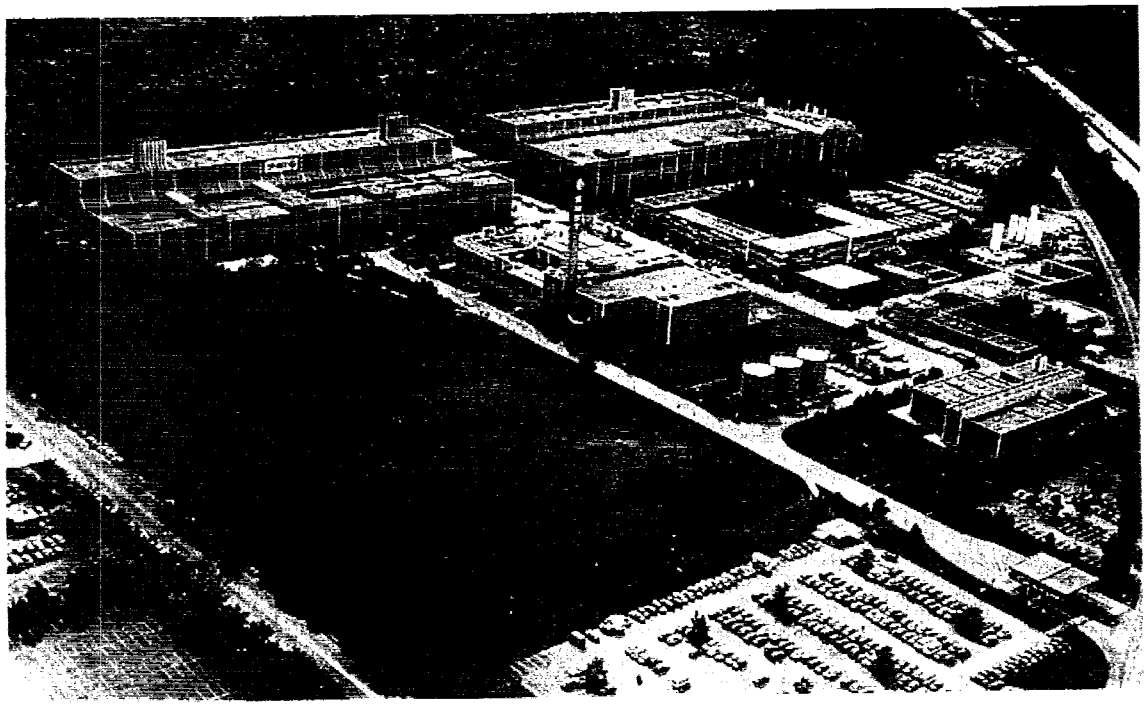
Sales: Integrated circuits und descretes: 4,190 Mill. US$

Total sales 1998 pro Marktsegment:



Produktion & Assembly an 16 Standorten





SMST Steckbrief

- Lokation: SMST Industrial Park Hulb in Böblingen, Germany
- Gegründet: 1. Januar, 1999
- Historie:
  - IBM Halbleiterwerk für DRAMs bis 1995
  - Philips / IBM Joint Venture 1995 - 1998
- Bau:
  1977: Produktionsbeginn
  1989: Umstellung auf 200 mm als erste europäische Fab
- Kapazität: 225.000 Wafers/Jahr in 200mm Wafers (YE 2000),
  ca. 300 Mill. Chips / Jahr
- Operation: 24 h / 7 Tage/Woche / 5-Schicht Modell
- Mitarbeiter: 750 (YE 2000)
- Quality Systems: ISO 9001, QS 9000, ISO 14001

Produktion

- Scope:
  Wafer Produktion für:
  - Logik
  - Embedded Memory
  - LCD Driver

  in Standard CMOS Technologie
  und Medium / High Voltage Derivaten
- Produktionsfläche:

Class 1:	2.100 m²
Class 1.000/10.000:	4.200 m²
Grey Room:	4.300 m²

- Fabricator Characteristics:
  - Litho-Cluster incl. SMIF-Technology (30%)
  - Oxide / Tungsten Chemical Mechanical Polish
  - Infrared Lot Tracking System FiLS
  - Merged Line für alle Technologien

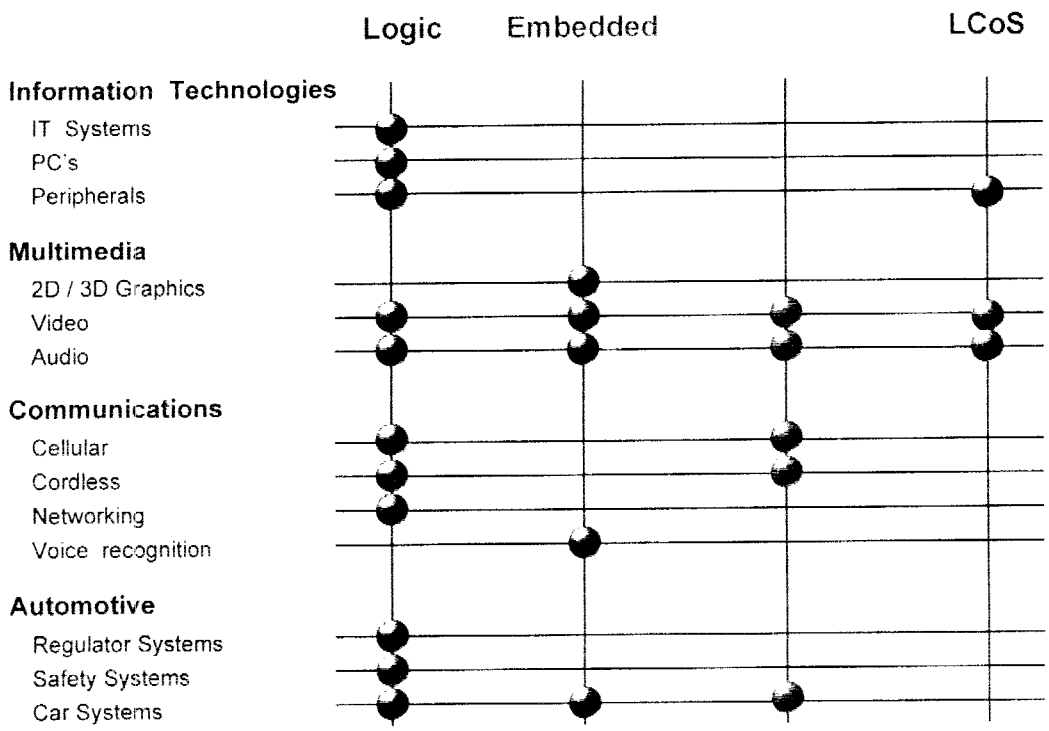
Services

Philips Semiconductors SMST GmbH

- IC Design support
- Chip Finishing
- Mask Making
- Silicon Production
- IC Test
- Assembly
- Reliability Evaluations
- Physical Failure Analysis

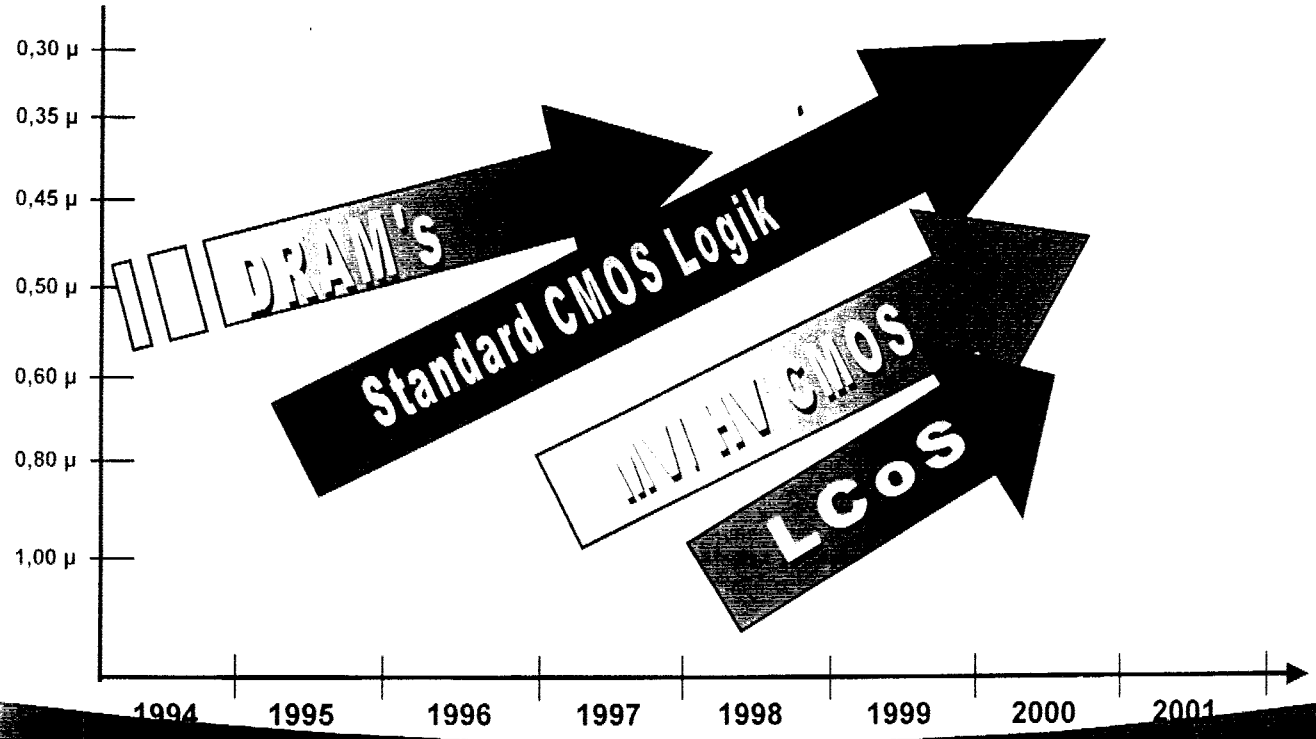
Philips SMST

Applications



Technology History

Philips Semiconductors SMST GmbH

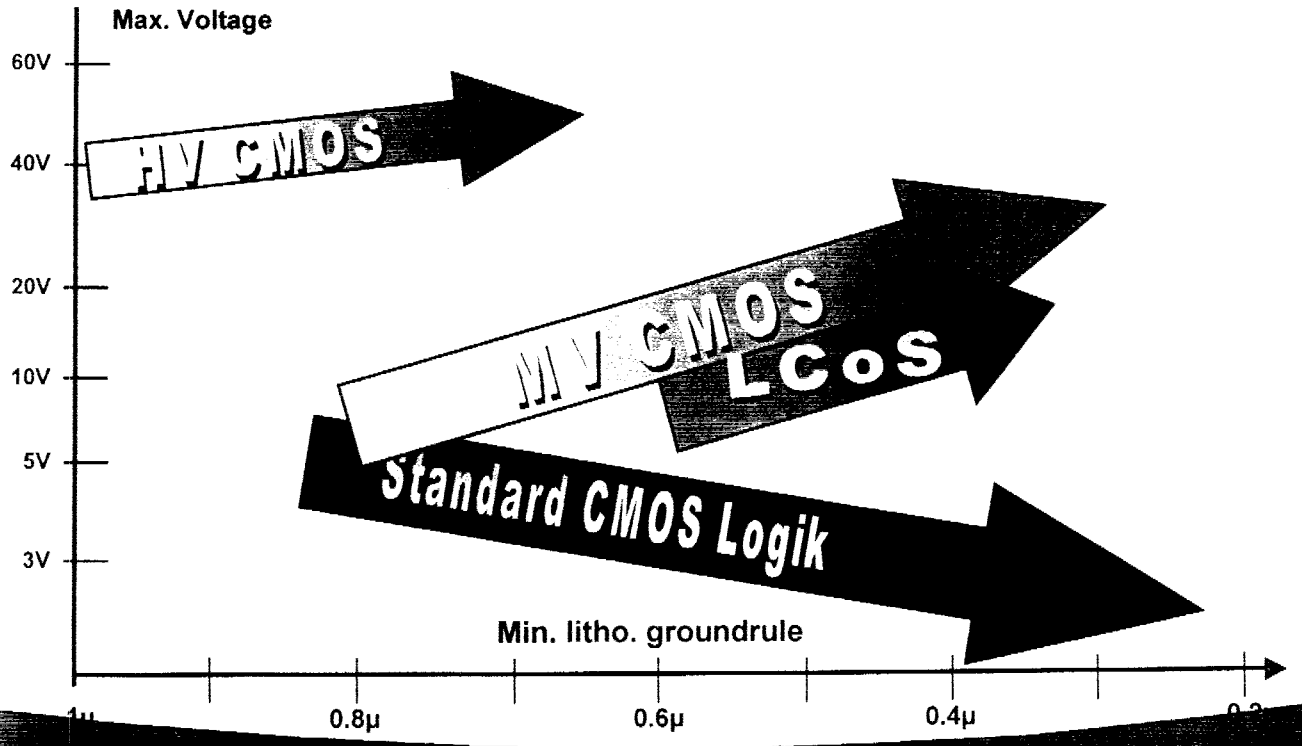


Let's make things better.

PHILIPS

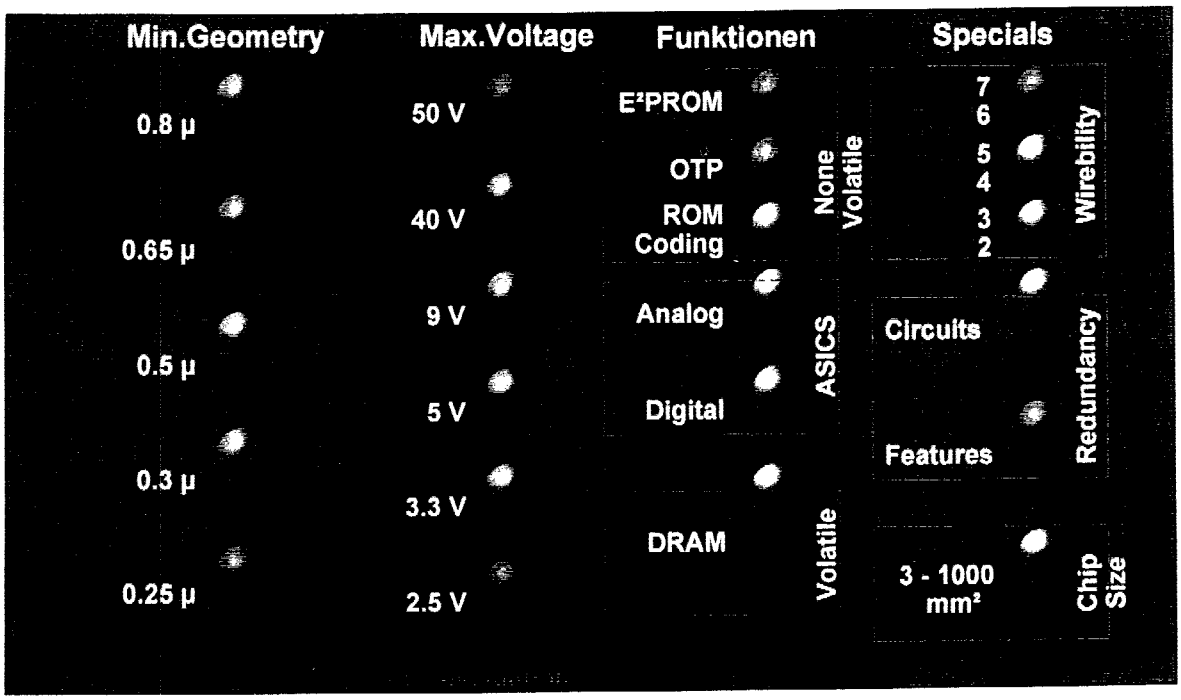
Technologie Portfolio

Philips Semiconductors SMST GmbH



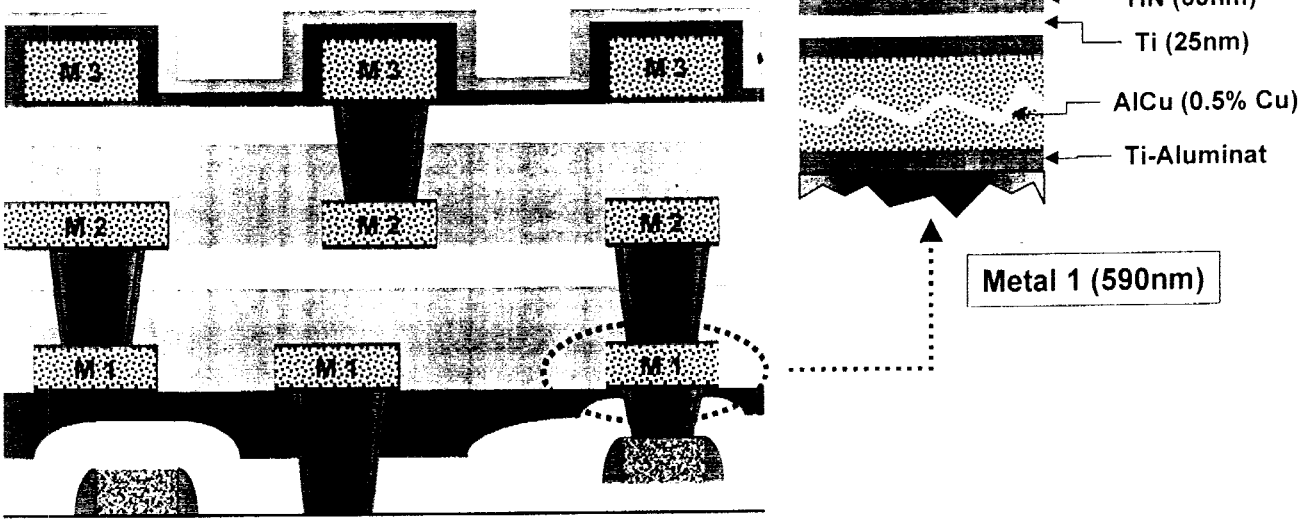
Let's make things better.

PHILIPS

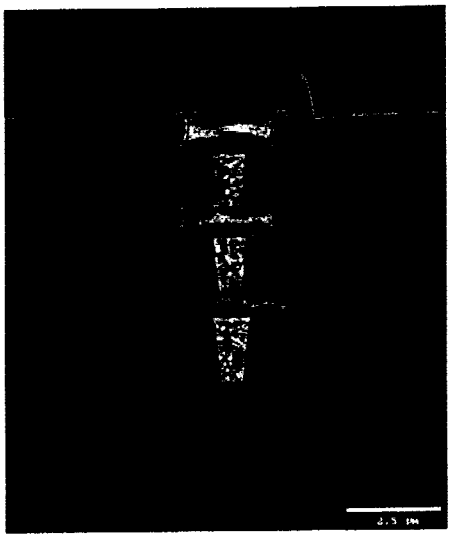


- Prozess Steckbrief:
  - Standard CMOS Prozess
  - 0.5 µm Logik Designs
  - LOCOS Device Isolation, 10nm Gate Oxide
  - Single / Double Polysilizium für Digital / Analog Anwendungen
  - Standard Version für 3V und 5V
  - 17 Maskenlagen
  - 3 Metalllagen
  - Planarisierung (CMP) für Device Passivierung und Interlevel dielectrics
  - „Stacked Via"-Technologie

Passivierung aus Nitrid / Oxid oder Polyimid



- **CMP Planarisierung:**

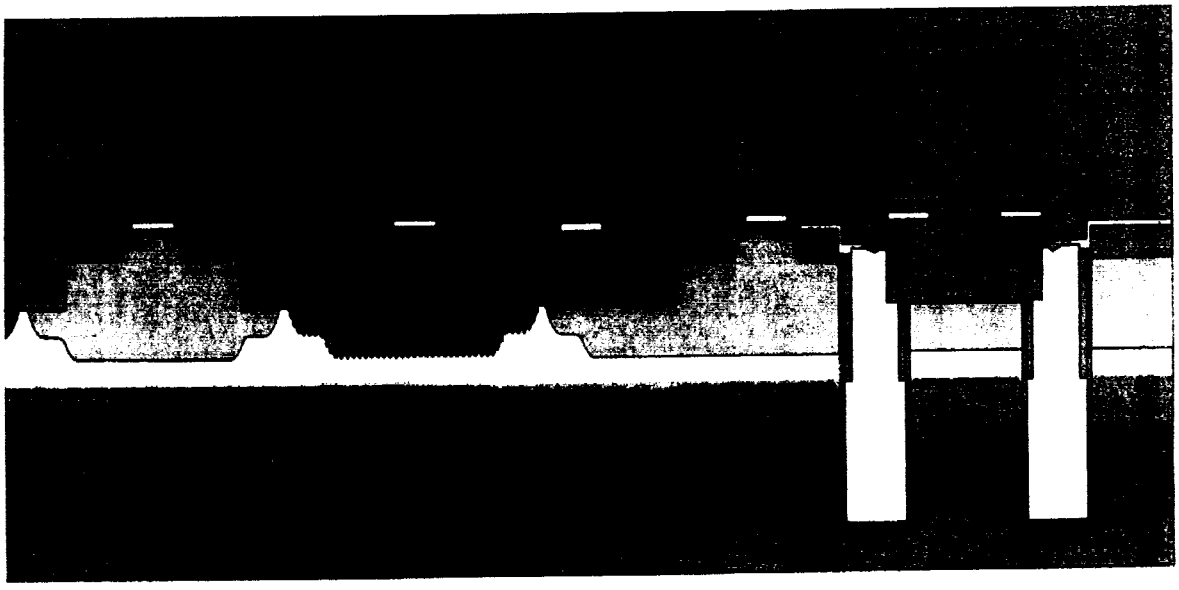


C100 seal ring (3 Metalllagen)

- Prozess Steckbrief:
  - CMOS Prozess zur Herstellung von Embedded Memory circuits.
  - Kombination aus Standard Logik und Standard DRAM Prozess
  - 0.45 µm Technologie
  - Shallow Trench Isolation (STI), 10nm Gate Oxide
  - Deep Trench Speicherkapazität im DRAM
  - 19 Maskenlagen, 3 Metalllagen
  - 512 KB - 24 MB DRAM size, skalierbar in Schritten von 512KB
  - DRAM cell size: 3.6 µm²

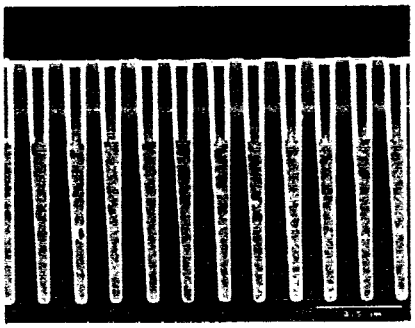
DRAM5L1
Philips Semiconductors SMST GmbH

- Technologie Schichtaufbau:

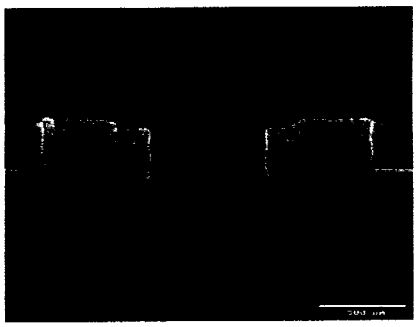


Let's make things better.
PHILIPS

- Deep Trench DRAM storage node:



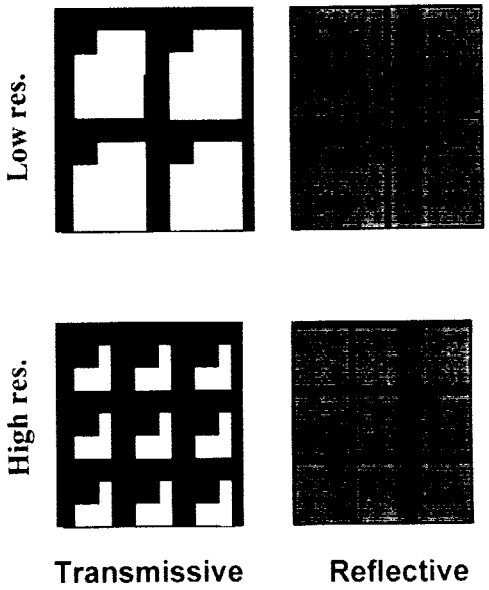
- Shallow Trench Isolation:

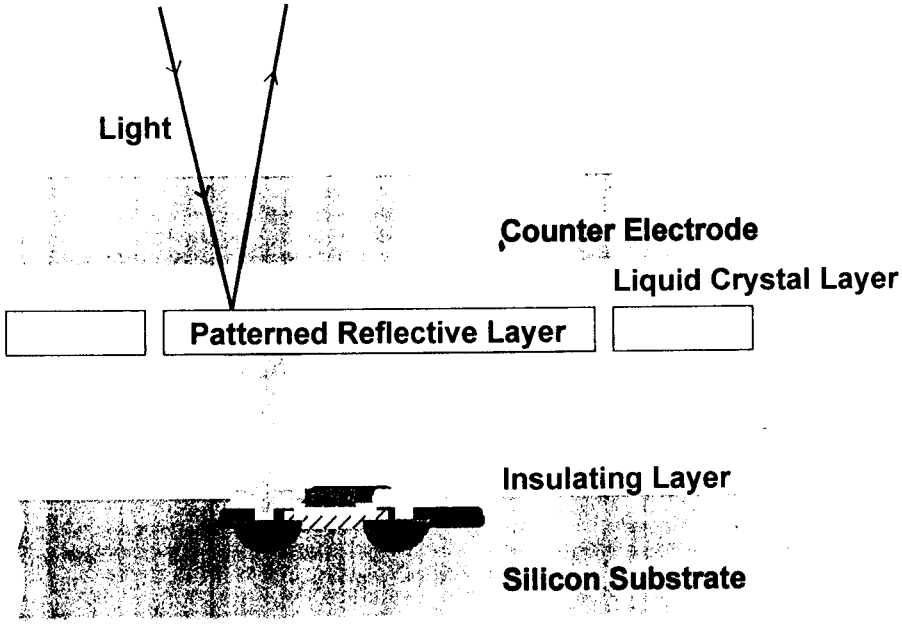


Liquid Crystal on Silicon (L C o S)

Vorteile gegenüber Transmissive LCD:

- Pixel Structure nicht sichtbar
- Höchste Auflösung möglich ohne Lichtverlust
- Single panel möglich
- Einfache Integration von Logikfunktionen





LCoS
Philips Semiconductors SMST GmbH

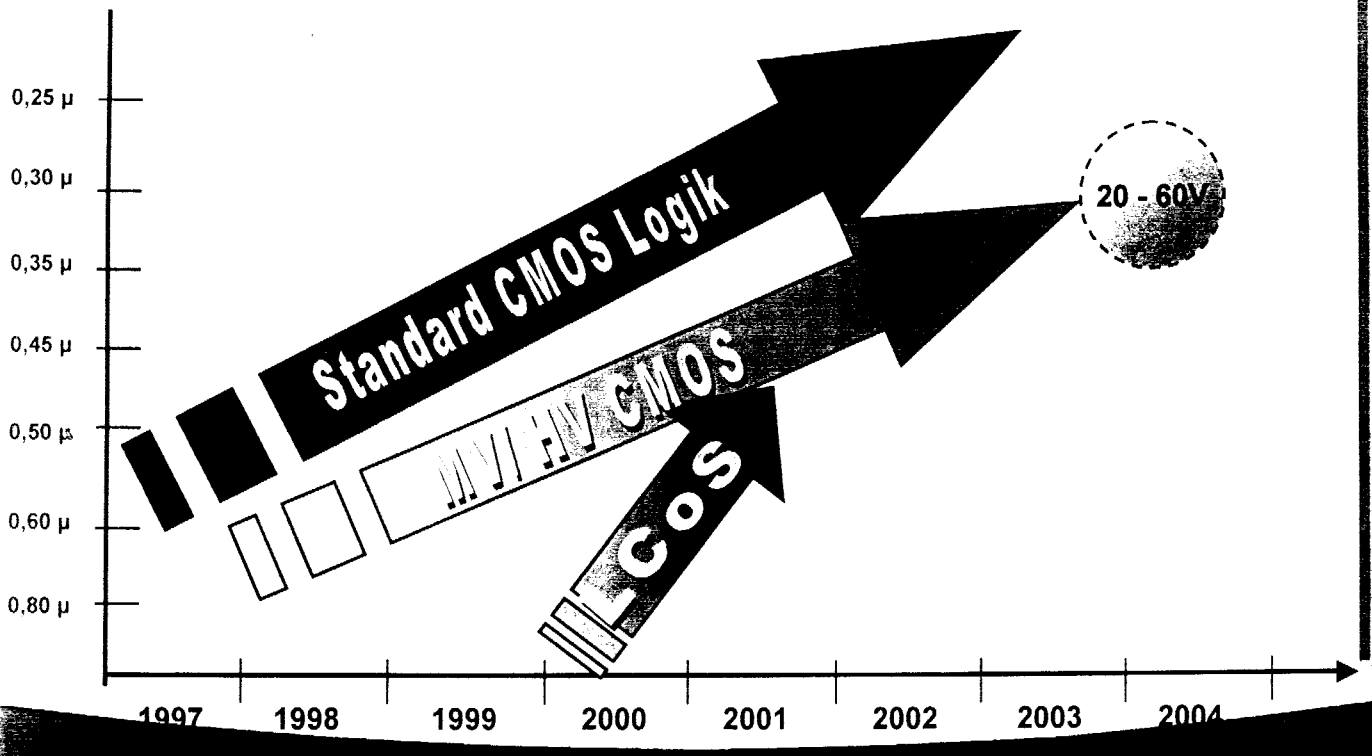
- **Prozess Steckbrief:**
  - **Medium Voltage CMOS Prozess**
  - **0,8µm LCoS Designs mit 20V Ausgang für Pixeltreiber**
  - **LOCOS Device Isolation, 15 nm / 40 nm Dual Gate Oxide**
  - **Deep Nwell für MV Devices**
  - **22 Maskenlagen**
  - **4 Metalllagen**
  - **Operating Voltage: 5V / 16V**
  - **Pixel Kondensator aus Metall / Substrat**

MPC Mannheim 28.1.2000 FHmannheim1.ppt / wl

Let's make things better.
PHILIPS

Technology Roadmap

Philips Semiconductors SMST GmbH



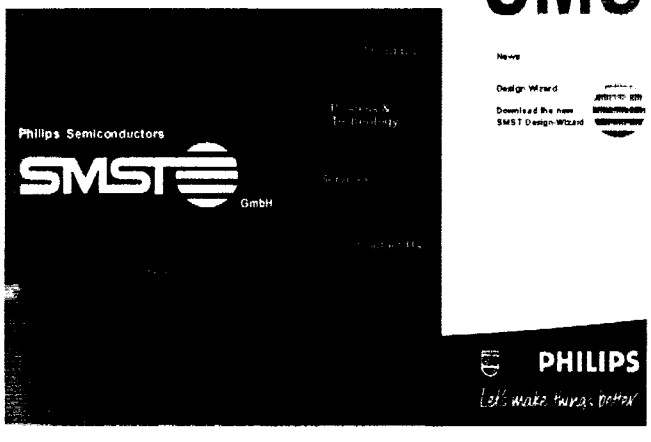
Let's make things better.

PHILIPS

Internet Home Page

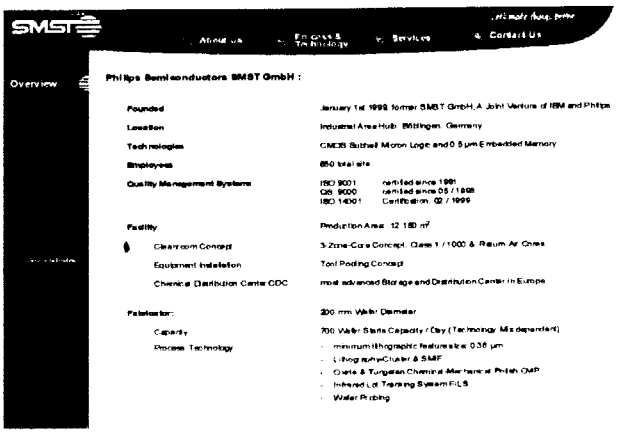
Philips Semiconductors SMST GmbH

SMST on the web:



Aktuelle Informationen über
Philips Semiconductors SMST GmbH unter:

http://www.smst.philips.com



Let's make things better.

PHILIPS

PCI-Bus Anbindung mit einem IP-CORE

Alexander Wink
DaimlerChrysler Aerospace AG, Wörthstraße 85, 89077 Ulm
Fachhochschule Ulm, Prittwitzstraße 10, 89075 Ulm

Zusammenfassung
Im Rahmen einer Diplomarbeit wurde bei der Firma DaimlerChrysler Aerospace AG in Ulm ein PCI Interface in einem FPGA entwickelt. Dabei wurde ein IP-CORE der Firma XILINX verwendet. Ein im Interface verwendeter FIFO wurde einer Referenzimplementierung von XILINX entnommen und an die eigenen Erfordernisse angepaßt. Realisiert wurde das Interface unter der Verwendung der Tools: Renoir, ModelSim, Leonardo Spectrum und der XILINX Alliance Software auf einem FPGA XC4028XLT-09 HQ240.

1 Aufgabenstellung

Im Rahmen einer Diplomarbeit bei der Firma DaimlerChrysler Aerospace in Ulm sollte ein Compact PCI Interface für einen Parameterextraktor entwickelt werden. Ein Parameterextraktor ermittelt aus einem empfangenen Radar Signal bestimmte Eigenschaften wie: Frequenz, Amplitude, Pulsdauer und andere Parameter. Die Steuerung erfolgt durch einen Power PC. Abbildung 1 zeigt eine vereinfachte Übersicht des Systems. Das PCI-Interface sollte in einem FPGA XC4028XLT der Firma XILINX integriert werden. Dabei wurde ein IP-CORE mit der Bezeichnung: „LogiCORE PCI 32" in der Version 2.0 von XILINX verwendet. Der im Interface verwendete FIFO wurde einer Referenzimplementierung entnommen und den eigenen Erfordernissen angepasst. Für die Arbeiten standen folgende Tools zur Verfügung:
- Renoir 99.1: Erstellung der Blockschaltbilder
- ModelSim5.2e: Simulation
- Leonardo Spectrum 1999.1e: Synthese
- XILINX Alliance 1.5i: Place and Route

Das PCI-Interface enthält 2 Adressbereiche. Einen I/O-Bereich der Größe 1kB für die Konfiguration des Parameterextraktors und einen Memory-Bereich der Größe 2MB für die Schnelle Übertragung der dort erzeugten Daten.

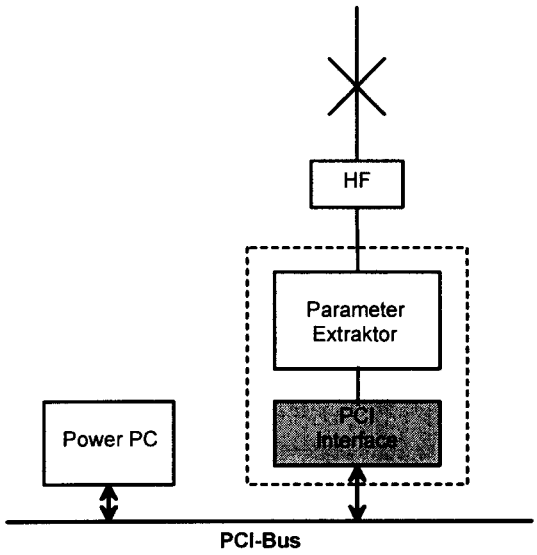


Abbildung 1: Systemübersicht

2 Eigenschaften des PCI-Busses

Der verwendete Compact-PCI-Bus ist eine Erweiterung des Standard PCI-Busses, wie er vor allem in aktuellen PCs zum Einsatz kommt. Für die Verwendung im industriellen Bereich werden verbesserte Stecker und eine Leiterkarte im Europa Format eingesetzt. Die elektrischen Eigenschaften und das Protokoll entsprechen dem normalen PCI Standard.
Der PCI-Bus hat eine Breite von 32 Bit und arbeitet mit einer Taktfrequenz von 33 MHz. Erweiterungen für 64 Bit und 66 MHz sind ebenfalls definiert. Er zeichnet sich durch eine hohe Datenübertragungsrate von 132 MB/s bei 32 Bit und 33 MHz aus. Der Bus existiert in zwei Ausführungen, für 3,3V und 5V Signalspannung. Die Übertragung erfolgt synchron mit der steigenden Taktflanke. Der Bus ist Multimasterfähig, das bedeutet es können mehrere Master auf einem Bus arbeiten. Master können Adressen auf den Bus legen und Transfers initiieren. Die Arbitrierung geschieht im

Hintergrund durch einen zentralen Arbiter. Es stehen drei Adressräume zur Verfügung. Dies sind: ein Konfigurations-Adressraum, der in jedem PCI-Device vorhanden ist. In ihm befinden sich unter anderem bis zu 6 Basis-Adress-Register, in denen je ein I/O- oder Memory-Bereich dekodiert wird. Jeder dieser Bereiche kann bis zu 2GB groß sein. In Intel Rechnern ist aus Kompatibilitätsgründen der I/O-Bereich auf 256 Byte begrenzt. Die Mindestgröße liegt im I/O-Bereich bei 16 Bytes und im Memory-Bereich bei 4 kB.
Der PCI-Bus ist nicht terminiert, sondern er arbeitet mit den Reflektionen. Es wird sog. „Reflected Wave Switching" benutzt. Das bedeutet, im ersten Moment wird nur der halbe Signalhub auf den Bus eingekoppelt. Nach der Totalreflexion am Busende, wird erst mit der rücklaufenden Welle der volle Signalhub erreicht. Deshalb sind für PCI-Signale spezielle I/O-Puffer nötig.
Da die Adress- und Datenleitungen gemultiplext werden, kommt der PCI-Bus mit nur 47 Signalen für ein Traget Device aus. Ein Master hat zwei zusätzliche Leitungen für die Arbitrierung. Der PCI-Bus ist Burst fähig, das bedeutet nach jeder Adressübertragung können mehrere Datenphasen folgen. Erst dadurch kann der Bus sehr effizient genutzt werden. Ein optimaler Schreibzugriff erfolgt durch einen 3-1-1-1 Burst. Das bedeutet, für die Übertragung des ersten Doppelwortes werden 3 Takte, für jedes folgende Doppelwort wird nur noch ein Takt benötigt. Die 3 Takte setzen sich wie folgt zusammen. Zwischen jeder Übertragung auf dem Bus ist ein Wartetakt vorgeschrieben. Danach wird die Adresse und anschließend daran ein oder mehrere Daten übertragen (W, A, D, D, D, ...). Bei Lesezugriffen wird zwischen der Adresse und den Daten ein weitere Wartetakt eingefügt, so daß ein 4-1-1-1 (W, A, W, D, D, D, ...) Zugriff entsteht. Dieser sogenannte Turn Around Cycle ist nötig, weil die Adresse vom Master auf den Bus gelegt wird, die Daten aber vom Target geliefert werden und sich sonst überschneidende Zugriffe auf dem Bus ergeben könnten. Die genauen Eigenschaften des (Compact) PCI – Busses sind in [1] und [2] zu finden.

3 PCI Interface

Abbildung 2 zeigt das Blockschaltbild des gesamten PCI-Interfaces, das im FPGA realisiert wurde. Auf der linken Seite befindet sich der PCI-Bus und auf der rechten Seite ist die Verbindung zum Parameterextraktor (Backend) angeordnet. Die weiß dargestellten Blöcke sind selbst programmiert, die anderen ganz oder Teilweise von XILINX übernommen.
Der Block „State Machine" steuert das Timing zum Backend und gibt direkt die Signale Write Enable und

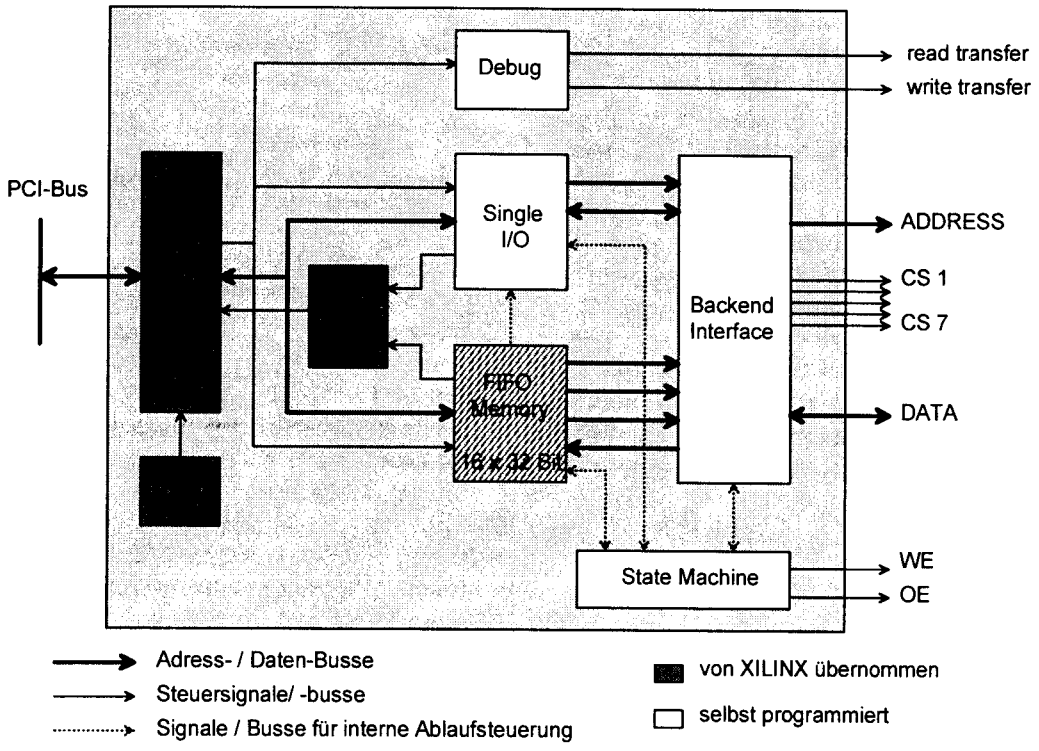


Abbildung 2: Blockschaltbild PCI Interface

Output Enable aus. Der Block „Single" ist für die Datentransfers im I/O-Bereich zuständig und unterstützt nur Einzelzugriffe. Da über den I/O-Bereich der Parameterextraktor konfiguriert wird, treten im Normalbetrieb nur sehr geringe Datenraten auf. Somit ist ein FIFO für diesen Bereich nicht nötig.
Mit dem Block „FIFO" werden die Transfers im Memory Bereich bearbeitet. Dieser Bock wurde aus einer Referenzimplementierung von XILINX übernommen und angepaßt. Er hat getrennte Adresss- und Datenbusse für die Schreib- und Leserichtung. Um diese Busse und die Busse des Blocks „Single" zusammenzufassen dient der Block „Backend Interface". Er enthält neben Multiplexern für Daten und Adressen auch einen Zähler, da der FIFO während eines Burst Transfers nur die Startadresse ausgibt. Alle weiteren Adressen müssen selbst erzeugt werden. Weiterhin werden in diesem Block auch die Chip Select Signale für das Backend erzeugt.
Der Block „Debug" dient zur Fehlersuche und zeigt über zwei Ausgänge an, ob Schreib oder Lesetransfers, die für das Interface bestimmt sind, im CORE dekodiert werden.

4 XILINX LogiCORE PCI32

Das gepackte Archiv des PCI-CORES enthält unter anderem:
- Anleitung
- Netzliste für PCI-CORE
- Simulationsmodell des COREs
- Testbench
- Skripte für Synthese und Place & Route
- Beispiel für PCI-Interface

Das Simulationsmodell des COREs besteht aus zwei VHDL Files, einem Wrapper File und dem eigentlichen Simulationsmodell, das durch den Wrapper instanziert wird.
Der PCI-CORE „LogiCORE PCI32" in der Version 2.0 unterstützt 7 Bausteine aus der XC4000XLT Serie von XILINX. Einige davon nur im One Waitstate Mode. Das bedeutet, daß während eines Bursts zwischen jedem übertragenen Doppelwort ein Wartetakt eingefügt wird. Dies ergibt einen 6-2-2-2 Burst. (W, A, W, W, W, D, W, D, W, ...). Die in der Diplomarbeit verwendete Version unterstützt Write Bursts mit 5-1-1-1 und Read Bursts mit 6-1-1-1. Die drei Wartetakte beim Übertragen des ersten Doppelwortes werden von der State Machine im CORE verursacht und können nicht verhindert werden.
Bei Transfers kann nur die Übertragung des ersten Doppelwortes durch einfügen von weiteren Wartetakten verzögert werden, alle nachfolgenden Daten werden mit 132MB/s übertragen und müssen mit dieser Geschwindigkeit geliefert oder abgeholt werden. Sollte das nicht möglich sein, darf man entweder nur Einzeltransfers zulassen, oder es muß ein FIFO eingesetzt werden. Letzteres ist zur Erhaltung der Performance zu bevorzugen.
Der PCI-CORE enthält alles das, was jedes PCI-Device zur Verfügung stellen muß. Siehe Abbildung 3. Er generiert und überprüft automatisch die Parität auf dem Bus. Er enthält bis zu 3 Adressdekoder und den kompletten Konfigurations-Header. Für die Ablaufsteuerung von Target Transfers ist eine Target State Machine enthalten. In der Version des PCI-COREs, die selbst Transfers initiieren kann ist eine Initiator State Machine integriert. Die Konfiguration des COREs geschieht über ein externes VHDL-File, das vor dem Download aus dem Internet direkt bei XILINX auf der Homepage individuell für das benötigte Design erstellt wird.
Durch den PCI-CORE selbst werden alle Signale vom PCI-Bus zum Benutzer Interface und in der umgekehrten Richtung um einen Takt verzögert.

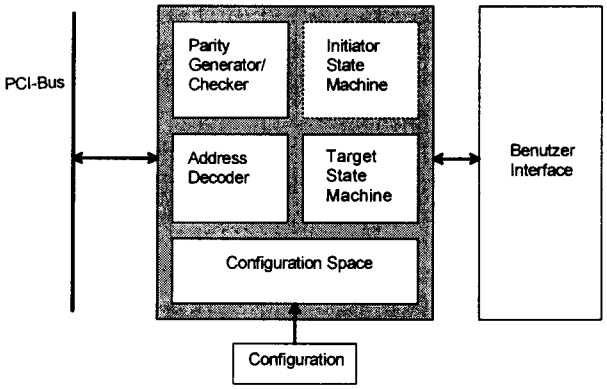


Abbildung 3: XILINX LogiCORE PCI32

5 PCI-Bus Bandbreite

Die Datenübertragungsrate auf dem PCI-Bus entspricht nicht immer den überall angegebenen 132 MB/s. Vielmehr ist das ein theoretischer Wert, der nur bei unendlich langen Bursts erreicht wird. In der Praxis stellt sich selten ein Wert höher als 80MB/s ein. Dies kommt vor allem dadurch zustande, daß die Übertragungen nur aus kurzen Bursts oder Einzeltransfers bestehen. Abbildung 4 zeigt den Zusammenhang zwischen der Burstlänge und der PCI-

Bandbreite bei optimalen Transfers, und bei Transfers mit dem PCI-CORE.

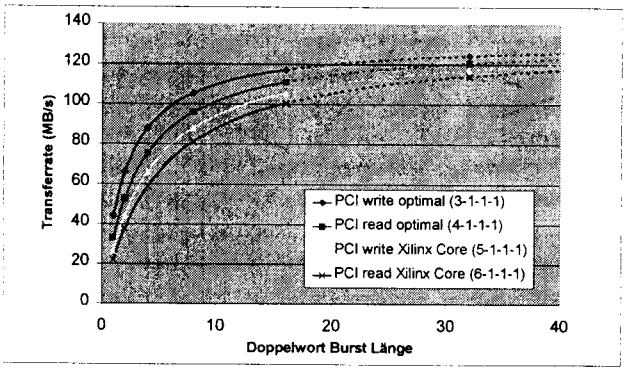
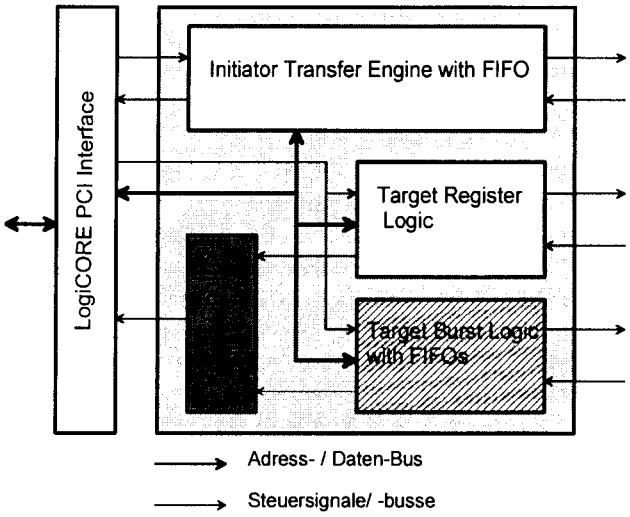


Abbildung 4: PCI-Bus Bandbreite

Die unterste Kurve stellt die Bandbreite bei Read Transfers mit dem PC-CORE dar. Wie zu sehen ist, erreicht man in diesem Fall mit Einzeltransfers nur etwas mehr als 20MB/s, wobei der Bus aber voll belegt ist. Bei länger werdenden Transfers nähert sich die Geschwindigkeit 132 MB/s. Bei einer Transferlänge von 16 Doppelworten, werden schon über 100 MB/s erreicht. Dies ist die maximale Länge, die der FIFO in diesem Design erlaubt. Bei einer Burstlänge von 512 Doppelworten, oder 2kB werden schon mehr als 130 MB/s auf dem Bus erreicht. Bei einem 6-2-2-2 Burst nähert sich die Übertragungsrate nur 66MB/s. Man sollte einen Baustein der nur solche Bursts zulässt nicht für Anwendungen einsetzen, in denen eine hohe Übertragungsrate gefordert wird, außerdem wird der Bus auch für andere Devices unnötig blockiert.

6 Synthesizable PCI Bridge Design

Um bei einem neuen Design nicht ganz von vorne beginnen zu müssen ist von XILINX ein Referenzdesign erhältlich, das neben einer FIFO Implementierung auch häufig verwendete Register für verschiedene Anwendungen enthält. Ein grobes Blockschaltbild ist in Abbildung 5 zu sehen.
Daraus wurden die Blöcke „Target Control Multiplexer" und „Target Burst Logic with FIFOs" in das eigene Design übernommen. Der Target FIFO musste an mehreren Stellen angepasst werden. z. B.: wurde der Adressbereich von 64 kB auf 2 MB erweitert. Der Multiplexer wurde unverändert übernommen. Er hat zwei Aufgaben. Erstens soll sicher gestellt werden, daß die Steuersignale der Nachfolgenden Blöcke (Target FIFO und Target Register) nur exklusiv auf den PCI-CORE durchgeleitet werden. Bei Einsatz einer einfachen ODER Verknüpfung, könnten Signale vom gera-



de nicht angesprochenen Block die Übertragung des anderen stören. Die Fehlersuche wird dadurch unnötig

Abbildung 5: Synthesizable PCI Bridge Design

erschwert. Weiterhin durchlaufen die Steuersignale des PCI-COREs mehrere Ebenen Kombinatorik und sollten unbedingt direkt von Flip-Flops angesteuert werden. Diese Flip-Flops sind im Block Multiplexer bereits enthalten. Dadurch ergibt sich eine zusätzliche Verzögerung der Steuersignale um einen Takt. Die Kombination aus Multiplexer und FIFO führt durch die Summe der Verzögerungen sowohl bei Schreib-, als auch bei Lesezugriffen zu einem 6-1-1-1 Burst.

7 Probleme

Um VHDL Code in Renoir als Blockdiagramm darzustellen kann die Funktion „Convert HDL to Graphics" verwendet werden. Hierbei wird aus der Entity ein Component erzeugt, welches in eigene Designs eingebunden werden kann. Beim Einlesen des Synthesizable PCI Bridge Designs sollten alle benötigten VHDL Files gleichzeitig eingelesen werden, da sonst Probleme bei der Instanzierung auftreten.
Weiterhin erkennt diese Funktion die Direktive „--synopsis translate_off" nicht als solche, sondern betrachtet sie als normalen Kommentar. Das führt dazu, daß sie nicht immer richtig in den von Renoir generierten VHDL Code übernommen wird. Für Leonardo Spectrum wird mit „--synopsis translate_off" und „--synopsis translate_on" Code markiert, der nicht synthetisiert werden soll. Diese Direktiven sind im Code des Wrapper-Files des PCI COREs und der RAM Beschreibung des FIFOs enthalten.

Beim Einlesen der Files des PCI-COREs sollte die Option „Instance Limit" aktiviert werden. Dadurch wird die Strukturbeschreibung des Simulationsmodells als reiner VHDL-Code importiert, und nicht für jedes Instanzierte Objekt ein eigener Block erstellt. Die Strukturbeschreibung enthält mehrere tausend Components und würde ein völlig undurchschaubares Blockdiagramm ergeben. Weiterhin würde dieser Vorgang mehr als eine Stunde dauern. Während des Konvertierens werden für die Components der Strukturbeschreibung Dummy Gatter erzeugt, die gleich gelöscht werden sollten. z.B.: X_AND2, X_AND3, usw.

Ein weiteres Problem ergab sich daraus, daß der PCI-Bus je Takt bis zu 4 Bytes gleichzeitig übertragen. Welche Bytes davon gültig sind, wird über vier Byte Enable Leitungen signalisiert. Es sind dabei alle möglichen Kombinationen erlaubt, z.B.: Byte 1 und Byte 3 gültig. Siehe Abbildung 6.

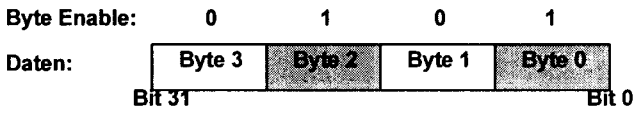


Abbildung 6: Byte Enable

Da auf der Verbindung zum Parameterextraktor nur ein Write Enable vorhanden ist, müssen Schreibzugriffe die nicht alle vier Byte Enable Signale gleich ansteuern, zurückgewiesen werden. Durch die Verzögerungen im PCI-CORE und im „Target Control Multiplexer" kann ein unerlaubter Zugriff erst drei Takte nach Auftreten auf dem Bus abgebrochen werden. Bei Einzelzugriffen ist das kein Problem, da der erste Transfer sowieso schon auf 6 Takte verzögert wird, und die Überprüfung keine zusätzlichen Wartetakte erzeugt. Während eines Bursts müssten aber alle folgenden Transfers um drei Takte verzögert werden. Es würde also ein 6-4-4-4 Burst entstehen. Dies wäre ein sehr großer Performance Verlust. Weiterhin kann der PCI-CORE, wie schon weiter oben beschrieben, während des Burst-Transfers keine Wartetakte erzeugen. Daher blieb bei diesem Design kein anderer Ausweg, als bei Schreibzugriffen im Memory Bereich die einzelnen Transfers nicht zu verzögern, sondern erst drei Takte nach Auftreten eines unerlaubten Zugriffs den Burst abzubrechen. Dadurch kann zwar nicht der fehlerhafte Zugriff verhindert werden, aber drei Takte später kann der Transfer dann abgebrochen werden. Auf diese Weise sind nur die 3 letzten Transfers im Burst nicht überprüfbar, da die Übertragung schon beendet ist, wenn die Signale zum Abbruch auf dem PCI-Bus erscheinen. Außerdem werden beim Parameterextraktor im normalen Betrieb aus dem Memory-Bereich nur Daten gelesen. Dabei ist die Kombination der Byte Enable Signale ohne Bedeutung, da immer vier gültige Bytes geliefert werden.

7.1 Synthese mit Leonardo Spectrum

Dadurch, daß Leonardo Spectrum nicht im offiziellen Design Flow des LogiCORE PCI 32 für die XC4000XLT Serie von XILINX enthalten ist, sind für dieses Tool auch keine Skripte oder Anleitungen erhältlich. Es musste durch längeres Probieren und Testen aus den verfügbaren Application Notes für Leonardo 4.2.2 oder PCI32 CORE für Virtex selbst eine Lösung erarbeitet werden. Als einfachster Weg für die Synthese hat sich folgende Vorgehensweise herausgestellt:

1. Einstellen des verwendeten FPGAs
2. Einlesen aller VHDL-Files in Leonardo Spectrum mit Ausnahme der Files des PCI-CORE Simulationsmodells, des Wrapper Files (pcis_lc.vhd und pci_lc_t.vhd) und der Verhaltensbeschreibung des RAMs das im FIFO verwendet wird (RAM16X1S.vhd). In diesen Files kommt das oben erwähnte „--synopsis translate_off" vor.
3. Das automatische Einfügen von Ein- und Ausgangstreibern für die Signale des PCI-Busses muß verhindert werden, da sie schon in der Netzliste des PCI-COREs enthalten sind und zu Problemen führen würden. (für jedes Signal: set_attribute -name nopad -value true -port <Signalname>)
4. Setzen der Constraints für das selbst erstellte Design.
5. Optimierung des Designs
6. Umwandeln der Namen in der Netzliste in Großbuchstaben, da sonst das mit dem PCI-CORE gelieferte Guide-File nicht passt. (lo2up)
7. Schreiben der Netzlisten im .XNF Format.

Danach kann das Place and Route wie in der Dokumentation des COREs beschrieben erfolgen.

7.2 Abkündigung des FPGAs

Wie schon vorher beschrieben unterstützt der verwendete PCI CORE in der Version 2.0 nur die FPGAs der 4000XLT Serie. Kurz vor Fertigstellung der Diplomarbeit wurde diese Serie der FPGAs abgekündigt. Die Lieferung des Bausteins verzögerte sich dadurch sehr stark. Die neuen FPGAs der Serie 4000XLA werden nur von der Version 3.0 des COREs unterstützt. Leider haben sich zwischen den beiden Versionen einige Veränderungen ergeben. So haben sich

einige Signalnamen verändert, einige Signale haben ein anderes Verhalten, die Ein- und Ausgangstreiber sind nicht mehr im PCI-CORE enthalten, der in den FPGAs enthaltene „STARTUP" Block, muß nun in der User Applikation instanziert werden, usw. Ein einfaches Einlesen des COREs in Renoir mit HDL to Graphics führt zu einigen Problemen mit der Simulation. In der Version 3.0 gibt es nur noch einen CORE der auch Masterfähig ist. Bei Einsatz als Target-Device müssen die entsprechenden Eingänge des COREs speziell verschaltet werden, damit sie während des Place and Routes nicht wegoptimiert werden.

8 Zusammenfassung und Ausblick

Für neue Designs sollte darauf geachtet werden, daß bei Transfers alle Byte Enable kombinationen unterstützt werden. Es sollten also 4 Byte Enable Signale zum Backend vorgesehen werden. Man sollte von beginn an mit der Version 3.0 des PCI-COREs arbeiten, auch wenn nur ein Target Device erstellt wird.

Aktueller Stand:

- Die Timing Simulation des Interfaces funktioniert
- Die Leiterkarte für das Interface ist vorhanden
- Die praktische Erprobung konnte noch nicht durchgeführt werden, da das benötigte FPGA noch nicht geliefert wurde.

9 Literatur

[1] PCI Special Interest Group: PCI Local Bus Specifikation, Revision 2.1, 1. Juni 1995

[2] PICMG PCI Industrial Computers Manufacturers Group: CompactPCI Specification, Version 2.0, Revision 2.1, 2. September 1997

[3] XILINX: LogiCORE PCI32 User´s Guide, Version 2.0

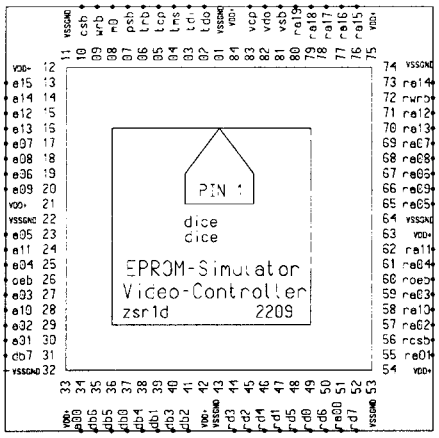
[4] XILINX: Synthesizable PCI Bridge Design User´s Guide

zsr1d

CAE-Projekt
zsr1d

W.Ludescher
FH
Ravensburg-Weingarten

25. Januar 2000



EPROM-SIMULATOR

Zusammenfassung

Der Schaltkreis zsr1d realisiert einen EPROM-Simulator mit Speicher-Verwaltungs-Einheit und erzeugt ein Video-Signal. Die Schaltung ist in einer 0.8μm-CMOS-Technologie aufgebaut und besitzt eine Komplexität von ca. 20.000 CMOS-Transistoren.

Einsatzgebiete:

- Nachbildung eines ROMs
- Video-Controller
- Memory-Mapper, Speicherverwaltung

Stand: Stand: 990923, Lu

1 Übersicht

1.1 Entwicklungsziel

Der Baustein zsr1 kann ...

- ...einem Mikroprozessor sein EPROM „vortäuschen"
- ...ein VIDEO-Signal erzeugen (CRT-Controller)
- Die Betriebsart des zsr1 ist durch den externen, globalen Kontrol-Pin ($m0$) und durch einen internen Registersatz definiert.

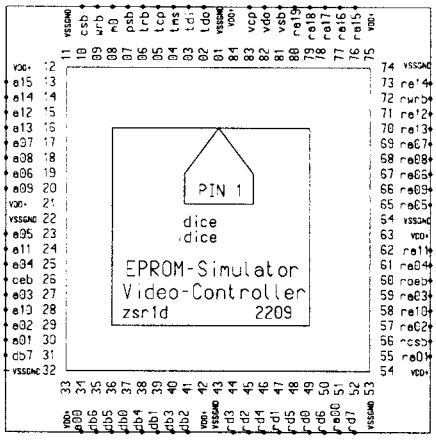
Als **„EPROM-Simulator"** nimmt zsr1 den Fußabdruck eines 64k-Byte-EPROMS entgegen und bildet ihn auf maximal ein MegaByte externen RAM- oder EPROM-Speicher ab. Die Zuordnung (Memory-Map) des Speicherbereichs erfolgt über interne Kontroll-Register, das Einschreiben von Daten kann über einen 8-Bit-Datenbus oder über eine aus fünf Pins bestehende Test-Schnittstelle erfolgen.

Als **„Datensichtgerät"** werden RAM- oder EPROM-Daten aus dem Simulations-Adressraum vom zsr1 eingelesen und in ein Videosignal umgewandelt. Pixel- und Synchronisations-Signale stehen an zwei Pins zur Verfügung, ein Video-Austastsignal kann bei Bedarf auf den Testausgang geschaltet werden. Das Bildformat beträgt 16 Text-Zeilen zu je 48 Zeichen, was problemlos auf ein einfaches Fernsehgerät (z.B. einen Video-Walkman) paßt.

Funktionsblöcke des zsr1 wurden im Rahmen eines technischen Wahlfachs (CAE Elektronischer Schaltungen) im Verlauf des Sommersemesters 1998 entworfen, simuliert und verifiziert.

Teilnehmer, 7. Semester Elektronik:

Büchele, Siegfried;
Bulach, Eric;
Elbs, Alexander;
Fischer, Stephan;
Genzel, Peter;
Hecker, Robert;
Köerber, Martin;
Kramer, Andreas;
Müller, Wolfgang;
Schmidt, Klaus;
Stoppel, Simon;
Sassano, Mario;
Waitschies, Eldor;



Chipfertigung:	IMS-Stuttgart
EDIF-Tape-out:	Ende SS98
Komplexität:	ca. 22.000 Transistoren
Technologie:	0.8μm CMOS

1.2 EPROM-Simulator, 64k-Mode

Ist $m0 = 0$, dann nimmt der Baustein den „Fußabdruck“ eines 64kByte-Eproms, bestehend aus 16 Adressleitungen $a00..a15$, 8 Datenleitungen $db0..db7$ und den Freigabe-Signalen *oeb* und *csb*, entgegen um sie auf 16 Adressleitungen $ra00...ra15$, 8 Datenleitungen $rd0...rd7$ und auf die Freigabe-Signale *roeb* und *rcsb* abzubilden. Auf diese Weise wird „hinter“ dem Baustein zsr1 ein externes RAM angesprochen, das sich auf der Prozessorseite wie ein EPROM verhält.

Der ASIC „sieht“ die Eingangspins $a00 \ldots a15$ (vom μP) und bildet sie auf die Ausgänge $ra00 \ldots ra15$ (als Speicher-Adresse) ab. Die Signale $ra00 \ldots ra15$ folgen den Signalen $a00 \ldots a15$, die Signale $ra16 \ldots ra19$ dekodieren die Adressleitungen $a00 \ldots a15$: $ra16$ wird low, wenn im μP-Adressraum $0xOXXX$, also der erste 4k-Block, auf dem Adressbus $a00 \ldots a15$ liegt. $ra17$ wird low, wenn der μP-Adressraum $0xFXXX$, also der letzte 4k-Block, angesprochen wird. $ra18$ wird low, wenn die μP-Adresse im den Bereich $0x1000 \ldots 0x7FFF$ fällt. ra19 wird low, wenn die μP-Adresse im den Bereich 0x8000 0xEFFF fällt. Die vier Netze können – allerdings nur, solange $m0 = 0$ und damit der 64k-Mode anliegt – als RAM/ROM/EPROM-Chip-Selects verwendet werden, um Bausteine „hinter“ dem zsr1 zu aktivieren. $ra16$ ist nützlich bei Prozessoren, die nach dem Reset ein Programm ab Adresse $0x0000$ erwarten, $ra17$ wird benötigt, wenn der Prozessor nach dem Reset z.B. seinen Startvektor aus dem hohen Adressraum abholt. $ra18$ und $ra19$ unterstützen den Zugriff auf 32k-RAMs. Die Ausgangssignale *vdo* und *vsb* liefern ein low-Signal, wenn der Baustein auf der μP-Seite die Adresse $0xFFFF$ (*vdo*) oder $0xFFFE$ (*vsb*) sieht. Die sechs Netze dekodieren somit den Zustand der Signale $a00 \ldots a15$.

Im Ergebnis lassen sich beispielsweise Speicher mit einem Adressraum von 4kByte im Segment 1, mit 32k in den Segmenten 2 und 3 und mit 4k im Segment 4 ansprechen. Die Signale *vdo* und *vso* werden aktiv low, wenn auf die Adressen $0xFFFF$ und $0xFFFE$ zugegriffen wird, was für spezielle Anwendungen (hardware-interrupt) nützlich ist.

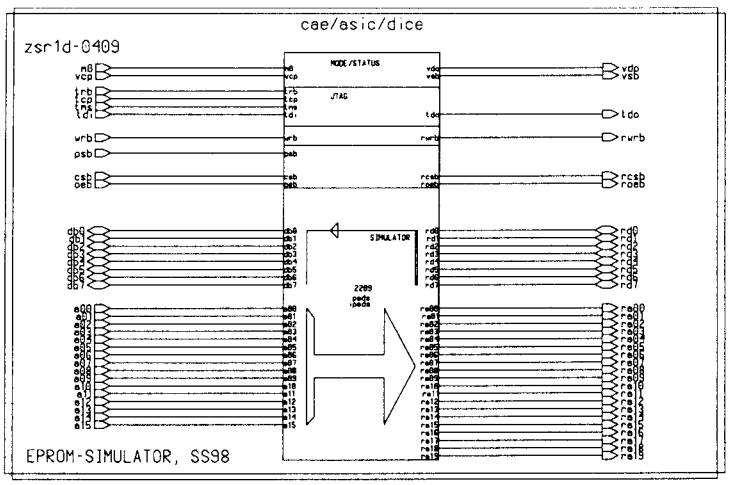


Abbildung 1: Blockschaltbild EPROM-Simulator

1.3 Serieller Zugriff über die Test-Schnittstelle

Um das RAM erstmals mit Daten zu füttern ist eine Test-Schnittstelle vorhanden, die aus den Eingangspins *trb*, *tcp*, *tdi*, *tms* und dem Ausgangspin *tdo* besteht. Damit kann das RAM addressiert, beschrieben und ausgelesen werden. Mit $m0 = 0$ wirken sich interne Zustände **nicht** an den ASIC-Klemmen aus. Dies soll ein unbeabsichtigtes Beschreiben des externen RAMs verhindern. Der Zugriff auf die internen Register ist aber möglich: dies erlaubt die vollständige Konfiguration des Bausteins, bevor die Betriebsart auf $m0 = 1$ gewechselt wird. Der externe Speicherbaustein sollte batteriegepuffert oder nicht-flüchtig sein - ein Prozessor würde in diesem Fall ein 64k-EPROM sehen, dessen Inhalt - falls notwendig - über die Testschnittstelle modifiziert werden kann.

Neben der seriellen Testschnittstelle kann der Baustein auch über den prozessorseitigen Datenbus $db(0:7)$ angesprochen werden. Dazu stehen die Pins *psb*, *csb*, *oeb* und *wrb* zur Verfügung. Um Register anzusprechen muß *psb* aktiv low, um Speicher anzusprechen muß *csb* aktiv low sein. Die Pins *oeb* und *wrb* entscheiden, ob gelesen oder geschrieben wird. Der ASIC verhält sich wie ein 8-Bit-Peripheriebaustein.

Über einen EPROM-Sockel ist kein Schreib-Zugriff möglich. Der Baustein arbeitet - auch ohne Takt und Reset - stabil, denn das interne Gedächtnis beeinflußt das Klemmenverhalten nicht, solange $m0 = 0$ ist.

1.4 EPROM-Simulator, Mega-Mode

Jetzt, gekennzeichnet durch $m0 = 1$, kann der Baustein einen erweiterten Speicherbereich ansprechen bzw. als Video-Controller arbeiten. In beiden Fällen müssen interne Register initialisiert sein – die Register sind zum Teil als transparente Latches ausgeführt und besitzen keinen Reset.

Diese Betriebsart wird man entweder zur Ansteuerung eines Datensichtgerätes (Video-Controller) oder zur Bereitstellung eines erweiterten Adressraumes (Mapper) nutzen. Eine Mischung beider Funktionen schließt sich zwar nicht aus, wird aber „optisch unangenehm", wenn Video-Controller und Prozessor zeitgleich auf das RAM „hinter dem ASIC" zugreifen. Der Prozessor greift ungehindert zu, dem Sichtgeräte-Teil wird dabei das RAM „weggenommen", was den Pixelstrom im Video-Teil verfälscht und stört.

Setzt man Pin $m0 = 1$, so ändert sich auch die Bedeutung der Pins $ra16..ra19$. Der prozessorseitige Adressraum (64kByte) wird in einen Adressraum von $16x64k - Byte$ abgebildet. Die Abbildung erfolgt blockweise, wobei ein Block aus einem zusammenhängenden 16kByte-Segment besteht, das in einem programmierbaren, beliebigen 64k-Abschnitt wieder auftaucht. So kann z.B. der prozessorseitige Adressraum $0x0000...0x3FFF$ in den Bereich $7x0000...7x3FFF$ zu liegen kommen. Die Zuordnung wird durch zwei Register gesteuert.

VIDEO-Mode, m0=1

Setzt man Pin $m0 = 1$, so ist ein Teil des 64k-Blockes auch als Video-Ram nutzbar. Das Video-Signal muß jedoch durch ein Bit in einem Konfigurationsregister auf die Ausgangspins *vdo* und *vso* freigeschaltet werden, andernfalls bleiben (wie im 64k-Mode) die Adressen $0xFFFE$ und $0xFFFF$ selektiert. Der Video-Controller bietet 16 Textzeilen zu je 48 Zeichen. Diese Auflösung wurde gewählt, um sicherzustellen, daß das auf dem Chip erzeugte Bild-Austast-Synchron-Signal (BAS-Signal) mit einem einfachen und preiswerten Video-Walkman darstellbar ist. Das Video-RAM des CRT-Controllers kann in einen beliebigen 64k-Abschnitt gelegt werden, wobei der Buchstaben-Generator (4k-Byte) immer an der Obergrenze ($F000...FFFF$) des jeweiligen Abschnittes zu liegen hat und das Video-RAM in einem beliebigen 4k-Block des gleichen Abschnittes angesiedelt ist. Das im Controller erzeugte BAS-Signal besteht aus 312 Videozeilen (kein Zeilensprung), aus denen mit dem Buchstaben-Generator die 16 Text-Zeilen aufgebaut werden.

2 Applikationsbeispiel

Das 64k-EPROM (links) wird auf ein 128k-RAM (rechts) durchgeschaltet. Im 64k-Mode werden vom 128k-RAM nur 64kByte verwendet, das Netz *ra*16 geht auf log0, wenn der Prozessor auf die ersten 4k-Byte des EPROM-Adressraumes zugreift. Das Netz *wrb* wird auf *rwrb* geschaltet und erlaubt es, über den 8 Bit breiten PC-Datenbus *db* auf den RAM-Datenbus *rd* durchschreiben, wenn $csb = 0$ ist.
*m*0 ist der Mode-Pin und bestimmt die Betriebsart. *psb* steht für Peripherie Select (low) und erlaubt es, über den 8 Bit breiten Datenbus *db* interne Register byteweise zu beschreiben. Daten müssen stabil sein, solange $wrb = 0$ ist.
vcp ist der Video-Pixel-Takt. Er beeinflußt die Adressen *ra*00..*ra*15 und die Ausgänge *vdo* und *vsb*, sofern sie durch ein internes Register auf diese Ausgänge geschaltet sind. Die Funktion ist nur mit $m0 = 1$ möglich; mit $m0 = 0$ wird auf *vdo* und *vsb* ein Zugriff auf die Adressen $0xFFFF$ und $0xFFFE$ erkannt.
Mittels *trb*, *tcp*, *tms* und *tdi* ist ein serieller Zugriff auf alle internen Register und damit auch auf die Daten des RAMs möglich. Der ASIC kann mit 4 Port-Bit eines PCs oder durch einen Single-Chip-Prozessor vollständig gesteuert werden. Das serielle Beschreiben des RAMs läuft wie folgt ab: DMA-Adresse – ab dieser Adresse werden Daten im RAM über *tdo* ausgelesen und über *tdi* eingetaktet – selektieren und definieren. Schreib-Freigabe selektieren, *tdo* auf RAM schalten und Schreibbit setzen. Anschließend RAM-Daten selektieren und eintakten – die DMA-Logik serialisiert die im RAM stehenden Daten. Ist das Schreib-Bit gesetzt, werden neue Daten auf das RAM gelegt und eingeschrieben. Anschließend wird die DMA-Adresse weitergezählt, um das nächste Byte auszulesen und danach ggf. zu überschreiben.

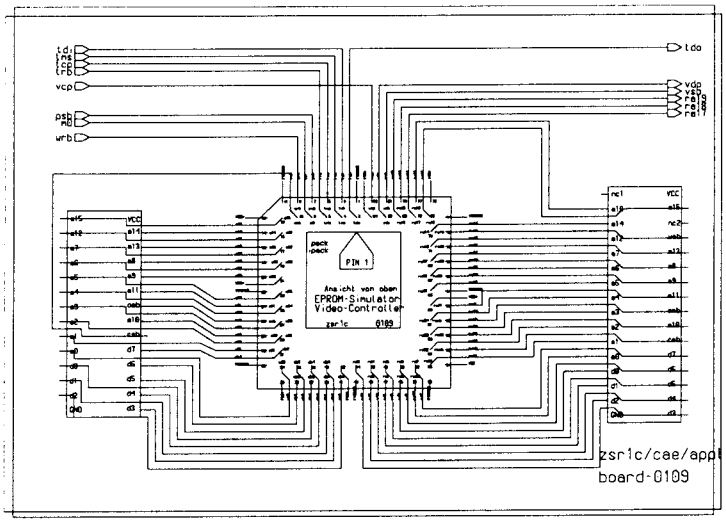


Abbildung 2: Ein 64k/128k-Byte-Eprom-Simulator

Das linke IC-Symbol stellt einen 28-Pin-DIL-Stecker dar, der auf den Sockel eines EPROMs (Intel Byte-Wide) passt. Mit einem 28-aderigen Flachbandkabel wird die Verbindung zur Simulator-Platine ausgeführt, „hinter " dem zsr1d befindet sich das RAM mit 128k-Bytes im 32-poligen DIL-Gehäuse.

3 Blockstruktur

3.1 Schaltbild „pads“

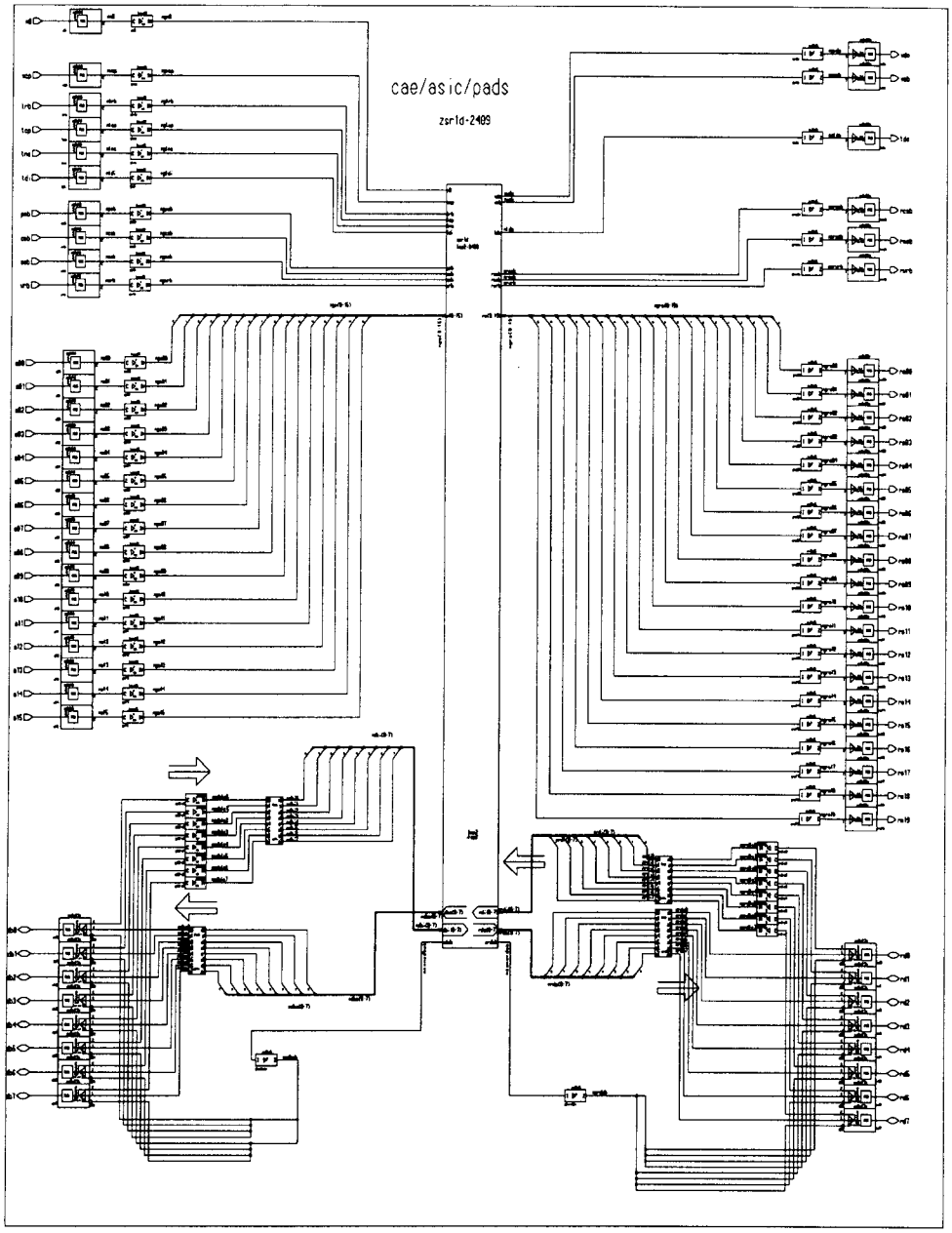


Abbildung 3: Periphere Zellen und Logik-Kern

Die Abbildung zeigt die peripheren Zellen und den Logik-Kern. Der μP-seitige Adressbus (links) wird nach rechts durchgereicht, der Datenpfad wird aufgetrennt, um einen seriellen oder parallelen Zugriff zu realisieren.

3.2 Schaltbild Logik-Kern „top2“

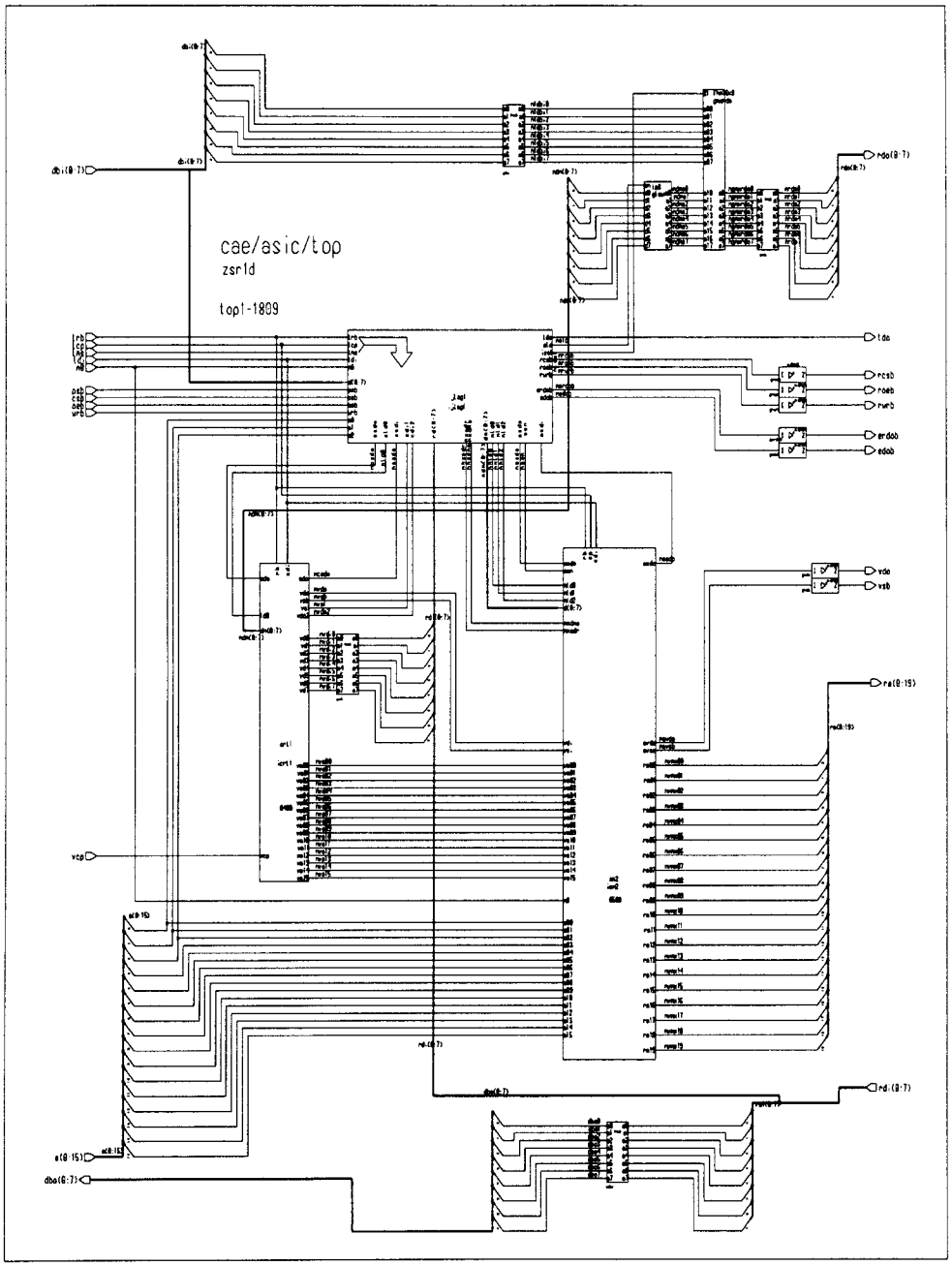
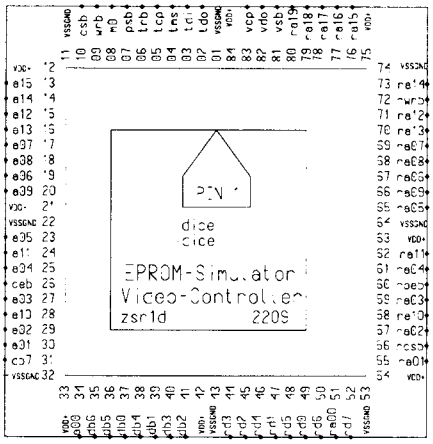


Abbildung 4: Logik-Kern mit Ladelogik, CRT und DMA

Die Abbildung zeigt den Logik-Kern mit Ladelogik, CRT und DMA. Der μP-seitige Adressbus und der Videocontroller (CRT) speisen den DMA-Multiplexer, um eine Adresse für das externe RAM zu erzeugen. Unabhängig davon kann DMA seriell geladen werden, um eine eigene Adresse zu definieren. Daten werden (links oben) parallel vom μP oder seriell mittels Test-Schnittstelle an den Daten-Multiplexer (rechts oben) angelegt und können dann am RAM-seitigen Datenbus *rdo* erscheinen. Das Auslesen des RAMs erfolgt sinngemäß, gespeist vom RAM-seitigen Datenbus *rdi* (rechts unten) entweder byteweise zum μP oder seriell durch die Test-Schnittstelle.

3.3 Pinbeschreibung „zsr1d“

01	VSSGND	P	Ground
02	tdo	O	Test Data Out
03	tdi	I	Test Data In
04	tms	I	Test Mode Select (low)
05	tcp	I	Test Takt (pos.Flanke)
06	trb	I	Reset (low)
07	psb	I	Peripheral Selekt (low)
08	m0	I	Betriebsart
09	wrb	I	uP-Write (low)
10	csb	I	uP-Chip-Select (low)
11	VSSGND	P	Ground
12	VDD+	P	+5V
13	a15	I	Adresse, vom PC
14	a14	I	Adresse, vom PC
15	a12	I	Adresse, vom PC
16	a13	I	Adresse, vom PC
17	a07	I	Adresse, vom PC
18	a08	I	Adresse, vom PC
19	a06	I	Adresse, vom PC
20	a09	I	Adresse, vom PC
21	VDD+	P	+5V
22	VSSGND	P	Ground
23	a05	I	Adresse, vom PC
24	a11	I	Adresse, vom PC
25	a04	I	Adresse, vom PC
26	oeb	I	Output Enable, wenn low
27	a03	I	Adresse, vom PC
28	a10	I	Adresse, vom PC
29	a02	I	Adresse, vom PC
30	a01	I	Adresse, vom PC
31	db7	B	Datenbus, PC
32	VSSGND	P	Ground
33	VDD+	P	+5V
34	a01	I	Adresse, vom PC
35	db6	B	Datenbus, PC
36	db5	B	Datenbus, PC
37	db0	B	Datenbus, PC
38	db4	B	Datenbus, PC
39	db1	B	Datenbus, PC
40	db3	B	Datenbus, PC
41	db2	B	Datenbus, PC
42	VDD+	P	+5V
43	VSSGND	P	Ground
44	rd3	B	Datenbus, RAM
45	rd2	B	Datenbus, RAM
46	rd4	B	Datenbus, RAM
47	rd1	B	Datenbus, RAM
48	rd5	B	Datenbus, RAM
49	rd0	B	Datenbus, RAM
50	rd6	B	Datenbus, RAM
51	rd7	B	Datenbus, RAM
52	ra00	O	Adresse, zum RAM
53	VSSGND	P	Ground
54	VDD+	P	+5V
55	ra01	O	Adresse, zum RAM
56	rcsb	O	RAM CHIP SELECT (low)
57	ra02	O	Adresse, zum RAM
59	ra03	O	Adresse, zum RAM
60	roeb	O	RAM OUTPUT ENABLE (low)
61	ra04	O	Adresse, zum RAM
62	ra11	O	Adresse, zum RAM
63	VDD+	P	+5V
64	VSSGND	P	Ground
65	ra05	O	Adresse, zum RAM
66	ra09	O	Adresse, zum RAM
67	ra06	O	Adresse, zum RAM
68	ra08	O	Adresse, zum RAM
69	ra07	O	Adresse, zum RAM
70	ra13	O	Adresse, zum RAM
71	ra12	O	Adresse, zum RAM
72	rwrb	O	RAM, WRITE (low)
73	ra14	O	Adresse, zum RAM
74	VSSGND	P	Ground
75	VDD+	P	+5V
76	ra15	O	Adresse, zum RAM
77	ra16	O	cs oder Adresse
78	ra17	O	cs oder Adresse
79	ra18	O	cs oder Adresse
80	ra19	O	cs oder Adresse
81	vsb	O	cs oder Video Synchron (low)
82	vdo	O	cs oder Video Data
83	vcp	I	Video Pixel Takt
84	VDD+	P	+5V



ROM-SIMULATOR

Kryptographiealgorithmus in einem FPGA

Bernd Köhler, Frank d'Argent

Fachhochschule Esslingen - Hochschule für Technik

1. Einleitung

Im Rahmen einer Gruppendiplomarbeit an der Fachhochschule Esslingen – Hochschule für Technik wurde mit den Entwurfswerkzeugen Renoir, ModelSim und Leonardo Spectrum ein ASIC für einen Kryptographiealgorithmus (DES) entworfen. Zielhardware war ein FPGA der Firma Xilinx.

Zum Programmieren und Testen des FPGA wurde eine ISA-Karte mit zugehöriger Software entwickelt.

Zunächst soll die Anwendung des entworfenen Chips beschrieben werden. Anschließend wird ein kurzer Einblick in das Thema Kryptographie gegeben, und dann anhand des verwendeten DES-Algorithmus die Realisierung in Hardware beschrieben. Darüber hinaus werden die verwendeten Design-Tools und die Funktion der entwickelten ISA-Karte vorgestellt.

2. Anwendung des Chip

Abbildung 1 zeigt *eine* mögliche Verwendung des DES-Chips:

Über ein MS-DOS-Anwendungsprogramm gesteuert, muß vom Sender ein geheimer Schlüssel eingegeben werden ① und eine Datei erstellt werden ②. Daraufhin wird diese Datei vom PC zu einer ISA-Einsteckkarte übertragen ③ und dort in einem Xilinx-FPGA nach dem DES-Algorithmus verschlüsselt. Nach dem Verschlüsselungsvorgang wird eine Datei mit gleicher Größe, jedoch verschlüsseltem Inhalt und geändertem Dateinamen, wieder zum PC zurückgeschrieben ④. Diese verschlüsselte Datei wird dann zum Empfänger übertragen ⑤. Der Empfänger der Nachricht muß zum Entschlüsseln nun den gleichen Schlüssel eingeben ⑥, gefolgt von dem Dateinamen der erhaltenen Datei. Nach dem erfolgreichen Entschlüsselungsvorgang ⑦,⑧ steht der

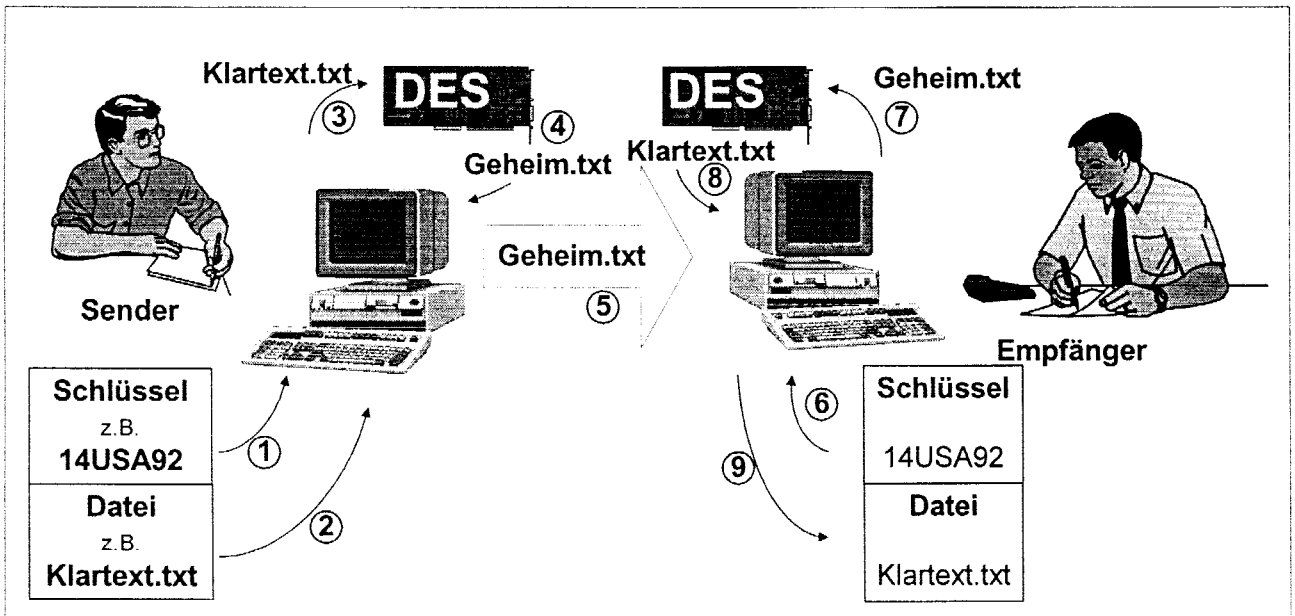


Abbildung 1: Verwendung der Hardware-Datenverschlüsselung

Klartext in einer weiteren Datei zur Verfügung ⑨.

Eine Verschlüsselung mit dem ASIC bringt eine erhöhte Sicherheit und ist bis zu hundert mal schneller als eine Softwarelösung.

3. Kryptographie

Sinn und Zweck der Kryptographie ist es, einen Klartext vor unberechtigten Personen (dem Feind) zu verbergen.

Die Wissenschaft der Kryptoanalyse versucht, die verschlüsselten Daten auch ohne Zugriff auf den Schlüssel wieder in Klartext zu überführen. Solch ein Versuch der Kryptoanalyse wird "Angriff" genannt. Es gibt mehrere Möglichkeiten, wie solch ein "Angriff" gestaltet werden kann, und tatsächlich kann jeder Code zumindest mit einem Brute-Force-Angriff (ein Ausprobieren aller nur erdenklichen Schlüssel) geknackt werden.

Es wird hauptsächlich in asymmetrische und symmetrische Algorithmen unterschieden. Bei den symmetrischen Algorithmen ist der Chiffrier- und Dechiffrierschlüssel meist identisch und muß von Sender und Empfänger vor der Kommunikation abhörsicher vereinbart werden. Die Sicherheit solcher Verfahren hängt meist von der Schlüssellänge ab. Vereinfacht gilt: Je länger der Schlüssel, desto sicherer das Verfahren. Die symmetrischen Verschlüsselungverfahren sind in der Regel sehr schnelle Verfahren und sind somit sehr gut geeignet, um große Datenmengen, wie sie in der heutigen Zeit vorkommen, zu verschlüsseln.

3.1. Data Encryption Standard

Ein bekanntes Beispiel für ein symmetrisches Verschlüsselungsverfahren ist das DES-Verfahren (Data Encryption Standard). Das Verfahren wurde bereits 1977 vom Amerikanischen Sicherheitsbüro (NSA) als Bundesstandard anerkannt und veröffentlicht.

Der DES gehört zur Familie der Blockchiffrierung. Die Daten werden dazu in Blöcke von 64 Bit aufgeteilt. Er liefert als Ergebnis 64-Bit-Chiffretext. Zur Ver- und Entschlüsselung werden der gleiche Algorithmus und der gleiche Schlüssel benutzt. Die Schlüssellänge beträgt nach Subtraktion der Paritätsbits noch 56 Bit. Der Schlüssel kann für jeden neuen Vorgang geändert werden, jedoch sollte man einige schwache Schlüssel vermeiden. Auf dieses Problem soll hier nicht weiter eingegangen werden.

DES bearbeitet immer 64 Bit lange Blöcke des Klartextes. Wie in Abbildung 2 gezeigt, wird dieser Block nach der Eingangspermutation in eine linke und rechte Hälfte aufgeteilt. Danach folgen 16 identische Runden, in denen die Daten und der Schlüssel verknüpft werden. Nach der 16. Runde

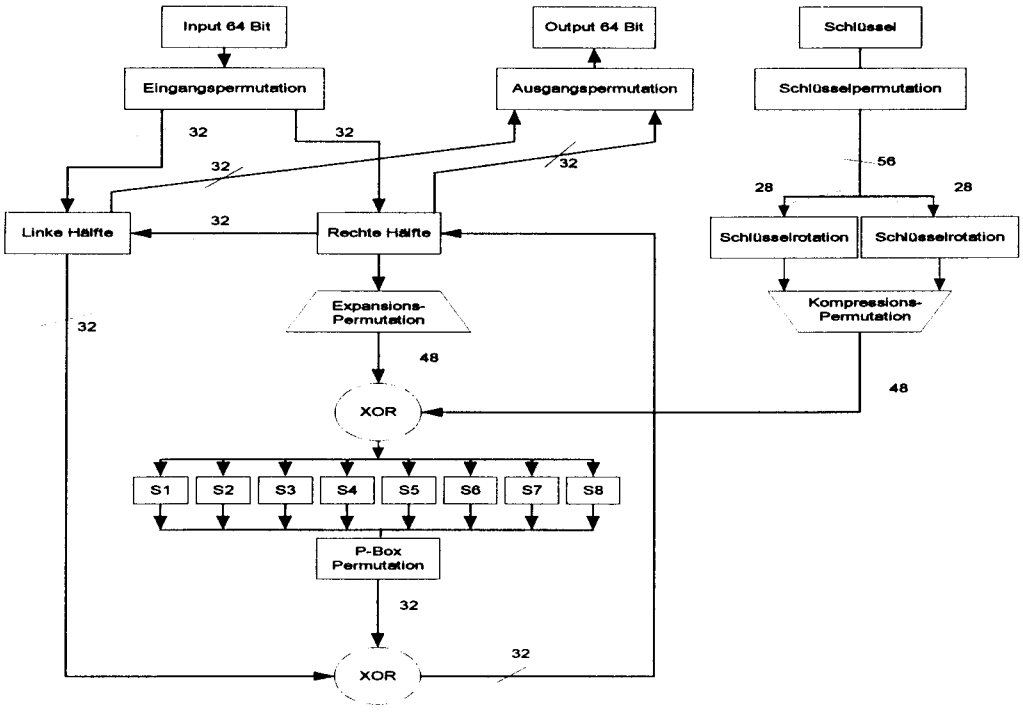


Abbildung 2: DES-Algorithmus

werden die zwei Hälften wieder zusammengefügt und durchlaufen als letzten Schritt die Ausgangspermutation.

4. Realisierung in Hardware

4.1. Permutationen

Im Algorithmus werden insgesamt acht verschiedene Permutationen benötigt. In VHDL sind alle Permutationen relativ einfach umzusetzen. Der folgende Ausschnitt einer VHDL Datei zeigt einen Ausschnitt aus der Schlüsselpermutation. Es handelt sich um einfache Zuweisungen.

```
KeyOut (0)  <=    KeyIn(57);
KeyOut (14) <=    KeyIn(10);
```

Eingangspermutation

Diese Permutation beeinflusst die Sicherheit des DES nicht. Die 64 Bit werden nach einer vom FIPS (Federal Information Processing Standards Publication) vorgegebenen Tabelle vertauscht (s.Quellenverzeichnis) .

Ausgangspermutation

Diese Permutation ist genau invers zur Eingangspermutation.

Schlüsselpermutation

Zu Beginn der Schlüsseloperation wird aus den 64 Bit jedes achte Bit entfernt und somit auf 56 Bit reduziert. Die entfernten Bits sind die Paritätsbits und führen somit nicht zu einem Datenverlust.

Kompressionspermutation

Bei jedem Rundendurchlauf werden verschiedene und auch nicht alle Bits verwendet. Es findet eine Reduzierung von 56 Bit auf 48 Bit statt. Darum wird diese Operation Kompressionspermutation genannt.

Zur Verdeutlichung der Kompressionspermutation ein Beispiel (Ausschnitt): vgl. Abbildung 3

Das Bit von Position 4 der Eingabe landet auf Position 16 der Ausgabe. Die acht Bits in Position 9, 18, 22, 25, 35, 38, 43 und 54 werden ignoriert. Die Tabelle 1 gibt vor, wie die Bits zugeordnet werden müssen.

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Bit 4 kommt auf die 16.Position

Tabelle 1: Kompressionspermutation

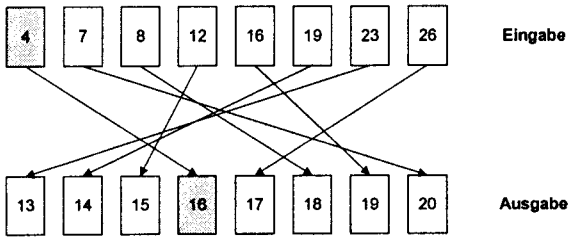


Abbildung 3: Ausschnitt der Kompressionspermutation

P-Box-Permutation

Die 32-Bit-Ausgabe der S-Boxen wird permutiert. Kein Bit wird doppelt benutzt und keines ignoriert.

Die Expansionspermutation

Die interessanteste der Permutationen expandiert die rechte Hälfte der Daten von 32 Bit auf 48 Bit. Da einige Eingabebits zwei Ausgangsbits beeinflussen, entsteht ein sogenannter ‚Lawineneffekt'.

4.2. Substitution

Die S-Boxen sind die wichtigste Stelle des DES. Alle anderen Operationen des Algorithmus sind linear und lassen sich nachvollziehen. Die S-Boxen sind nichtlinear und gewährleisten deshalb die Sicherheit des DES. Für die Umsetzung in Hardware muß die Zusammensetzung der verschiedenen S-Boxen verstanden werden.

Jede Box erhält eine 6-Bit-Eingabe und liefert eine 4-Bit-Ausgabe. Die S-Box ist eine Tabelle aus vier Zeilen und 16 Spalten. Jeder Eintrag der Tabelle besteht aus einer 4-Bit- Zahl.

Die Funktion der Auswahl soll an einem Beispiel erklärt werden:

Die Eingabe sei 110010_b. Das erste und letzte Bit zusammen (also 10_b) bestimmen die Zeile der S-Box. Hier ergibt dies die Zeile 2_d. Die mittleren vier Bits (hier 1001_b) ergeben die Spalte. Das ergibt die Spalte 9_d. Es gilt somit der Eintrag in der zweiten Zeile und der neunten Spalte. Nun muß noch die Zählweise ab 0 beachtet werden und man landet beim Wert 12_d, was wiederum 1100_b ergibt.

Bei der S1-Box wird also für den Eingangswert 110010 der Wert 1100 substituiert.

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Tabelle 2: Aufbau der S1-Box

Bei der Umsetzung der S-Boxen in Hardware wurden die mittleren vier Bit (für die Spalte) getrennt ausgewertet. Mit dem ersten und letzten Bit wurde in einem Multiplexer die Zeile bestimmt.

Sinn dieses Verfahrens sollte ein Geschwindigkeitsvorteil durch Parallelisierung sein.

Die Umsetzung der einzelnen Zeilen der S-Boxen (S1,...,S8) erfolgte mit Hilfe von Wertetabellen. Für alle S-Boxen sind somit acht mal vier Wertetabellen

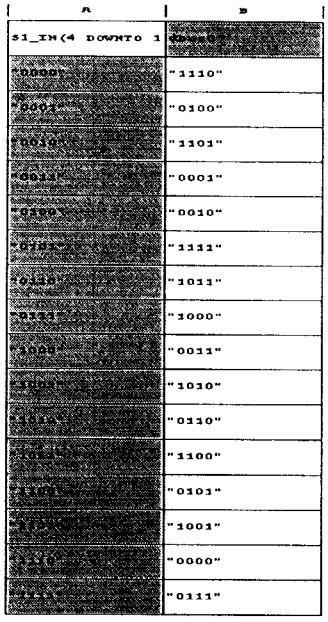


Abbildung 4: 1.Zeile der S1-Box.

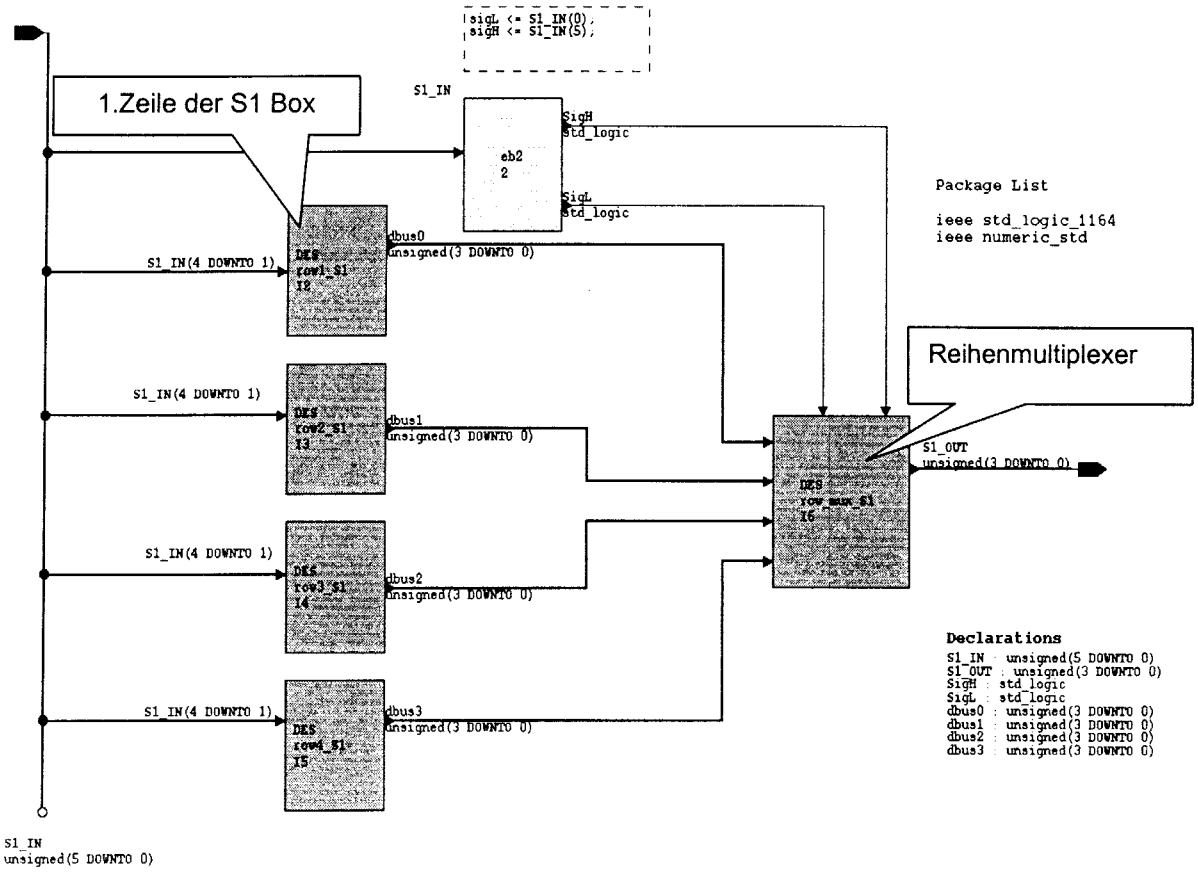


Abbildung 5: Umsetzung einer S-Box in Renoir

erforderlich. Die sechzehn Werte der Zeilen müssen in binärer Form eingegeben werden. Die Werte geben den Zeileninhalt der vom FIPS veröffentlichten S-Boxen wieder. Die Abbildung 4 zeigt die erste Zeile der S1-Box.

Die Umsetzung des Multiplexers in Renoir erfolgte in diesem Fall als Flowchart. Je nach Kombination des ersten und letzten Bits (also je nach Index der Zeile) wird die entsprechende Zeile ausgegeben.

Nach dem kompletten Verschlüsselungsvorgang werden die chiffrierten Daten wieder über den ISA-Bus zum PC geschrieben.

Register für linke und rechte Datenhälfte

Nach der Eingangspermutation werden die Daten aufgeteilt und taktflankengetriggert in zwei 32-Bit-Registern gespeichert. Das rechte Register benötigt einen Output Enable-Eingang, um den Ausgang hochohmig zu schalten, damit auf dem angeschlossenen Bus kein Schreibkonflikt auftreten kann.

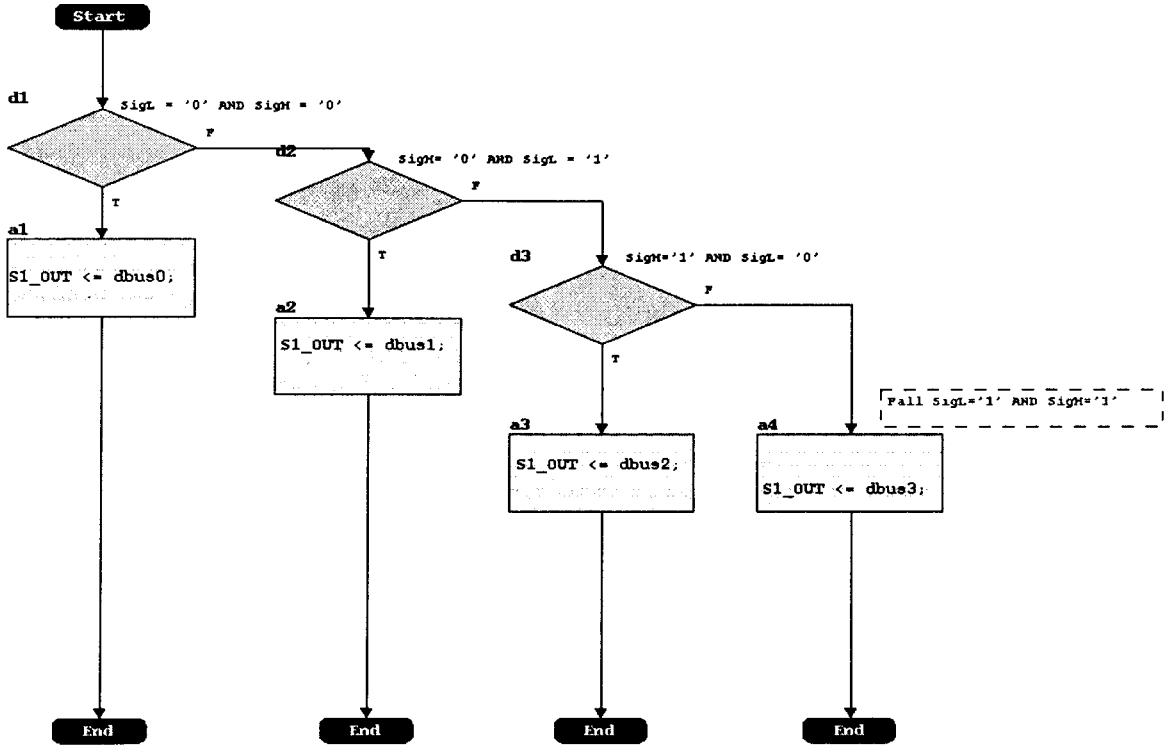


Abbildung 7: Reihenenmultiplexer

4.3. Register

Zur Zwischenspeicherung der Daten bis zu ihrer Gültigkeit benötigt der DES fünf unterschiedliche Register.

Eingangsregister

Die Daten und auch der Schlüssel werden vor jedem Chiffriervorgang vom ISA-Bus in ein 64 Bit Eingangsregister geschrieben. Dies geschieht byte-weise in acht Vorgängen hintereinander. Gleich-zeitig wird ein Paritycheck durchgeführt, der im Falle einer Schlüsselübertragung verwendet wird, um die richtige Übertragung des Schlüssels zu bestätigen Ein Chiffrieren mit falschem Schlüssel wäre katastrophal, da das Ergebnis nicht mehr zu entschlüsseln wäre.

Ringregister

Nach der einleitenden Schlüsselpermutation wird der Schlüssel halbiert und in zwei Ringregistern gespeichert. Die Ringregister haben die Aufgabe, den Schlüssel bei jedem der sechzehn Rundendurchläufe des Algorithmus zu rotieren. Dazu benötigen die Register einen Decrypt/Encrypt-Eingang, da beim Ent- und Verschlüsseln unterschiedlich rotiert wird. Nach Ablauf der sechzehn Runden muß jedes Register wieder in seinem Urzustand stehen. Dies wird dadurch erreicht, daß in den sechzehn Runden öfters doppelt rotiert wird. Insgesamt kommt man so auf 28 Rotationen, was einer kompletten Runde des 28-Bit-Ringregister entspricht.

4.4. Bus

Für die Kommunikation der Module innerhalb des DES benötigt man Busbreiten von 8 Bit bis zu 64 Bit. Solche Busbreiten sind für die Umsetzung im FPGA aufgrund der Kanalstruktur nicht trivial. Im verwendeten Xilinx FPGA steht nur eine begrenzte Anzahl von programmierbaren Verbindungen zur Verfügung (max. 110 General Purpose Interconnect Verbindungen, max. 44 Longlines und zusätzlich noch Direct Interconnect Verbindungen).

Denkbar wäre, Teile der Daten blockweise seriell zu übertragen, um die Busbreiten zu reduzieren. Dies würde aber ein erneutes Zwischenspeichern der Daten erfordern, da die Daten zur Weiterverarbeitung immer in voller Breite anstehen müssen. Dies wiederum würde mehr Logikblöcke beanspruchen und die Durchlaufzeit des kompletten Systems erhöhen. Serialisierung führt somit zu keiner guten Lösung, die hohen Busbreiten sind für die DES Anwendung unverzichtbar.

4.5. Steuerwerk

Alle Funktionseinheiten des DES-Algorithmus werden durch ein Steuerwerk koordiniert.

Die Umsetzung in Renoir erfolgte durch einen Moore-Zustandsautomaten (State-Machine). Für jeden einzelnen Verschlüsselungs- und Entschlüsselungsvorgang gibt es einen eigenen Pfad durch den Automaten. Die Zustände in Abbildung 8, die mit dem mehrfachen Rand gekennzeichnet sind, deuten an, daß es sich um hierarchische Zustände handelt. Dies kann man aus Gründen der Übersichtlichkeit wählen. In die Unter-Zustände gelangt man durch Doppelklicken auf den jeweiligen Zustand. Insgesamt wurden für den kompletten Zustandsautomaten ca. 120 Zustände benötigt.

Ablauf der Statemachine

Im Regelfall werden die 64 Bit für den Schlüssel in die Ringregister und anschließend die 64 Bit der Klartextdaten in die Datenregister geschrieben.

Wenn die Daten gesendet wurden, gibt das DOS-Anwendungsprogramm die erforderlichen Signale, um mit der Verschlüsselung zu beginnen. Dies erfolgt im FPGA so schnell, daß in der nächsten Programmzeile schon wieder die Daten zurück gelesen werden können.

Wenn die Software die Gültigkeit der Daten meldet, reagiert das Steuerwerk mit *Busy*. Die Software liefert nun noch die weitere Information, ob ent- oder verschlüsselt werden soll, worauf der entsprechende Pfad in der Statemachine eingeschlagen wird. Nachfolgend wird nur der Verschlüsselungsvorgang beschrieben. Die tieferen Hierarchien des Encrypt-Zustandes sind hier beschrieben, jedoch nicht abgebildet.

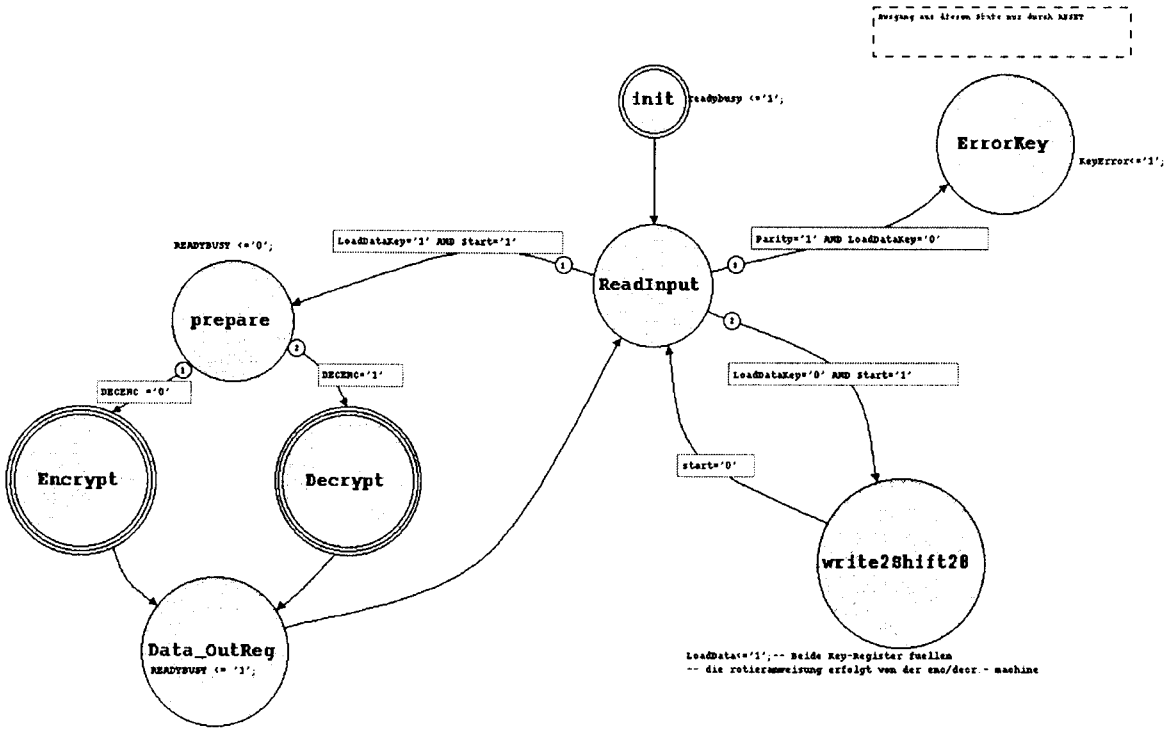


Abbildung 8: Oberste Hierarchieebene der DES-Statemachine

Die Entschlüsselung geschieht analog zur Verschlüsselung.

In der Verschlüsselung wird zuerst das XOR und das rechte Datenregister auf Tristate geschaltet. Die Eingangspermutation wird danach freigeschaltet. Ein Buskonflikt wird damit verhindert. Wenn die Daten beim nächsten Takt in den Registern stehen, wird die Eingangspermutation gesperrt und das Datenregister sowie das XOR freigegeben. Da jetzt alles zum Verschlüsseln vorbereitet ist, können nun 28 Rotationen erfolgen. Es wird abhängig vom Rundenindex um jeweils ein oder zwei Bit rotiert. An beide Ringregister wird dafür ein Takt-Signal gegeben. Allein für die Rotationen benötigt das Steuerwerk 16 Zustände. Da nach jeder Runde die Daten vom rechten Register zum neuen linken Registerinhalt werden, benötigt man jeweils ein weiteres Signal, welches die Vertauschung einleitet.

Nach Ende der sechzehn Runden sind die 64-Bit-Daten nach DES-Standard verschlüsselt. Jetzt ist das Ringregister mit dem Schlüssel wieder an seiner ursprünglichen Stelle, und der Zustand *Busy* kann nun wieder auf *Ready* wechseln. Sollen nun weitere Daten verschlüsselt werden, kann dies sofort anschließend erfolgen, ohne daß der Schlüssel neu geladen werden muß.

Im Falle eines Reset werden alle Signale auf den voreingestellten Wert zurückgesetzt. Ein Reset wird zum Beispiel im Fehlerfall bei der Schlüssel-paritätsprüfung ausgelöst.

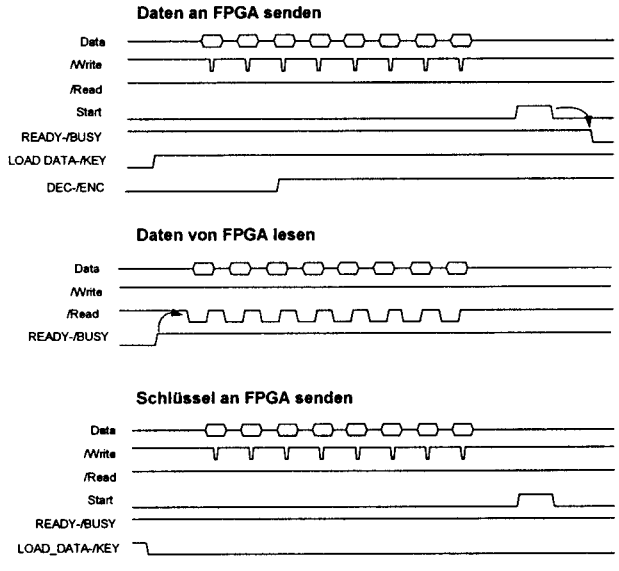


Abbildung 10: Timing der auftretenden Datentransfers

Obenstehende Timing-Diagramme zeigen die prinzipiellen Signal-Zeitverläufe an der Schnittstelle FPGA / ISA-Karte. Man erkennt, daß die Daten und der Schlüssel jeweils in 8-Bit-Paketen ausgetauscht werden.

5. Design-Tools

Für den Systementwurf des DES-Verschlüsselungs-algorithmus und seine Implementierung in ein FPGA wurden folgende Softwarepakete verwendet: Mentor Graphics Renoir, Leonardo Spectrum, Modeltech ModelSim und Xilinx Alliance.

Diese Tools sollen kurz vorgestellt werden.

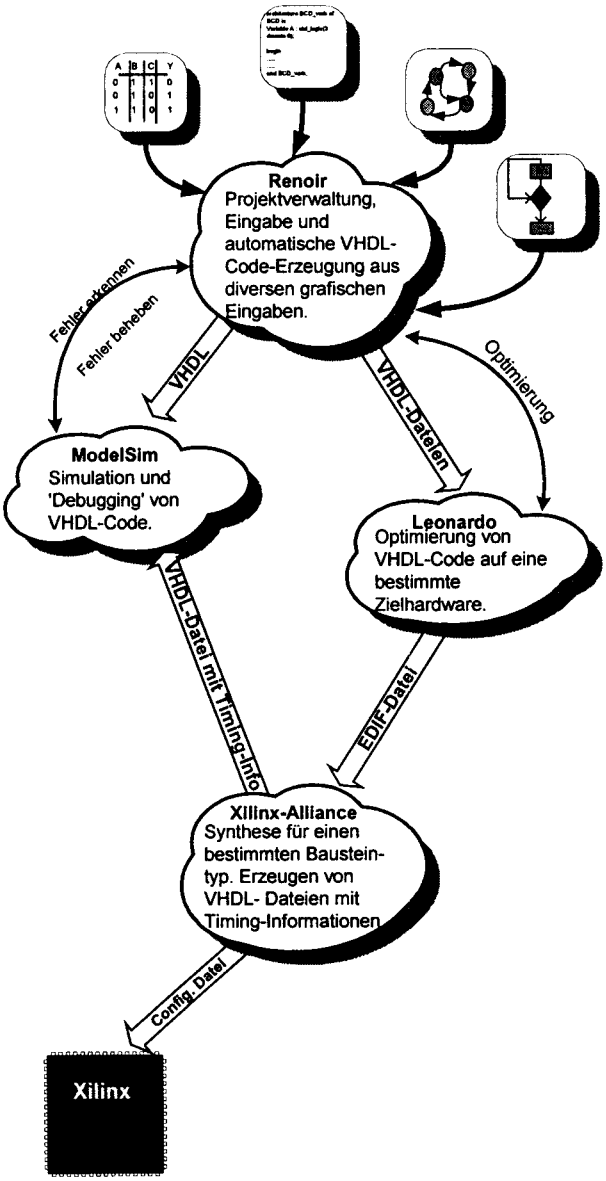


Abbildung 9: Verwendete Design-Tools

Mentor Graphics Renoir

Mit diesem Programm wird der Entwurf erstellt und das Projekt als ganzes verwaltet. Dazu erstellt man grafisch ein Top-Level-Design und danach wird schrittweise Block für Block mit Funktionalität gefüllt.

Zur Beschreibung der Blockfunktionalität können verschiedene grafische Eingabemethoden verwendet werden (Blockschaltbild, Wahrheitstabelle, State-Machine-Diagramm, Flußdiagramm). Wahlweise kann hier auch in VHDL programmiert werden.

Renoir wandelt die Eingaben in VHDL-Code um. Die Funktionsbeschreibungen werden dabei zu Architectures. Die zugehörigen Entity-Deklarationen erzeugt Renoir aus den Symbolen der einzelnen Blöcke.

Modeltech ModelSim

Mit ModelSim können VHDL-Dateien kompiliert und simuliert werden. ModelSim arbeitet dabei mit Renoir zusammen, so daß man in den Renoir-Schaltbildern einzelne Signale markieren kann, die bei ModelSim dann im Wave-Fenster angezeigt werden.

Darüber hinaus bietet ModelSim die Möglichkeit, den VHDL-Code Zeile für Zeile zu debuggen, um Fehler in der Programmierung zu finden.

Leonardo Spectrum

Leonardo Spectrum dient zur Optimierung von VHDL-Code. Dazu wählt man die Zielhardware aus (z.B. FPGA von Xilinx, 3100er-Familie). Leonardo kann nun den VHDL-Code auf die in der Hardware vorhandenen Ressourcen hin optimieren.

Leonardo erkennt auch nicht synthetisierbaren VHDL-Code, z.B. Wait-Anweisungen. Nach erfolgreicher Optimierung kann man eine EDIF-Datei erstellen, die dann weiter verwendet werden kann.

Xilinx Alliance

Mit Xilinx-Alliance kann die EDIF-Datei, die mit Leonardo erstellt wurde, eingelesen werden. Dann erfolgt die Auswahl eines bestimmten Bausteintyps (z.B. XC3195A im PLCC84-Gehäuse). Der Benutzer hat die Möglichkeit, bestimmte Rahmenbedingungen vorzugeben (z.B. Pin-Belegungen). Anschließend wird dann für diesen Baustein eine Programmierdatei erzeugt.

Mit dem FPGA-Editor besteht die Möglichkeit, das Layout auf dem Chip zu betrachten. Nachfolgendes Bild zeigt den Floorplan zweier 64-Bit-Register im XC3195A. Dieser Teil des Gesamtentwurfs konnte im Rahmen der Diplomarbeit implementiert und getestet werden.

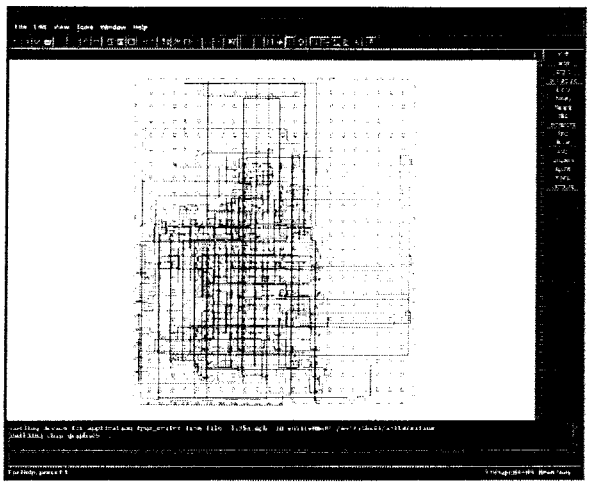


Abbildung 11: Floorplan im FPGA-Editor

Der komplette DES-Algorithmus war für den verwendeten XC3195A-Baustein (484 verfügbare CLBs) zu groß. (110% CLB-Ausnutzung) .

In einem Spartan XCS40XL (mit 784 CLBs) wurde ein Ergebnis von 74% CLB Ausnutzung berechnet. Dieser Baustein wäre also für das Gesamtdesign geeignet.

6. ISA-Karte als Testumgebung

Die entwickelte ISA-Karte mit zugehöriger Software dient zum Programmieren des FPGA und zum Testen der DES-Grundfunktionen.

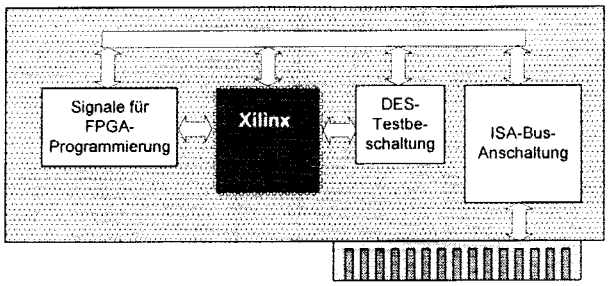


Abbildung 12: Prinzip der ISA-Karte

Die ISA-Steckkarte gliedert sich in 4 Bereiche: Das FPGA der Firma Xilinx, die ISA-Busanschaltung, eine Beschaltung zum Durchführen von Funktionstests und einen Schaltungsteil, der Signale zur Programmierung des FPGA erzeugt und bereitstellt. Abbildung 13 zeigt die fertige Lochrasterplatine und die Einteilung der verschiedenen Funktionsbereiche.

Im folgenden werden nun die einzelnen Funktionsgruppen kurz erläutert.

ISA-Busanschaltung

Die Anschaltung an den ISA-Bus übernimmt mehrere Aufgaben:

- Adressdekodierung der Adresse auf dem ISA-Bus. Die ISA-Karte belegt dabei 4 Adressen. Mit Hilfe von DIP-Schaltern kann die Basisadresse frei eingestellt werden. Wird zum Beispiel die Basisadresse 300h eingestellt, dann wird von der Karte der Adressbereich 300h bis 303h belegt.

- Trennen der 8 Datenleitungen des ISA-Bus vom "internen" Datenbus. Zum Schutz des PC werden die 8 Datenleitungen nur zu dem Zeitpunkt durchgeschaltet, zu dem sie auch verwendet werden. Dies wird durch einen bidirektionalen Treiberbaustein realisiert.

- Erzeugen von CS-Impulsen. Abhängig von der anliegenden Adresse und davon, ob es sich um einen Schreib- oder Lesezugriff handelt, werden CS-Impulse erzeugt, die zur Ansteuerung von Registern etc. verwendet werden können. Insgesamt gibt es 8 verschiedene CS-Impulse, 4 bei Lesezugriffen und 4 bei Schreibzugriffen.

DES-Testbeschaltung

Die DES-Testbeschaltung besteht im wesentlichen aus zwei 8-Bit-Registern. Eines dient zum Einlesen und das andere dient zum Ausgeben von Daten.

Mit Hilfe dieser Register werden die Signale zur Steuerung des Verschlüsselungsablaufs erzeugt bzw. Statussignale vom FPGA eingelesen.

Signale für die FPGA-Programmierung

Dieser Schaltungsteil erzeugt die für die Programmierung des FPGA nötigen Signale. Dies geschieht mit Hilfe eines 8-Bit-Ausgaberegisters und einer monostabilen Kippstufe. Die Statussignale des FPGA werden mit einem weiteren 8-Bit-Eingaberegister eingelesen.

Weitere Features

Zur Erleichterung der Inbetriebnahme besitzt die ISA-Karte 4 Leuchtdioden, die direkt über entsprechende Vorwiderstände am FPGA angeschlossen sind. Damit können interne Signale nach außen geführt und angezeigt werden.

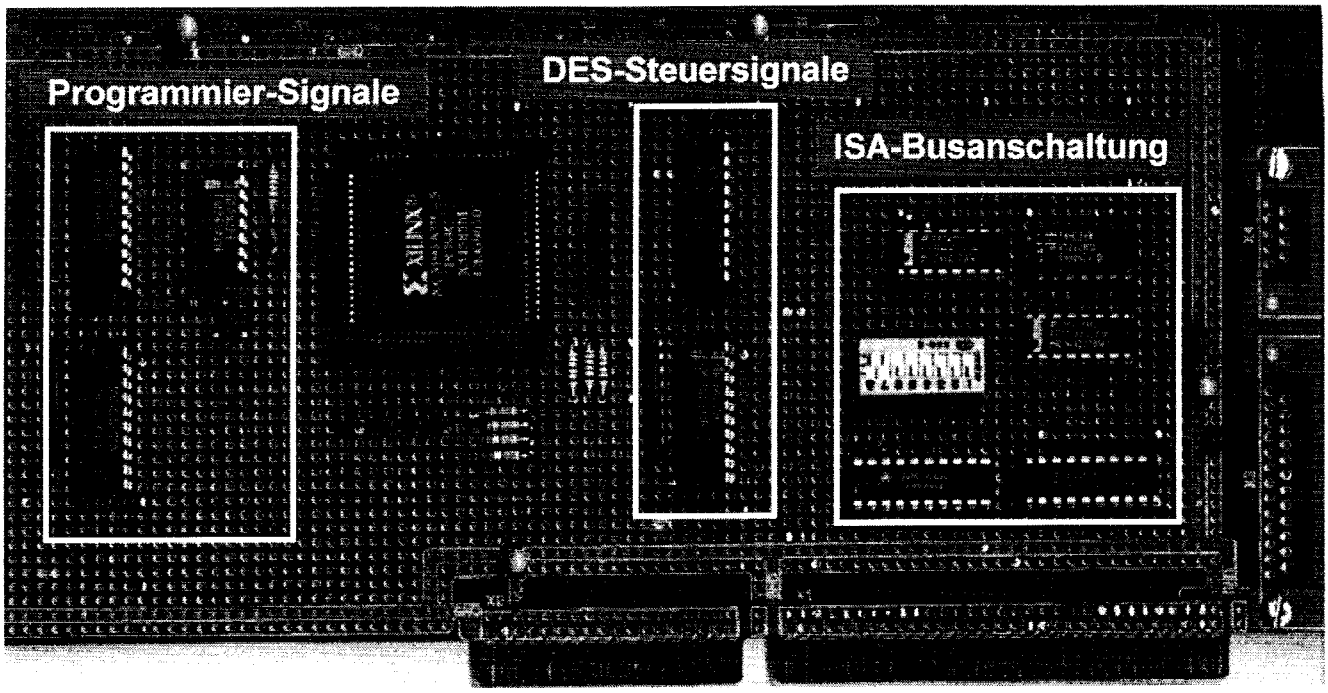


Abbildung 13: ISA-Karte als Lochrasterplatine

Verzeichnis der verwendeten Quellen

Literatur

Angewandte Kryptographie

Protokolle, Algorithmen und Sourcecode in C

Bruce Schneier, Addison-Wesley, 1996

ISBN 3-89319-854-7

Interfacetechnik zum Messen, Steuern, Regeln mit dem Industrie-PC

Michael Rose

Vogel Buchverlag Würzburg, 1991

ISBN 3-802304845

Internet

Entwurf und Test eines Verschlüsselungschips

Studentische Projektgruppe zum Entwurf und Test eines Kryptographie-Chips.

Universität Stuttgart

http://www.ra.informatik.uni-stuttgart.de/~stankats/

Technologien - Grundlagen der Kryptographie

http://www.utimaco.de/technik/krypto.htm

National Bureau of Standards

Federal Information Processing Standards Publication

http://www.itl.nist.gov/fipspubs/fip46-2.htm

VHDL-Kurzbeschreibung

http://tech-www.informatik.uni-hamburg.de/vhdl/

Xilinx-Online-Dokumentation

http://www.xilinx.com/

PLL-Synthesizer zur Frequenzvervielfachung

Dipl. - Ing (FH) Jürgen Hauser
Prof. Dr.- Ing Dirk Jansen
Fachhochschule Offenburg, ASIC-Design-Center
Badstr. 24, 77652 Offenburg
☎ :0781/205-274, Fax: 0781/205-174
E-Mail: hauser@fh-offenburg.de

Der beschriebene PLL erzeugt aus 32,768 kHz des Uhrenquarzes ein 11,0755 MHZ Signal, welches als Takt eines hier nicht weiter dargestellten IC verwendet wird. Der PLL besteht aus einem VCO nach dem Multivibratorkonzept, einer gepulsten Ladungspumpe und einem Doppel - Phasendetektor. Die Frequenzstabilität ist besser als ± 5kHz, die Stromaufnahme beträgt nur wenige Mikroamper. Die Chipfläche einschließlich des Schleifenkondensators beträgt 0,2mm² in einer 0,5µm CMOS-Technologie.

1. Konzept des Gesamtsystems

Der PLL-Synthesizer soll aus einem 32,768 kHz - Signal ein 11,0592 MHz - Signal erzeugen. Das Ausgangssignal hat somit eine um Faktor 337,5 höhere Frequenz. Da Teiler nur mit ganzzahligen Teilerverhältnissen realisiert werden können, gibt es nur die Möglichkeit, ein Teilerverhältnis von 338 bzw. 337 zu wählen oder auf eine niedere (halb so große) Vergleichsfrequenz herabzusetzen. Letzteres wurde untersucht, führte aber zu einem sehr ungünstigen Verhalten, da die nun in der Regelschleife wirksame Abtastrate im Verhältnis zur realisierten Loop-Bandbreite zu niedrig ist und damit die Schleife instabil bzw. sehr unruhig wird. Es wurde deshalb ein Teilerverhältnis von 338 gewählt, die Frequenzabweichung ist noch im Rahmen der Toleranz und wird deshalb in Kauf genommen.

Der Phasendetektor soll vom Typ 4 sein. Dieser Phasendetektor hat die besten Eigenschaften. Er hat zwei Ausgangssignale:

-Up: VCO-Frequenz zu niedrig

-Down: VCO-Frequenz zu hoch

Da es sich sowohl beim Eingang als auch beim Ausgang um digitale Signale handelt, können Teiler und Phasendetektor digital aufgebaut werden.

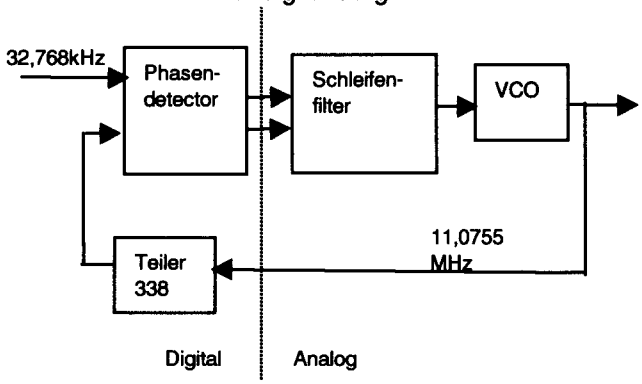


Abbildung 1: Ansatz des PLL-Synthesizers

Der VCO wird als analoges System aufgebaut, sein Ausgangssignal ist jedoch rein digital. Der VCO soll ein Signal liefern, dessen Frequenz abhängig von der Ansteuerung durch den Phasendetektors ist.

2. Der Analogteil

2.1. *Ansätze für den VCO*

Es gibt mehrere Möglichkeiten, einen VCO zu realisieren. Es wurden zunächst folgende Möglichkeiten untersucht:

1). Ringoszillator mit Inverterstufen

2). H-Brücke mit Inverter als Schwellwertschalter, der frequenzbestimmende Kondensator liegt im Querzweig der Brücke (Abb. 2).

3). Zwei geschaltete Stromquellen mit Differential Pair als Schwellwertschalter (Abb. 3).

Ansatz 1: Ringoszillator

Der Ringoszillator erwies sich für den gewünschten Frequenzbereich als ungeeignet, da die Schaltzeit der Inverter für 0,5µm CMOS zu kurz ist und sehr viele Stufen benötigt werden. Bei Verwendung halb-analoger Stufen (Inverter mit extra Kapazität) ist die Schaltschwelle nicht exakt genug definiert und es gibt starken Einfluß von Versorgungsspannung und Temperatur.

Ansatz 2: H-Brücke mit Inverter als Schwellwertschalter

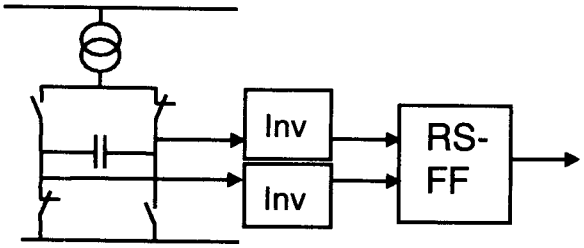


Abbildung 2: H-Brücke mit Inverter

Der zweite Ansatz verfolgt eine steuerbare Stromquelle, der frequenzbestimmende Kondensator liegt im Querzweig der Brücke. Zwei Inverter arbeiten als Schwellwertschalter und steuern ein RS-Flip-Flop an. Durch das Umschalten des Kondensators treten negative Spannungen auf. Ein potentialfreier Kondensator erfordert einen Analogprozeß, ein einseitig auf auf Masse liegender Kondensator ist einfacher zu realisieren. Demgegenüber ist die Ersparnis einer Stromquelle für beide Umladeprozesse flächenmößig unbedeutend.

Ansatz 3: Zwei geschaltete Stromquellen mit Diff. Pair und Hysterese

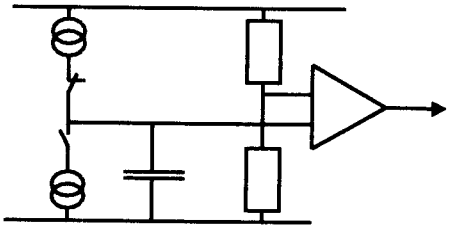


Abbildung 3: Zwei geschaltete Stromquellen mit Diff. Pair

Beim dritten Ansatz wird der Kondensator durch zwei umschaltbare Stromquellen versorgt. Die Stromquellen sind über einen Stromspiegel gekoppelt und somit ist das Ausgangssignal symetriert worden. Beim Einsatz eines komparators mit einer Hysterese von einigen mV können hohe Frequenzen erzeugt werden. Als Schwellwert-schalter wurde eine Differential Pair Schaltung untersucht. Die Einstellung der Hysterese ist aber problematisch und kann nur mit Widerständen erfolgen, die in der Technologie viel Platz erfordern. Der relativ kleine Hub zwischen den Umschaltpunkten ist zudem ungünstig und und erfordert hohe Verstärkung im Komparator.

2.2. *Realisierte Lösung des VCO*

Bei der Realisierung wurde nun versucht, die Vorteile aus Ansatz 2 und Ansatz 3 so zu verbinden, dass ein VCO mit den gewünschten Eigenschaften entsteht. Abbildung 4 zeigt die realisierte Lösung.

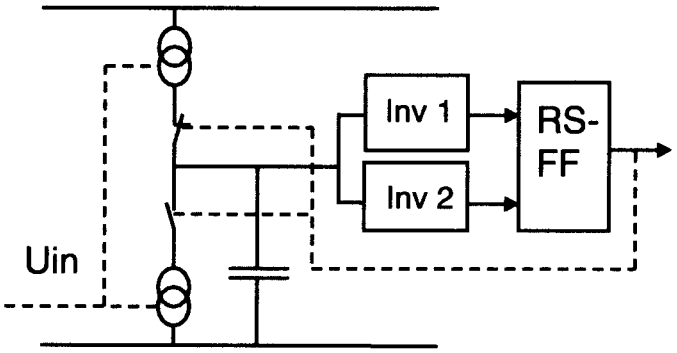


Abbildung 4: Realisierter Ansatz (Multivibrator)

Die Stromquellen zum Laden und Entladen des Kondensators wurden aus Ansatz 3 übernommen. Diese Variante verbraucht den geringsten Strom. Die Signalauswertung wurde dagegen aus Ansatz 2 übernommen und modifiziert. Beim Einsatz von zwei gleichen Schwellwertschaltern (Inverter) würde man keine Hysterese haben. Daher wurden die Schaltpunkte verschoben. Dieses Verschieben der Schaltpunkte wurde durch Einfügen zusätzlicher Transistoren erreicht.

Mit dem Ausgangssignal des RS-Flip-Flop werden die Stromquellen zum Laden des Kondensators geschaltet. Abb. 5 zeigt den realisierten Schaltplan. Die Schaltung arbeitet robust über einen großen Frequenzbereich mit praktisch jeder Kapazität.

Diese Lösung ähnelt dem im Timer 555 realisierten Multivibrator-Prinzip und funktioniert in der 0,5µm-Technologie bis über 100 MHz.

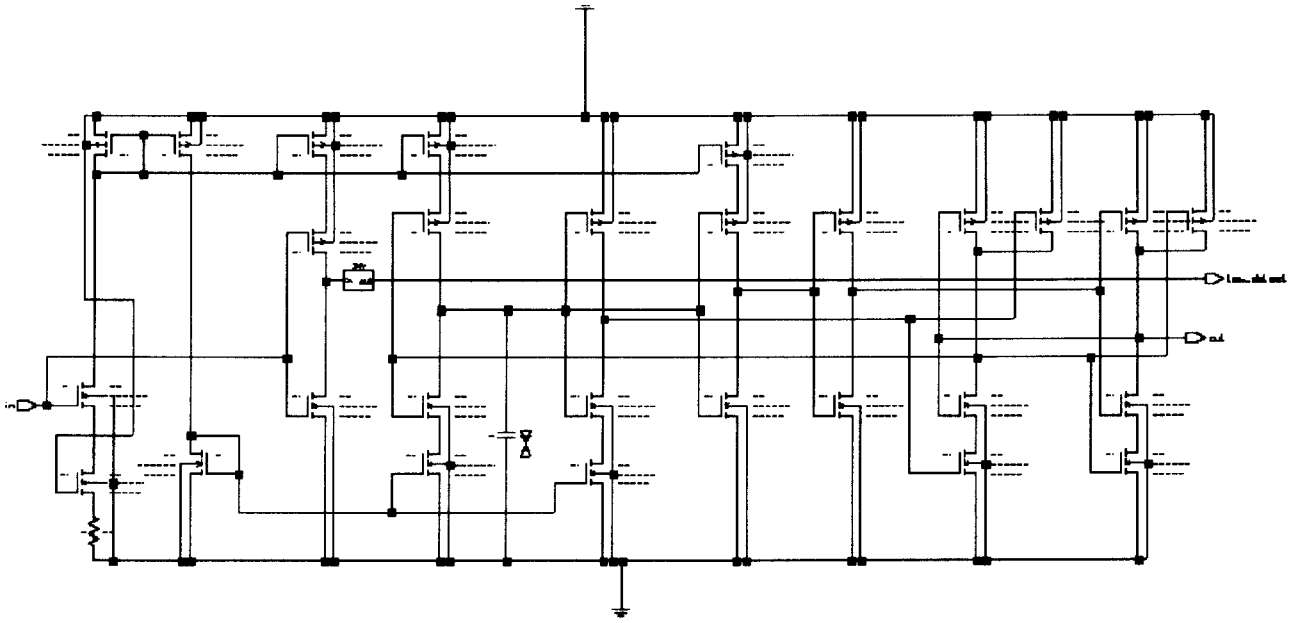


Abbildung 5: Schaltplan des VCO

2.3. Erzeugung der VCO-Steuerspannung

Der VCO benötigt eine Eingangsspannung,deren höhe die Ausgangsfrequenz bestimmt.. Für eine hohe Frequenzstabilität wird eine stabile Spannung benötigt. Dies bedeutet, dass ein großer Kondensator am Eingang des VCO eingesetzt werden muss. Um die Frequenz ändern zu können, muss die Spannung am Kondensator variabel sein. Dies kann man durch geschaltete Stromquellen erreichen. (Prinzip der Ladungs-pumpe)

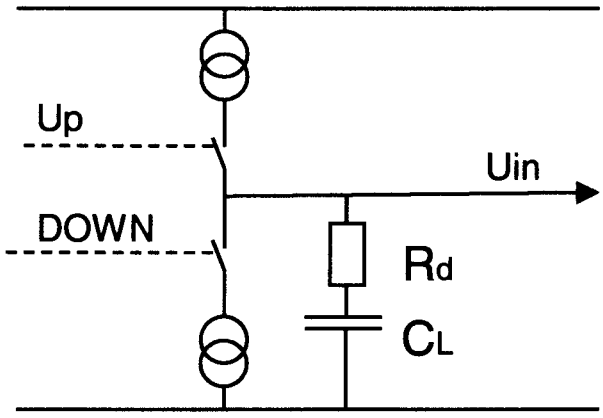


Abbildung 6: VCO-Steuerspannungserzeugung

Die Abschaltung des VCO kann hier ebenfalls realisiert werden. Da der VCO bei einer Eingangsspannung von 0 V kein Ausgangssignal liefert, braucht man nur den Kondensator kurzzuschließen.

Der Kondensator C_L bestimmt die Zeitkonstante der Regelschleife, die Dämpfung wird durch den Widerstand R_d in bekannter Weise eingestellt [Best].

2.4. Simulation des Analogteils

In Abbildung 7 ist das Zusammenspiel der Eingangsspannungserzeugung und des VCO's zu sehen. Es ist gut erkennbar, wie die VCO-Eingangsspannung den "up" und "down"- Signalen folgt.

Es ist auch zu erkennen, dass sich beim Ansteigen des VCO-Eingangssignales der Takt beschleunigt, entsprechend beim Abfallen des Eingangssignales sich der Takt verzögert. Bei gleichbleibendem Eingangssignal bleibt auch der Takt gleich.

Zu dem Zeitpunkt, an dem das Eingangssignal des VCO in eine Gerade parallel zur Nulllinie übergeht, geht auch das "Down"-Signal (Y4) auf Null. Diese Zustandsänderung wird leider von der Legende verdeckt.

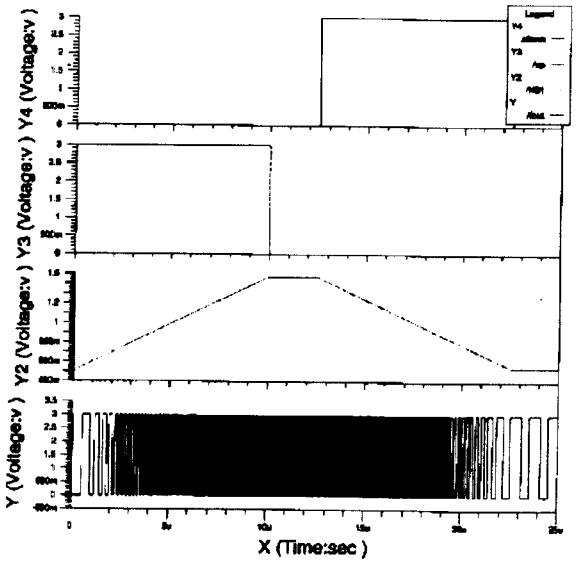


Abbildung 7: Simulationsergebnis des Analogteils

3. Digitalteil

Der Digitalteil ist in VHDL entwickelt worden. Dieser sollte, wie in Abbildung 1 zu erkennen ist, aus einem Teiler und einem Phasendetektor bestehen. Um das System simulieren zu können, wurde der bereits entwickelte Analogteil in VHDL nachgebildet. Hierbei wurde die Funktionalität der Ladungspumpe und der VCO zusammengeführt. Da die Ladungspumpe mit digitalen Steuerimpulsen angesteuert wird und der VCO letztlich ein digitales Ausgangssignal abgibt, handelt es sich um einen Block, der sich einfach in einer digitalen VHDL-Simulation einfügen läßt. Der VHDL-Code des Block stellt eine Verhaltensbeschreibung dar. Als Entwicklungstool wurde Renoir verwendet.

3.1. Nachbildung des Analogteils

Da die gemischte Simulation von VHDL und Analogdesign sehr komplex ist und der Mixed-Mode-Simulator Continuum sich als nicht geeignet herausstellte, wurde der Analogteil in VHDL nachgebildet. Es wurde versucht, das VHDL - Model der analogen Schaltung so genau wie möglich anzugleichen. Dazu wurde ein nur zur Simulation verwendeter Hilfstakt von 1 GHz dem VHDL-Model hinzugefügt, was einer Diskretisierung des Analogteils in 1ns-Schritten entspricht.

Zuerst wird die Kennlinie der Stromquelle zur Erzeugung des VCO-Eingangssignals nachgebildet. Um den "Ladezustand des Kondensators" möglichst oft zu berechnen, wurde das oben erwähnte 1 GHz-Signal benötigt. Durch diesen Takt wird der Prozess zur Berechnung des Ladezustandes jede ns durchlaufen. Der Ladezustand wird mit dem "filter_out" - Signal dargestellt.

```
architecture RTL of VCO is
signal Q_a : std_ulogic := '0';
signal delay : TIME := 0 ns;
signal schwellw : Time :=1000 ns;
signal low_det_int : std_ulogic :='1';

begin
  cp : process(up, down, clkghz)
  Variable filter_out : TIME := 200000 ns;--integer;2000000
  Variable p : TIME := 0 ns;
    BEGIN
      if up='1'
        then filter_out := filter_out + 624 ns;
          p := -15 ns;
      elsif down='1'
        then filter_out := filter_out - 624 ns;
          p := 15 ns;
      else p:= 0 ns;
      end if;
    delay <= 2500 ns - ((filter_out / 10000 ns) * 1 ns) + p;
  end process cp;
 Q_a <= NOT Q_a AFTER delay;
 Q <=Q_a;
  low_det <= low_det_int;
end RTL;
```

Listing 1: Auszug aus dem nachgebildeten VCO

Danach wird die Kennlinie des VCO aufgenommen und durch Zahlenwerte in VHDL (Listing 1) nachgebildet. Das "filter_out" - Signal dient als Grundlage zur Berechnung der Verzögerungszeit. Mit dem Befehl "delay" und dem errechneten Zahlenwert läßt sich dann die Erzeugung der VCO-Frequenz durch "Q_a<= NOT Q_a AFTER delay " realisieren.

Der VHDL-Code ist nur ein Behaviour - Model. Er kann nicht synthetisiert werden. Durch vergleichende Simulation von Analogteil und VHDL - Model wurde eine Verifikation vorgenommen.

3.2. Stabilität

Bei dem gewählten Teilerverhältnis von 1:338 ist das Verhältnis der Loop-Bandbreite zu der Abtastrate von 32,768kHz immer noch relativ klein, dass es so zu Stabilitätsproblemen des VCO kommt. Die Loop-Bandbreite muss also deutlich verringert werden, was nur möglich ist durch:

-Verringerung des Stromes der UP/DOWN-Stromqellen
-Vergrößerung des Kondensators C_L
-Verdopplung der Vergleichsfrequenz durch Auswertung der steigenden und der fallenden Flanke.

Die Änderung des Kondensators, der mit 60pF sowieso das größte Bauteil darstellt, verbietet sich wegen des großen Flächenverbrauchs. Das Vermindern des Ladestromes ist direkt auch nicht möglich, da schon am unteren Ende der Möglichkeiten gearbeitet wird. Bei weiterem Absenken des Stromes nehmen die parasitären Effekte zu starken Einfluss.

Die realisierte Lösung verwendet deshalb ein Laden/Entladen des Schleifenkondensators C_L in Form von kurzen, etwa 90 ns langen Impulsen, die aus dem VCO-Signal durch eine sequentielle Logik abgeleitet werden.

Die Verdoppelung des konnte einfach durch Verdoppelung des Phasendetektors, der einmal direkt und einmal invertiert angesteuert wird, und Kombination der Ausgangssignale realisiert werden.

Der Nachteil dieser Lösung ist allerdings, dass das Führungssignal von 32kHz ein Taktverhältnis von 50% aufweisen muss, was bei dem verwendeten Quarzoszillator annähernd der Fall ist.

3.3. Taktbegrenzer

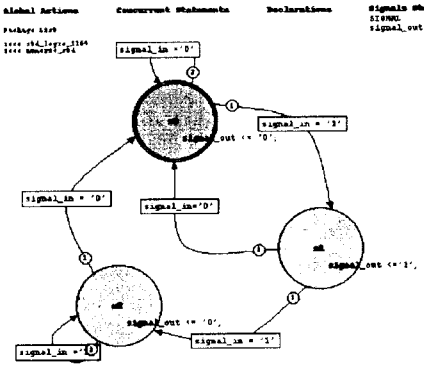


Abbildung 8: Statediagramm des Taktbegrenzers

Der Taktbegrenzer ist eine Statemachine mit drei Zuständen. Zustand "S0" ist der Ruhezustand. Es erfolgt keine Ladung des Kondensators. Zustand "S1" ist der Ladezustand. Die Statemachine springt in diesen Zustand, wenn vom Phasendetektor ein Flankenwechsel von '0' auf '1' kommt. Nur in diesem Zustand erfolgt eine Änderung der Kondensatorladung. Dieser Zustand ist genau eine 11MHz-Taktzeit lang . Zustand "S2" ist der Wartezustand. Die Statemachine verharrt hier so lange, bis der Ladebefehl des Phasendetektors wieder zu '0' wird.

Um aber noch akzeptable Startzeiten zu erreichen, muss der Taktbegrenzer überbrückt werden können. Das Steuersignal hierfür ist das "low_detect" vom VCO. Wenn das "low_detect" '0' ist, wird der Taktbegrenzer unwirksam.

Das Umladen des Loop-Kondensators C_L in diskreten Ladungspaketen ΔQ entspricht regelungstechnisch einer Anstiegsgeschwindigkeitsbegrenzung (Slew Rate) mit dem Effekt einer Bandbreitenverringerung.

3.4. Phasendetektor

Der Phasendetektor vergleicht die beiden Phasen. Er besteht aus 2 flankengesteuerten, asynchronen Flipflops [Eshr], die gegeneinander verschaltet einen Zustandsautomaten ergeben. Der Aufbau besteht aus Standardgattern. Die Laufzeiten durch diese Gatter sind durchaus kritisch, wodurch sich eine Technologie und Systemabhängigkeit ergibt. Für die hier verwendete Technologie wurde die Funktionalität verifiziert.

Bei einer Phasenverschiebung zwischen Referenzfrequenz und VCO soll der Phasendetektor die Signale so ausgeben, dass die Phasenverschiebung korrigiert wird.

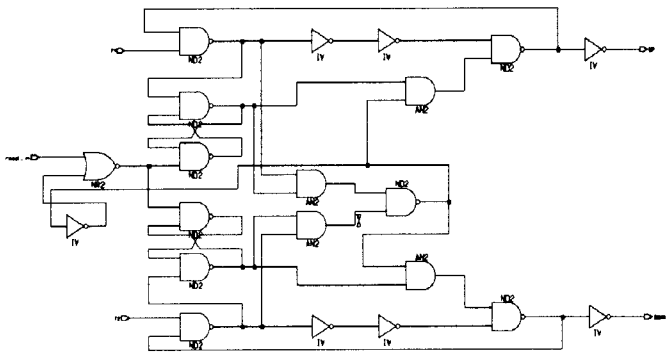


Abbildung 9: Schaltplan des Phasendetektors

Es werden 2 Phasendetektoren gleichen Aufbaus verwendet. Der erste Phasendetektor arbeitet nur auf der fallenden Flanke der Eingangssignale, der 2 PD auf der steigenden Flanke. Die Ausgangssignale beider Detektoren werden so miteinander kombiniert, dass sowohl ein Frequenzvergleich als auch ein Phasenvergleich erfolgt. (Abb. 10)

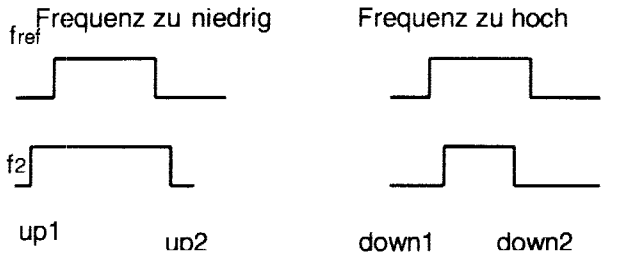


Abbildung 10: Zusammenarbeit der Phasendetektoren

Der zweite Detektor verbessert die Eigenschaften des PLL erheblich, da bei kleinen Frequenzunterschieden die beiden Detektoren gegeneinander arbeiten.

4. Realisiertes System

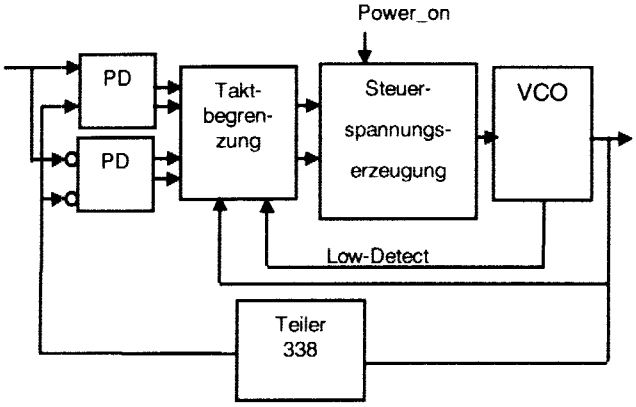


Abbildung 11: Blockdiagramm des Systems

Der Digitalteil besteht nun aus einem Teiler durch 338, den beiden Phasendetektoren, vier Taktbegrenzern und Logik zur Zusammenschaltung der Ausgangssignale der Taktbegrenzer. Auch ist diese Logik dafür zuständig, dass bei dem entsprechenden "low-detect"-Signal die Taktbegrenzer überbrückt werden.

Der Analogteil besteht aus der Steuerspannungserzeugung und dem VCO.

5. Routing

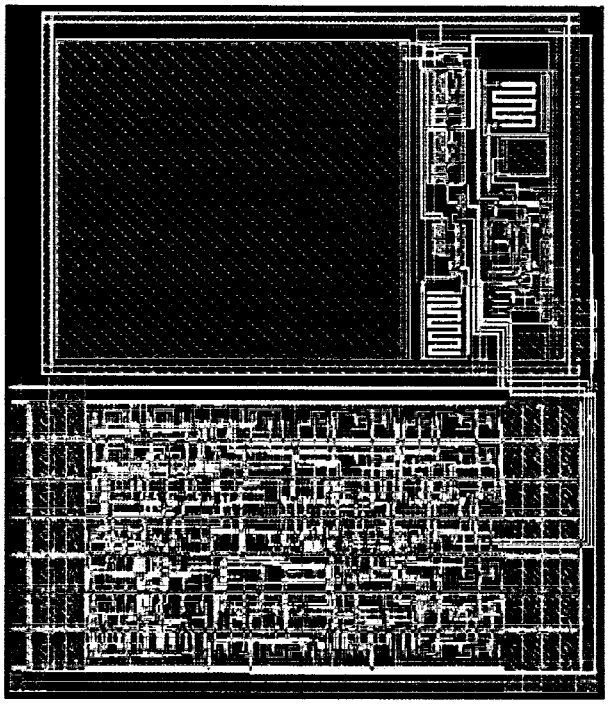


Abbildung 12: Routing des PLL

Der PLL wurde in Mietec 0,5µm geroutet. In dieser Technologie stehen 3 Metallisierungslagen zur Verfügung. Kondensatoren können mit Hilfe einer zweiten Poly-Lage realisiert werden.

Das Routing umfasst den ca 0,09mm² großen Analogteil und den Digitalteil mit ca. 0,08mm² . Die Gesamtgröße ergibt somit ca 0,2mm².

Der Digitalteil wurde mit AutoCells geroutet. Zum Routen des Anologteils und Zusammensetzen der beiden Blöcke wurde die IC-Station verwendet.

6. Zusammenfassung

Aufgabe	Frequenzvervielfachung
Randbedingung	Low-Power
Zieltechnologie	Mietec 0,5µm CMOS
Ergebnis	Mixed-Signal-Design
	Phasenrauschen <10kHz (simuliert)
	Abschaltbar
	Stromverbrauch: wenige µA(Betrieb), <1µA (abgeschaltet)
	Größe 0,2mm²
	Andere Frequenzen durch Änderung der Teilerverhältnisse

7. Literatur:

[Titz] Titze Schenk: Halbleiterschaltungstechnik, Springer Verlag

[Jan] Dirk Jansen: Vorlesungsskript Elektronische Schaltungstechnik

[Hau] Jürgen Hauser: Projektbericht Entwicklung eines PLL zur Generierung eines 11MHz-Taktes aus einem 32kHz-Takt

[Best] Best: Theorie und Anwendung des Phase Locked Loops; AT-Verlag

[Eshr] Weste/Eshragian: Principles of CMOS VLSI design; Edinson Wesley

Konfigurierbarer 14-Bit Sigma-Delta-Wandler

Markus Schnitzer
Fachhochschule Darmstadt, Haardtring 100, 64295 Darmstadt

1. Aufgabenstellung

Im Rahmen dieser Diplomarbeit wurde ein A/D-Wandler nach dem Sigma-Delta-Verfahren entworfen, der bei gleichem Dynamikbereich in der Auflösung und im Frequenzbereich durch geeignete Maßnahmen konfigurierbar ist. Es mußte ein Modell entwickelt werden, das den oben genannten Anforderungen genügt.
Dieses Modell wurde dann auf einen bestimmten Auflösungs- und Frequenzbereich ausgelegt und durch die Hardwarebeschreibungssprache VHDL umgesetzt. Durch das Simulations Programm *ModelSim* der Firma Modeltech bestand die Möglichkeit, den VHDL-Entwurf des konfigurierbaren Sigma-Delta-Wandlers zu überprüfen.

2. Grundlagen des Sigma-Delta-Wandlers

Der Sigma-Delta-Wandler wird im Gesamtsystem mit den dazugehörigen Komponenten betrachtet. Zu diesen Komponenten gehören der Modulator, der Dezimierer und der Tiefpaß. Da bei der Betrachtung der Konfigurationsmöglichkeiten nur die Komponenten *Sigma-Delta-Modulator* und *Dezimierer*, aufgrund ihrer Wandlerparameter wie Abtastrate *OSR*, Abtastfrequenz f_A, Nyquistfrequenz f_N, Dezimierungsfaktor D_{Dez} und Wortfaktor F_{Dez}, eine wichtige Rolle spielen, wird die Komponente *Tiefpaß* vernachlässigt.

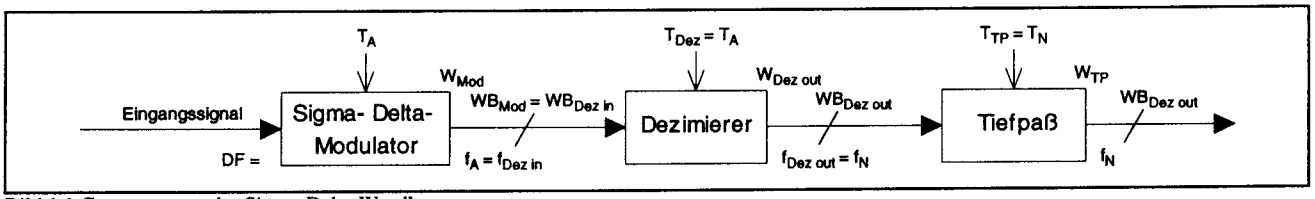


Bild 1.1 Gesamtsystem des Sigma-Delta-Wandlers

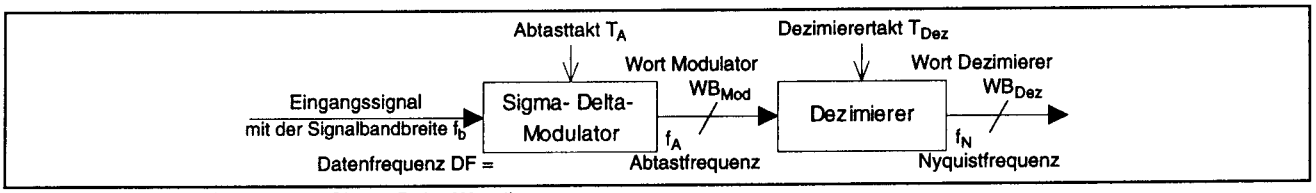


Bild 1.2 Vereinfachte Struktur des Sigma-Delta-Wandlers

1. Modulator [1]
   Mit dieser ersten Komponente wird das analoge Eingangssignal mit der Signalbandbreite f_b durch eine hohe Abtastfrequenz f_A abgetastet, so daß pro Abtasttakt T_A ein Bit entsteht.
   Durch diesen Modulator erhält man eine serielle Bitfolge, bei der die Breite der Pulse proportional zur Amplitude der angelegten Eingangsspannung ist (**P**uls**D**ichten**M**odulation).

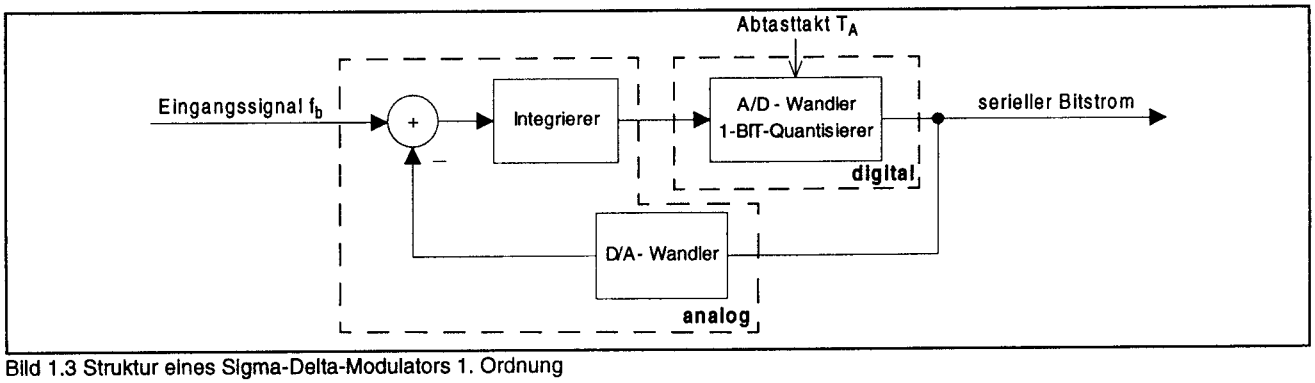


Bild 1.3 Struktur eines Sigma-Delta-Modulators 1. Ordnung

2. Dezimierer [2]
   Durch den Dezimierer wird die Datenfrequenz f_A der Modulator-Ausgangsdaten auf die Nyquistfrequenz $f_N \geq 2 \cdot f_e$ reduziert. Dabei entspricht die Reduktion einer Entnahme der Ausgangsdaten des Dezimierers bei jedem Nyquisttakt T_N.
   Zudem werden mit dem Dezimierer die 1-Bit breiten Ausgangsworte des Modulators in die Wortbreite WB_{Dez} umgewandelt. Es ergeben sich somit Datenworte, wie sie in herkömmlichen A/D-Wandlern vorhanden sind.

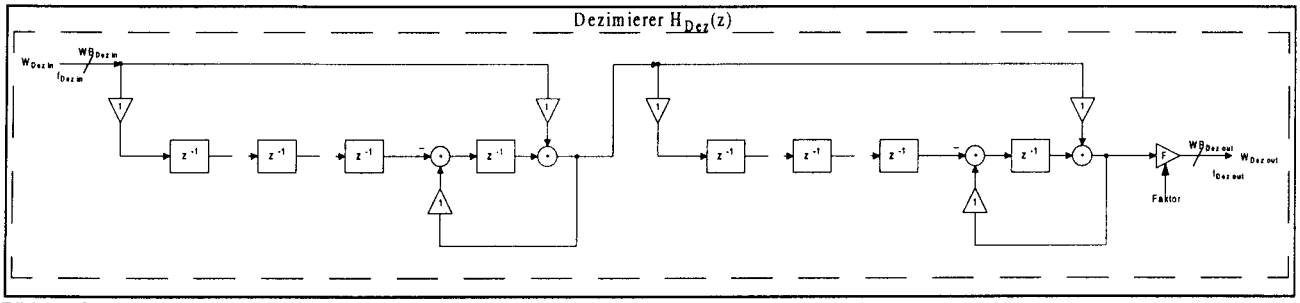


Bild 1.4 Struktur des Dezimierers

3. Konfigurationsmöglichkeiten des Sigma-Delta-Wandlers

Anhand des vereinfachten Systems des Sigma-Delta-Wandlers werden die Konfigurationsmöglichkeiten an den Komponenten *Modulator* und *Dezimierer* untersucht.
Dabei betreffen die Konfigurationsmöglichkeiten die Auswirkungen auf Abänderungen der internen wandlerspezifischen Parameter, wie Abtastrate *OSR*, Abtastfrequenz f_A an der Komponente *Modulator*, Wortfaktor F_{Dez} und Dezimierungsfaktor D_{Dez} an der Komponente *Dezimierer*. Außerdem betrifft die Konfigurationsmöglichkeit die Auswirkung auf externe Abänderung der Signalbandbreite f_b.

Konfigurationsmöglichkeiten an der Komponente *Modulator*:
- Abänderung der Abtastfrequenz f_A bei konstanter Signalbandbreite f_b
  => Auswirkung auf Auflösungsbereich
- Abänderung der Abtastfrequenz f_A bei konstanter Abtastrate OSR
  => Auswirkung auf Frequenzbereich
- Abänderung der Signalbandbreite f_b bei konstanter Abtastfrequenz f_A
  => Auswirkung auf Auflösungs-, Frequenzbereich
- Abänderung der Signalbandbreite f_b bei konstanter Abtastrate *OSR*
  => Auswirkung auf Frequenzbereich

Konfigurationsmöglichkeiten an der Komponente *Dezimierer*:
- Abänderung des Wortfaktors F_{Dez} bei konstantem Dezimierungsfaktor D_{Dez}
  => Auswirkung auf Auflösungsbereich
- Abänderung des Dezimierungsfaktors D_{Dez} bei konstantem Wortfaktor F_{Dez}
  => Auswirkung auf Frequenzbereich

Gleichungen:

$$OSR = \frac{f_A}{2 \cdot f_b}$$

$$WB_{\Sigma\Delta} = \frac{10 \cdot \log\left(\frac{3(2n+1)}{2\pi^{2n}}\right) dB + (2n+1) \cdot 10 \cdot \log(OSR) dB - 1{,}76 dB}{6{,}02 dB}$$

$$W_{\Sigma\Delta} = 2^{WB_{\Sigma\Delta}}$$

$$W_{Dezoutmax} = W_{\Sigma\Delta} = W_{Mod} \cdot F_{Dez} = W_{Mod} \cdot m^k \cdot F$$

$$0 \leq WeBe < W_{Dezoutmax}$$

$$WeBe = 2^{WB_{\Sigma\Delta}} - 1$$

$$f_A = 2 \cdot OSR \cdot f_b$$

$$f_N \geq 2 \cdot f_b$$

$$f_N = f_A \cdot D_{Dez} = f_A \cdot \frac{1}{k \cdot m}$$

$$D_{Dez} = \frac{1}{OSR}$$

$$D_{Dez} = \frac{1}{k \cdot m}$$

n=Ordnung Modulator
WB=Wortbreite
W=Wort
WeBe=Werte-Bereich

4. Modell eines konfigurierbaren Sigma-Delta-Wandlers

Die kursiv gedruckten Abänderungen wurden dazu genutzt, ein Modell des Sigma-Delta-Wandlers zu entwerfen, bei dem der Auflösungs- und Frequenzbereich getrennt voneinander variiert werden kann.

4.1 Modellentwicklung für variablen Auflösungsbereich

Mit der Änderung der Abtastfrequenz f_A bei konstanter Signalbandbreite f_b am Modulator ergibt sich eine Auswirkung auf die Abtastrate *OSR* und damit auf den Dynamikbereich.

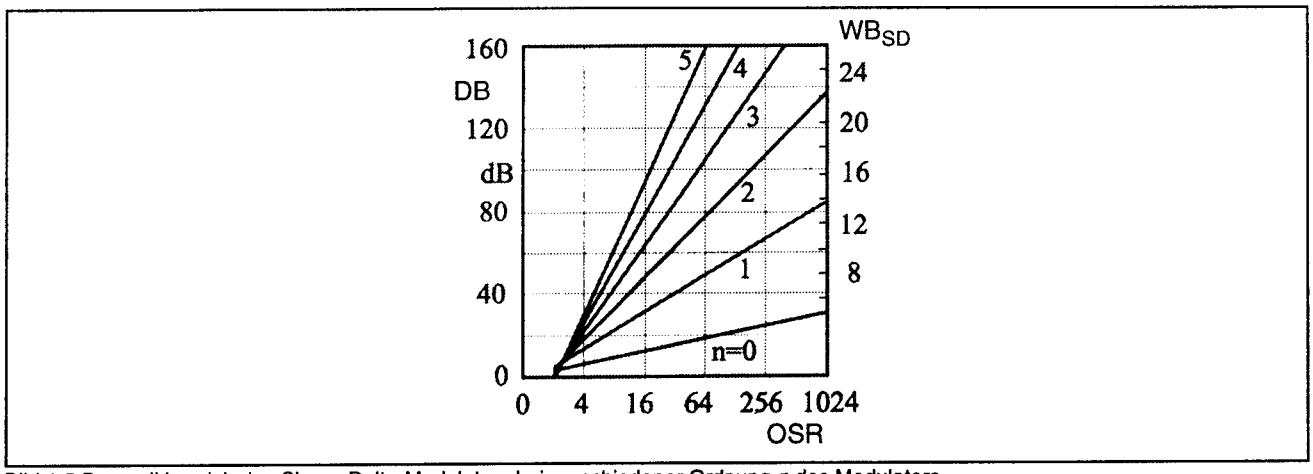
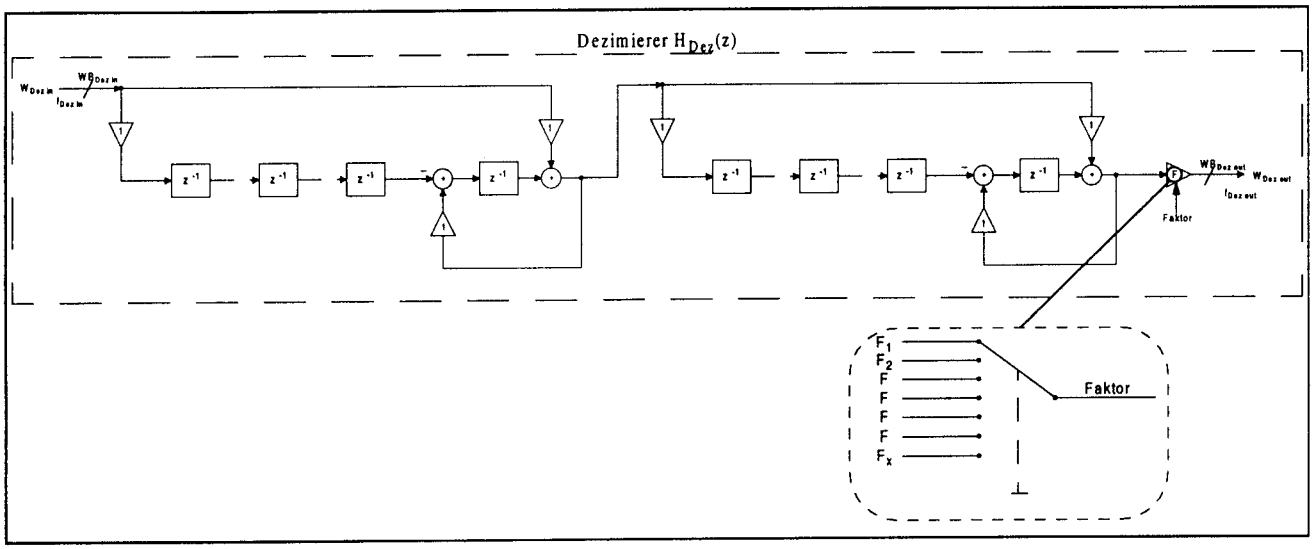


Bild 1.5 Dynamikbereich des Sigma-Delta-Modulators bei verschiedener Ordnung *n* des Modulators

Dies kann dazu genutzt werden, um mit der Abtastrate *OSR* die erreichbare Wortbreite $WB_{\Sigma\Delta}$ festzulegen.

Durch die Wortbreite $WB_{\Sigma\Delta}$ ist zugleich das maximal erreichbare Ausgangswort $W_{Dez\ out\ max}$ und die durch dieses maximale Ausgangswort gekennzeichnete Grenze des Wertebereichs $0 \leq WeBe < W_{Dezoutmax}$ für $U_{in} < U_{FS}$ definiert.

Der Dynamikbereich und die damit verbundene Wortbreite wird beibehalten.
Durch die Änderung des Faktors *F* des Wertefaktors F_{Dez} am Dezimierer ist nun die Möglichkeit gegeben, durch einen wählbaren Faktor das Ausgangswort des Dezimierers $W_{Dezout} = W_{Dezin} \cdot F_{Dezin} = W_{Dezin} \cdot m^k \cdot F$, und damit den Wertebereich, innerhalb der durch das maximal erreichbare Ausgangswort $W_{Dez\ out\ max}$ vorgegebener Grenze $0 \leq WeBe < W_{Dezoutmax}$, zu variieren



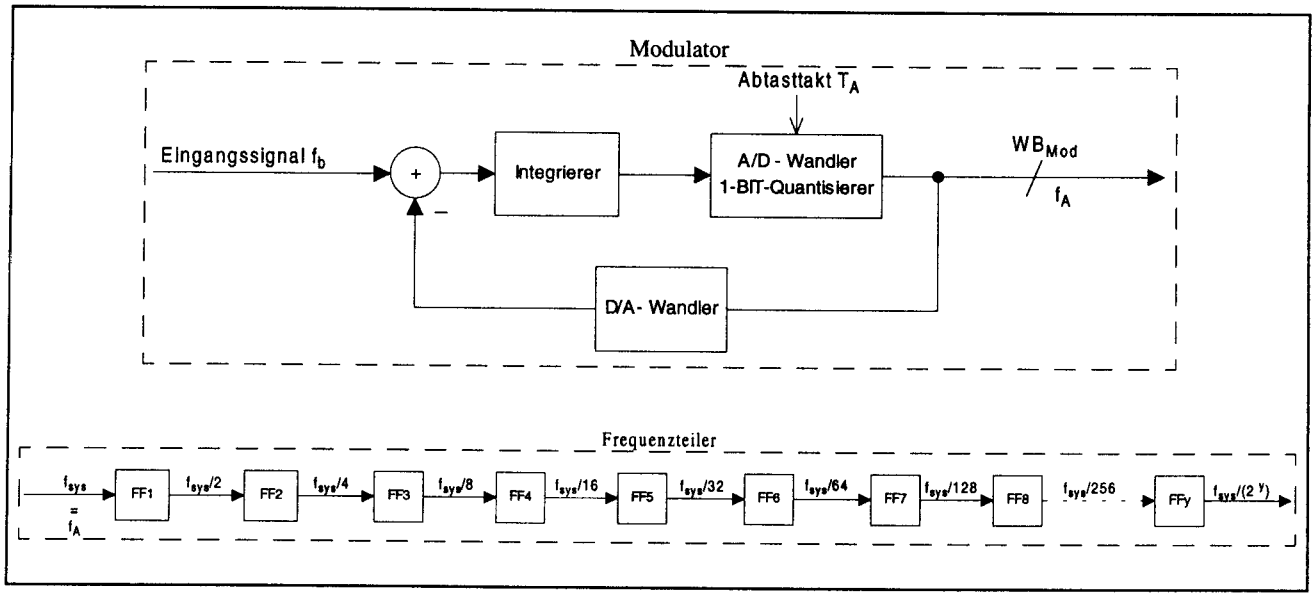
4.2 Modellentwicklung für variablen Frequenzbereich:

Bei vorgegebener Signalbandbreite f_b kann durch die Änderung des Dezimierungsfaktors D_{Dez} die Anzahl der Verzögerungselemente m für die aus der Signalbandbreite sich ergebende Nyquistfrequenz f_N einmal festgelegt werden.

Mit der Abänderung der Abtastfrequenz f_A bei konstanter Abtastrate *OSR* besteht nun die Möglichkeit mit der einmal festgelegten Struktur des Dezimierers und damit der gleichen Anzahl der Verzögerungselemente m eine andere Signalbandbreite $f_b = \frac{f_A}{2 \cdot OSR}$ zu verarbeiten.

Abtastfrequenz f_A	Abtastrate OSR	Signalbandbreite f_b
2,048 MHz	1024	1 kHz
4,096 MHz	1024	2 kHz
8,192 MHz	1024	4 kHz
16,384 MHz	1024	8 kHz
32,768 MHz	1024	16 kHz
65,536 MHz	1024	32 kHz

Mit der Verwendung eines Frequenzteilers, der die Abtastfrequenz f_A herunterteilt, wird die Entnahme der Daten zu den bestimmten Datenfrequenzen zur Weiterverarbeitung gewährleistet.



5. Auslegung des konfigurierbaren Sigma-Delta-Wandlers auf 14-Bit

Das zuvor entwickelte Modell soll nun für eine abstufbare Auflösung bzw. Wortbreite von maximal 14-Bit bis minimal 4-Bit und einer Signalbandbreite von 4 kHz ausgelegt werden. Dazu müssen die Parameter des Modulators und die Parameter des Dezimierers festgelegt werden.

5.1 Festlegung der Parameter des Modulators

Bei der Umsetzung des konfigurierbaren Modells des Sigma-Delta-Wandlers mit einem Modulator erster Ordnung $n=1$ für $WB_{\Sigma\Delta}=14$ ist nach $OSR=\frac{WB_{\Sigma\Delta}\cdot 6{,}02dB+1{,}76dB-10\cdot\log\left(\frac{3(2n+1)}{2\pi^{2n}}\right)dB}{10\cdot(2n+1)}$ eine Abtastrate von 1024 notwendig.

Bei einer Signalbandbreite von 4 kHz ergibt dies nach $f_A=2\cdot OSR\cdot f_b$ eine Abtastfrequenz von 8,192 MHz.

Mit der Wortbreite $WB_{\Sigma\Delta}=14$ läßt sich das Wort $W_{\Sigma\Delta}=2^{WB_{\Sigma\Delta}}=2^{14}=16384_{(10)}$ erreichen.

Damit ist zudem der Wertebereich $WeBe=2^{WB_{\Sigma\Delta}}-1=2^{14}-1=16383_{(10))}$ festgelegt.

5.2 Festlegung der Parameter des Dezimierers

Für die Reduktion des Dezimierers von der Datenfrequenz $f_A=8{,}192MHz$ am Eingang des Dezimierers auf die Datenfrequenz $f_N=4\cdot f_b=16kHz$ ist ein Dezimierungsfaktor von $D_{Dez}=\frac{2}{1024}=\frac{1}{512}=\frac{1}{k\cdot m}$ notwendig.

Da der Faktor *m*k* für die benötigten Verzögerungselemente steht, wurde somit deutlich, daß mit einem Dezimierer 512 Verzögerungselemente, d.h. Register mit einer bestimmten Wortbreite verwendet werden müßten. Durch eine Aufteilung in zwei Dezimierer mit einer geschickten Verteilung der Datenfrequenzen an den Dezimierer 1 und Dezimierer 2 gelang es den Registeraufwand von 512 auf 48 Registern zu reduzieren.

Möglichkeit	$f_{Dez\ in} = f_A$	$f_{Dez1\ out}$	$f_{Dez\ out} = f_N$	D_{Dez1}	D_{Dez2}	D_{Dez}	$Register_{ges}$
1	8,192 MHz		16 kHz	1/512		1/512	512
2	8,192 MHz	32 kHz	16 kHz	1/256	1/2	1/512	258
3	8,192 MHz	64 kHz	16 kHz	1/128	1/4	1/512	132
4	8,192 MHz	128 kHz	16 kHz	1/64	1/8	1/512	72
5	*8,192 MHz*	*256 kHz*	*16 kHz*	*1/32*	*1/16*	*1/512*	*48*
6	8,192 MHz	512 kHz	16 kHz	1/16	1/32	1/512	48

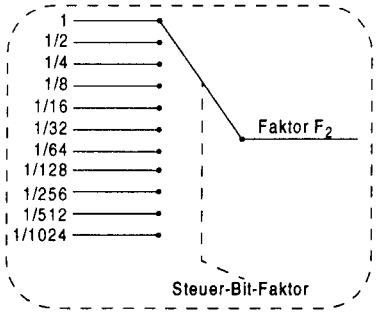
Für die Ordnung *k*, der dem Modulator nachgeschalteten Komponenten, muß mindestens $k=n+1$ (mit n=Ordnung des Modulators) erfüllt sein, damit der erzielbare Rauschabstand im Digitalteil nicht verschlechtert wird. Um dies zu erfüllen, wurde die Ordnung *k* der Dezimierer auf $k=2$ festgelegt.

Als nächstes wurden mit dieser Aufteilung die Wortfaktoren der einzelnen Dezimierer bestimmt.
Der sich daraus ergebende Wortfaktor F_{Dez} des in zwei aufgeteilte Dezimierer wird dazu genutzt die Faktoren F_1, F_2 derart festzulegen, so daß eine Abstufung von 14-Bit auf 4-Bit vorgenommen werden kann.

$$F_{Dez1} = m_1^{\,k} \cdot F_1 = 16^2 \cdot F_1 = 256 \cdot F_1 \qquad F_{Dez2} = m_2^{\,k} \cdot F_2 = 8^2 \cdot F_2 = 64 \cdot F_2$$

$$F_{Dez} = F_{Dez1} \cdot F_{Dez2} = 256 \cdot F_1 \cdot 64 \cdot F_2 = 16384 \cdot F_1 \cdot F_2$$

Der Faktor F_1 wird auf 1 festgelegt und der Faktor F_2 wird von 1 bis 1/1024 wählbar gestaltet.



Im Wertebereich $WeBe = 2^{WB_{\Sigma\Delta}} - 1$ ergibt sich mit dem Faktor von 1 der Wertebereich von $0_{(10)}$ - $16383_{(10)}$ und mit dem Faktor 1/1024 der Wertebereich von $0_{(10)}$ - $15_{(10)}$.

Das entwickelte Modell des konfigurierbaren Sigma-Delta-Wandlers wurde anschließend in VHDL programmiert. Mit dem Tool *ModelSim* der Firma Mentor Graphics wurde es simuliert, um somit die Funktionsweise nachzuprüfen.

6. Literaturverzeichnis

[1] Prof. Dr. Wolfgang Reinhold, Thomas Wurlitzer
Sigma-Delta-ADC in Software
Design & Elektronik S. 90 - 99, Ausgabe 10.1998

[2] Steffen Becker, Technische Universität Dresden
Diplomarbeit Entwurfskonzept für Sigma-Delta-A/D-Wandler, 1996

Allgemeine Literaturhinweise:

James C. Candy, Bruce A. Wooley and Oconnell J. Benjamin
A Voiceband Codec with Digital Filtering
IEEE Trans. Circuits Syst., vol. COM-29,pp.. 815 - 830, June 1981

J. C. Candy
Decimation for sigma delta modulation
IEEE Trans. Commun., vol. COM-34, pp. 249 - 258, Januar 1986

S. Chu and C. S. Burrus
Multirate filter design using comb filters
IEE Trans. Circuits Syst., vol. CAS-31, pp. 913 - 924 November 1984

The Parallel Hough-Transform Systolic Array ASIC

A. Epstein[1,2], G. U. Paul[2], B. Vettermann[2], C. Boulin[1] and F. Klefenz[3]

1) European Molecular Biology Laboratory, Postfach 102209, 69012 Heidelberg

2) Fachhochschule-Mannheim, Windeckstrasse 110, 68163 Mannheim

3) Q2, 23, 68161 Mannheim

Abstract

Many pattern recognition problems can be solved by mapping the input data into an n-dimensional feature space in which a vector indicates a set of attributes (Ohlson, 1992). One powerful pattern recognition method is the Hough-Transform. In reducing the n-dimensional feature space to two dimensions the coordinate transform can be executed by a systolic array consisting of time-delay processing elements and adders (Durbin, 1990). The ASIC implementation of the Hough-Transform as a systolic array for real-time recognition of curved tracks in multi-wire drift chambers is presented. The array can handle 32 parallel input data streams. It mainly consists of 512 identical programmable processing elements. 16 histogram pixels in the feature space are produced in parallel per clock cycle. The ASIC is implemented in 0.6µm CMOS, 2 metal layer technology (CUB) from AMS and will operate with a clock frequency of 100 MHz. The interconnectivity pattern of the processing elements required to initialise the chip according to the pattern recognition task is computed on the host computer using the Hough-Transform equations. This pattern is then downloaded to the chip via the data input lines. The Hough-Transform ASIC is suitable for a wide range of pattern recognition applications. The integrated Hough-Transform circuit will allow the real-time execution of this transform with an increase in speed in the range of two orders of magnitude compared to previous implementations.

1. The Hough Transform

The Hough Transform is a coordinate transform, which maps the input data directly to an n-dimensional feature space, in which the aggregating clusters indicate the occurrence of a feature. The attributes of a feature are quantitatively coded by the corresponding n-dimensional feature vector. The feature attributes are mapped linearly along the orthogonal feature axes.

The power of the Hough transform derives from the linearity of the feature maps. Input tuple coordinates and feature coordinates are coupled by the corresponding Hough transform equations (Ballard, 1981). The Hough transform equations can be given analytically for simple patterns like straight lines, circles and trigonometric functions (Duda, 1972). Each input tuple is translated to its associated trajectory in the corresponding feature space. Multiple crossings of trajectories in the feature space indicates that a feature forming input tuple set belongs to the same feature.

The multiple intersections of the trajectories leads to clustering in the feature space. However, the intersection density peaks sharply for the best possible fit of an observed feature, therefore a single feature is represented in the feature space as a point distribution with characteristic decreasing profile (Davis, 1992).

The Hough transform has been used originally in the field of high energy physics for the detection of charged particle tracks in bubble chambers (Hough, 1959). Since then the Hough transform has been used as a standard image analysis tool for pattern recognition (Hough, 1962). The Hough transform algorithm is known to be computational intensive (Van Swaaij, 1990).

1.1 Parametric representation of lines

For simplicity, the principle of the Hough transformation will be described for a straight line.

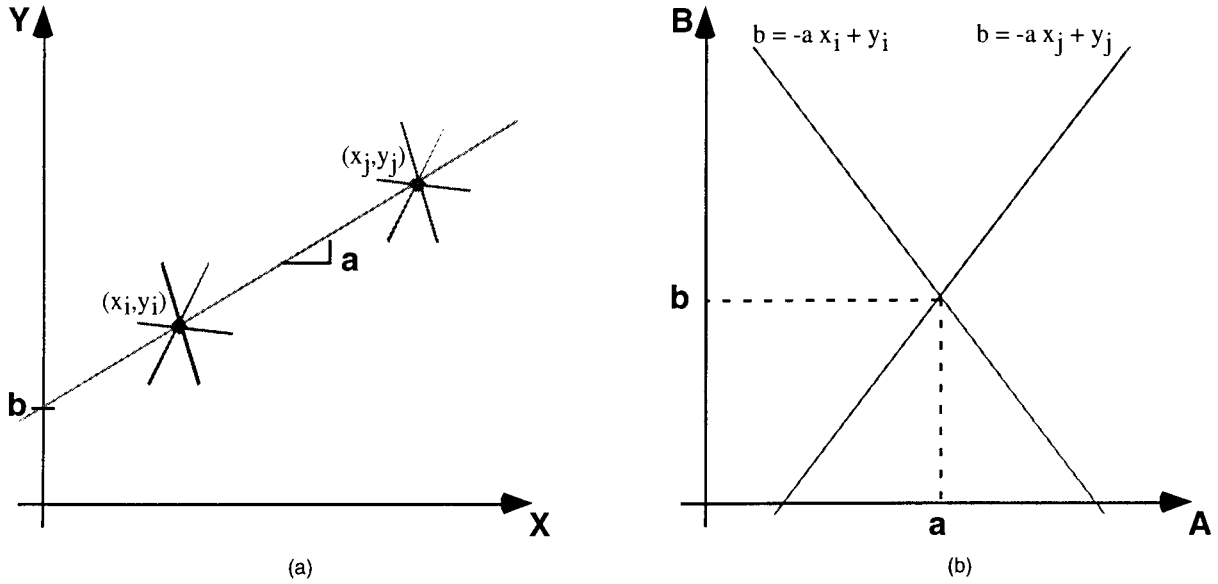


Figure 1.1: Representation of Hough Transform: left is the data space and right the parameter space.

One may consider a point P_i (x_i,y_i). The general equation for a straight line crossing this point is:

$$y_i = a\,x_i + b$$

Infinite number of straight lines of this form cross the point Pi, with different values of a and b. These can be represented in the (a,b) space, or the 'parameter space' or 'feature space', as:

$$b = -a\,x_i + y_i$$

Each point P_j (x_j,y_j) in the (x,y) space will similarly have an infinite number of straight lines crossing it which are represented in the parameter space as a second straight line.

The crossing point of both lines represent the straight line (in the (x,y) space) which has the same slope a and Y-axis crossing point b. This straight line crosses both P_i and P_j. This is demonstrated in figure 1.1.

1.2 Representation of traces in the OPAL detector

Charged particles travel in a homogenous magnetic field in a circular track. Such tracks, which pass through the origin, can be described in polar coordinates. Each such circle is specified by a starting angle φ_s and a radius of curvature r_c. The starting angle is the tangential angle of the circle at the origin and the radius of curvature is the circles radius (figure 1.2).

From figure 1.2 one can see that the relation between (φ_s, r_c) and (r, φ) of a point on the track is:

$$\frac{r/2}{r_c} = \sin(\varphi - \varphi_s)$$

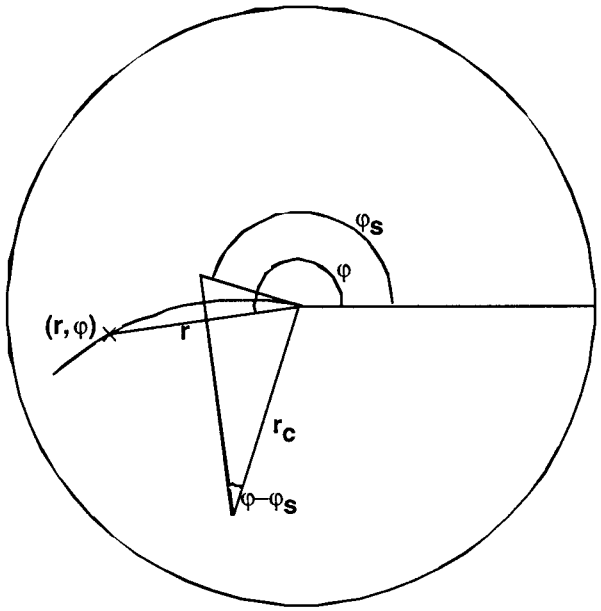


Figure 1.2: Mapping of (r, φ) and $(1/r_c, \varphi_s)$.

Each track is defined by a set of track points (r,φ). Each point (r,φ) in the detector, together with the origin, specifies a class of tracks with different φ_s and r_c which pass through both the origin and the point (figure 1.3 a). The following equation defines these tracks:

$$\frac{1}{r_c} = \frac{2}{r} \times \sin(\varphi - \varphi_s)$$

Each point (r,φ) in the detector can be represented as a sinus curve with amplitude 2/r and phase shift φ in the parameter space ($1/r_c$,φ_s), or the Hough-space (figure 1.3 b).

1.3 The Hough Algorithm

Each point in the (r, φ) space (figure 1.4 a) is translated into the Hough-space and represented there as a sinus curve as shown above (figure 1.4 b). If all the points share a common possible trace (circle) passing through each point and the origin, all the corresponding sinus curves will cross each other in a single point. The coordinates of this point in the Hough-space are the parameters of the common circle.

1.4 The discrete Hough Algorithm

In the discrete form, the Hough transform is a histogram accumulating technique. The feature space is subdivided into a grid of histogram cells, the number of which defines the granularity of the feature space. The Hough transform is therefore, in its discrete form, a histogram updating procedure in which for each point in the (r, φ) we update the histogram in the Hough-space. The result is a 2-D histogramm representing for each point ($1/r_c$,φ_s), the probability of the existence of such a track. Figure 1.5 shows the representation of an ideal track with r_c = 10 m and φ_s = 0 degree, with a sharp peak that falls exactly in the right bin.
Although very powerful, the Hough algorithm can not be easily used for real time applications because it is so computational intensive.
The processing required for each event is:

1. The calculation of a 2-D matrix.
2. The addition of two matrixes.
3. The detection of peaks in the resulting histogram.

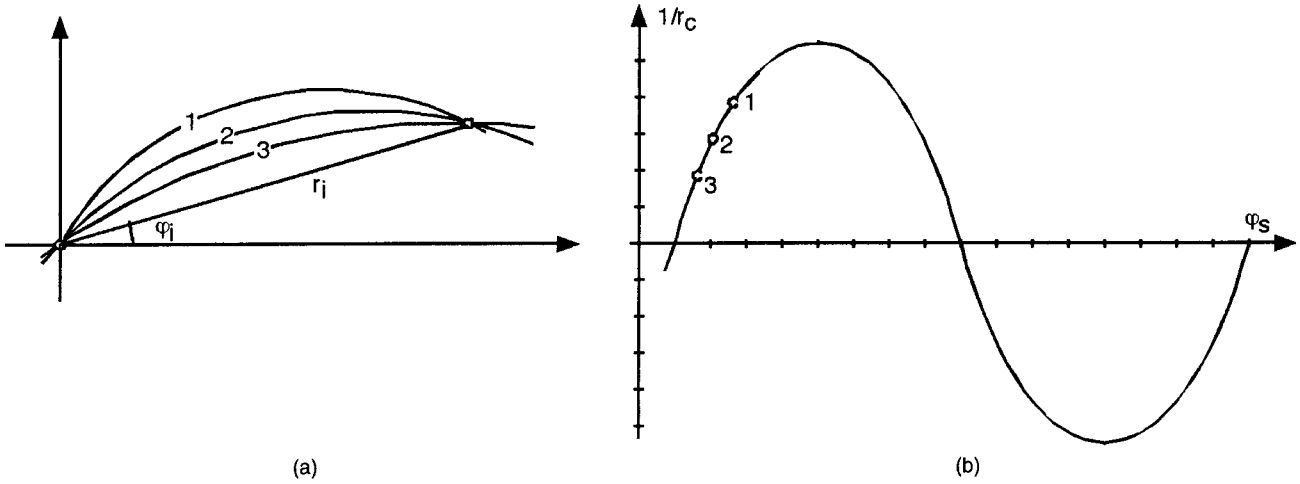


Figure 1.3: a) Three possible traces through the origin and the point (r_i,φ_i). B) The three possible traces mapped in the Hough space

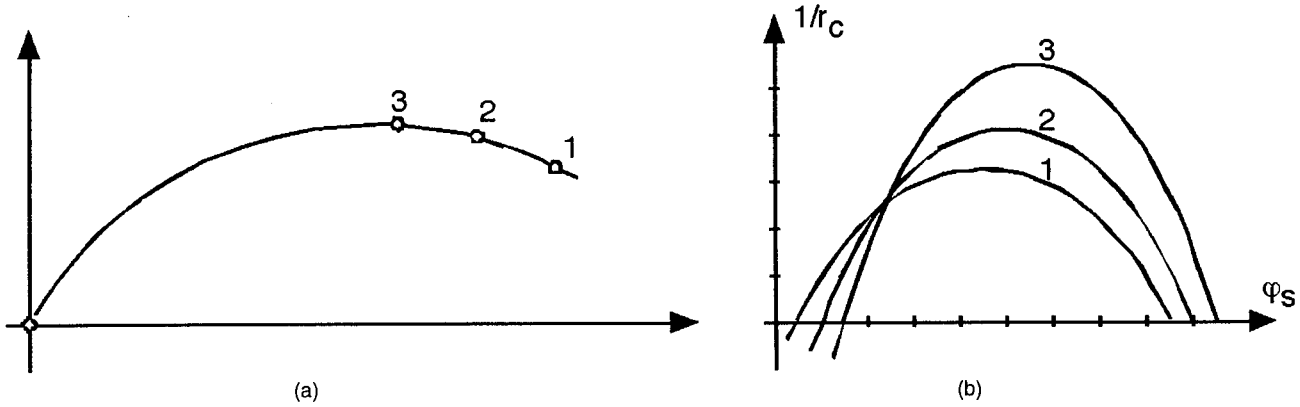


Figure 1.4: a) Three points which have a common possible trace. b) The representation in the Hough space of these three points. The intersection point of the three sinus curves represent the common trace.

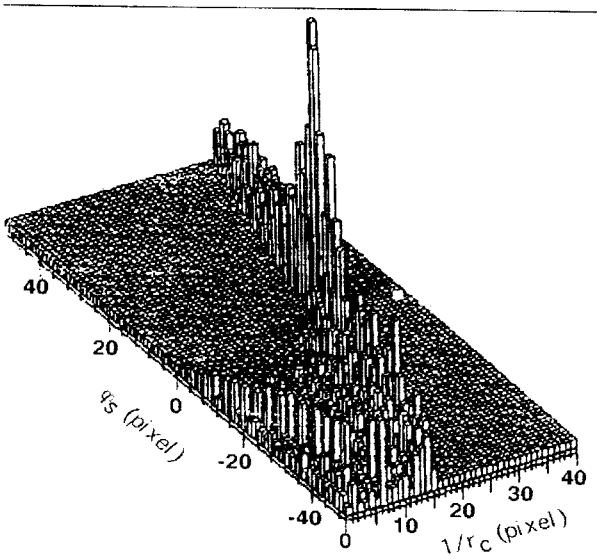


Figure 1.5: Example of a 2-D histogramm in the Hough-space. In this example the peak indicates a trace with rc=10 m (Klefenz 1992).

2. The Delay Line Method

The philosophy behind the fast Hough-transform ASIC is the detection of slopes or curvatures using an array of programmable delay elements. To demonstrate the principle of slope detection using tapped delay lines, we use here a set of 8 delay lines with linearly spaced propagation delay. The left most line has the highest propagation speed and the right one the lowest.

In figure 2.1a, we see the system at time t=0. The delay lines contain zeros and data is shifted in from the bottom. The data contains three distinct straight lines which will be shifted gradually into the delay line array. After the first line (horizontal) is shifted in, it forms a horizontal line at the first set of delay line taps (figure 2.1b).

The second line will enter the array within the next few clock cycles. After propagating up the delay lines for a while, it will change it's angle until it becomes also horizontal (figure 2.1c). By that time the first line assumes a negative slope. This is due to the fact that the points at the left propagate upwards faster then at the right. Similarly, the third line will get all it's points aligned a short while later and this will also occur at a higher tap (figure 2.1d). By adding the number of ones aligned at all the tap levels, one can detect when a straight line entered the matrix and what was it's slope. In a similar manner, one can use different delay line speeds to detect curvatures instead of slopes.

Figure 2.2 shows the complete Hough-transform system for one sector of the OPAL detector. The wire signals produced by a circular particle trace. Each wire has a known distance from the origin (r). The pulse's time delay is proportional to the distance between each wire and the trace. This delay can be corrected to be proportional to the points angle. The correction can be done in an FPGA and is redundant for very small angles.

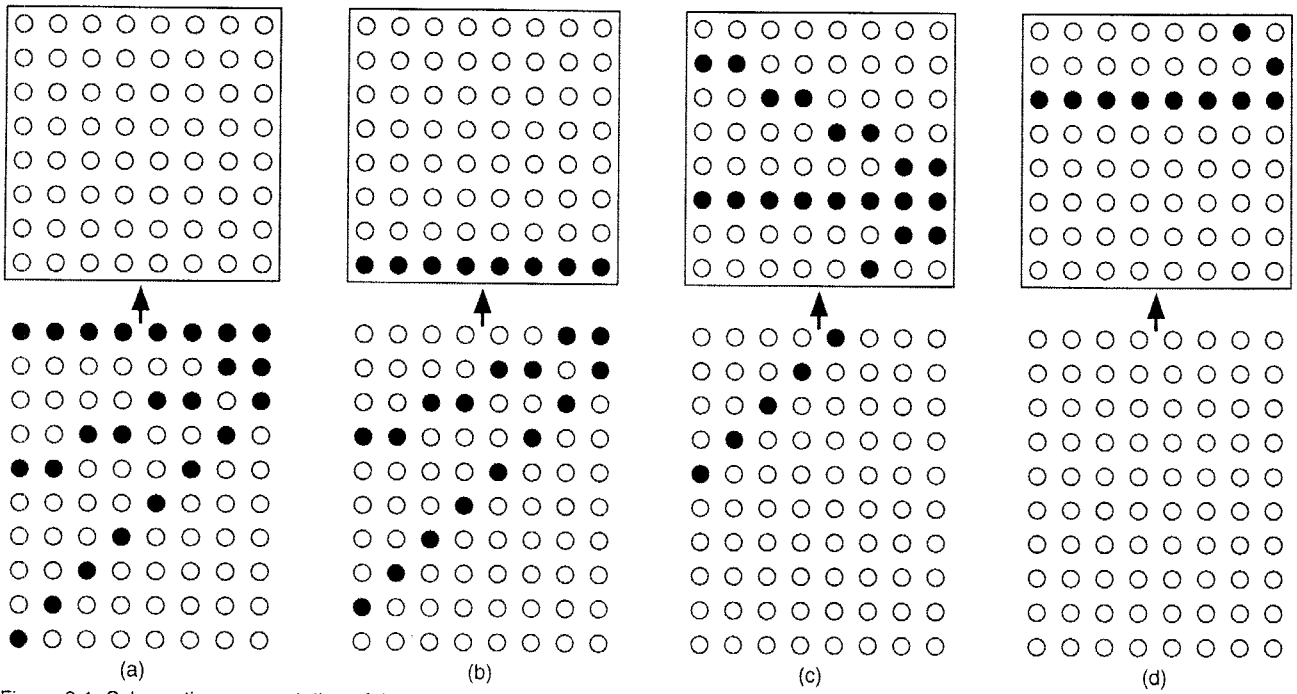


Figure 2.1: Schematic representation of the delay line method for detecting straight lines (and curvatures). At the left is the system at t=0.

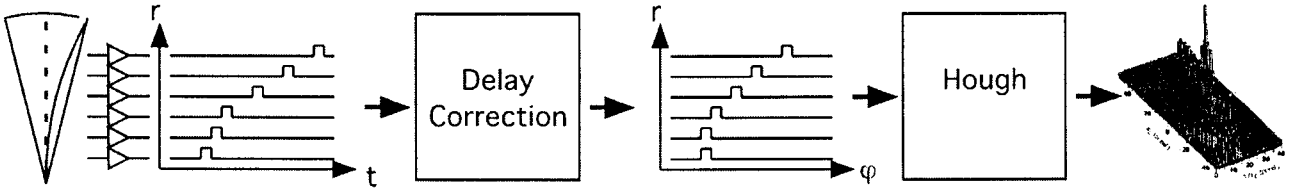


Figure 2.2: The Hough-Transform implementation for the OPAL drift chamber.

The corrected wire signals are then fed into a Hough-transform ASIC, based on programmable delay cells and adder elements to sum the aligned active pulses in a row. The output from the ASIC is one (r,j) histogram line per clock cycle.

3. The Hough Transform Processing Unit

3.1 Architecture

Figure 3.1 shows the data flow through the processing unit.

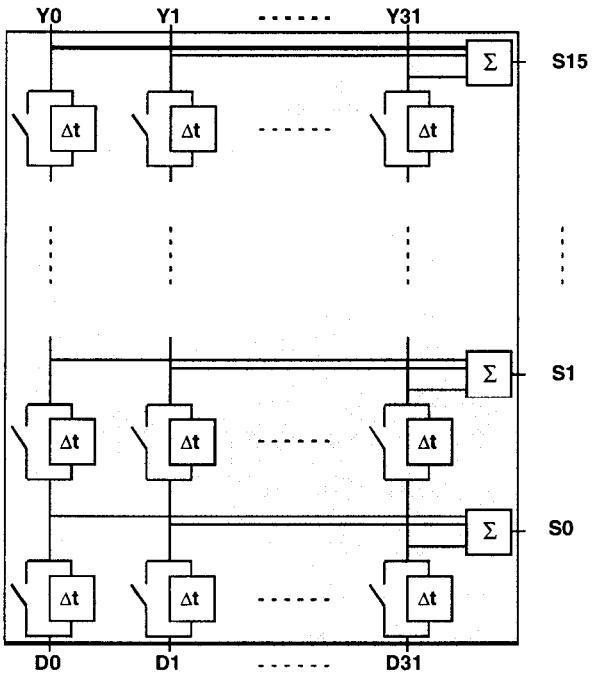


Figure 3.1: The Hough-Transform ASIC architecture

It consists of 512 programmable delay cells organised in 16 rows and 32 columns. An adder is associated with each row, producing the number of active bits aligned in the row at each cycle. The 16 6-bit sums are the line's histogram values.

The data flow within one column is determined by the configuration of the delay cells. Each cell can produce one or zero clock-cycles delay.

3.2 *The delay element*

Fig. 3.2 shows a functional diagram of the delay element. It consists of a flip-flop (FF) which is used to produce a one clock-cycle delay. The delay is switched off by bypassing the delay FF via a direct connection from the data input to the output. One more FF is used to store the configuration information (the switch's position). The configuration information is loaded though the data input and is shifted into the delay cells within 16-cycles of configuration.

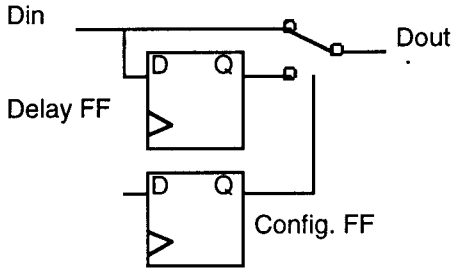


Figure 3.2: Functional diagram of the delay cell.

The critical path in the delay cell is the propagation delay from the input through the bypass to the output. This is because when a few delay cells (up to 16) are programmed to operate at zero delay mode, without any active FF between them, the propagation delay of all the cells + interconnection path adds up to the set-up time. The best solution we found is the use of three-state inverters. That way the total cell's propagation delay is only about 270 ps (with 0.6 μ CMOS). With efficient routing one can achieve an operating frequency of up to 200 MHz without pipelining within the columns.

The configuration FFs as well as the delay FFs were implemented using the DFS8 cell, which is designed for implementation of JTAG boundary scan logic. The cell gives a low surface combination of a FF and an input multiplexer which is used in our case for mode of operation control (shifting data or configuration). Boundary scan logic is redundant in such an ASIC because one can shift data to any FF in the system using the normal I/Os.

The delay cell is about 30 NAND gate equivalent. Further reduction can be achieved by implementing a standard cell for the delay element in transistor level design. The delay cells schematic diagram is shown in figure 3.3.

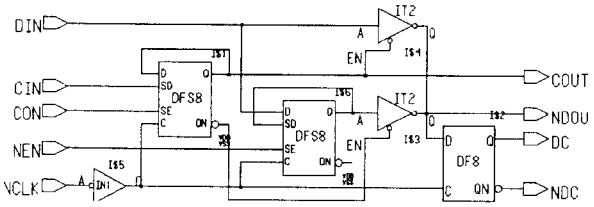


Figure 3.3: Schematic diagram of the delay cell.

3.3 *The Adder Tree*

The implementation of the adders was done in a tree structure (figure 3.4), starting with 8 adders with 4 one bit inputs and a 3-bit binary output. Such an implementation allows us to save some logic and save on trace length as the first adder elements are located close to the associated delay cells. It also allows an easy pipeline structure as pipeline registers are located between the tree levels.

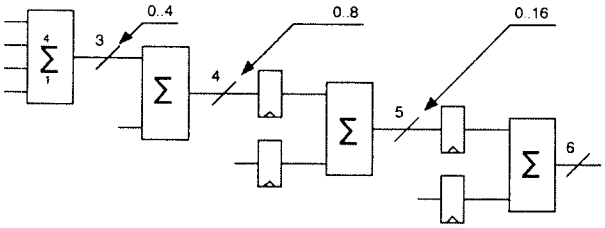


Figure 3.4: The adder's tree structure.

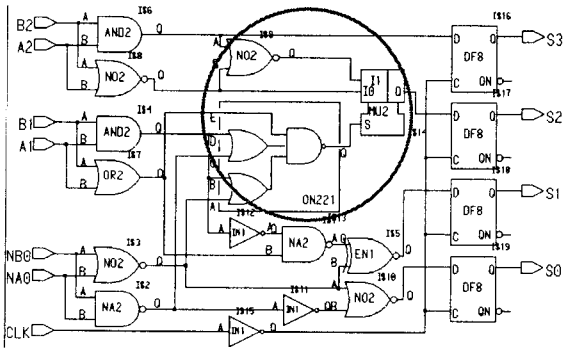


Figure 3.5: Schematic diagram of the adder tree's 2nd stage. In the circle is the simplified logic for the top two sum bits.

More reduction in gate-count was achieved by implementing adder stages, which are optimized for the redundant code used. The possible combinations out of the 4 x 1-bit adder are only 0..4. Therefore the next adder stage does not have to handle all the combinations including 5..7 in one of it's 3-bit inputs. This results in a simplified logic structure of the top two bits of the adder, which are normally the most complex. The same saving was applied to all the cells in the tree.

Figure 3.5 shows the schematic diagram of the 4 x 1-bit adder and the schematic of the second, 2 x 3-bit adder stage. In the circle is the simplified logic for producing the top two bits of the sum. The saving increases with 'wider' adder cells.

3.4 *The Peak Detection*

In order to save some more processing time from the host processor, threshold detection discriminators were added next to the adder units (figure 3.6). The threshold levels can be loaded during chip configuration in parallel to the delay-cell configuration information.

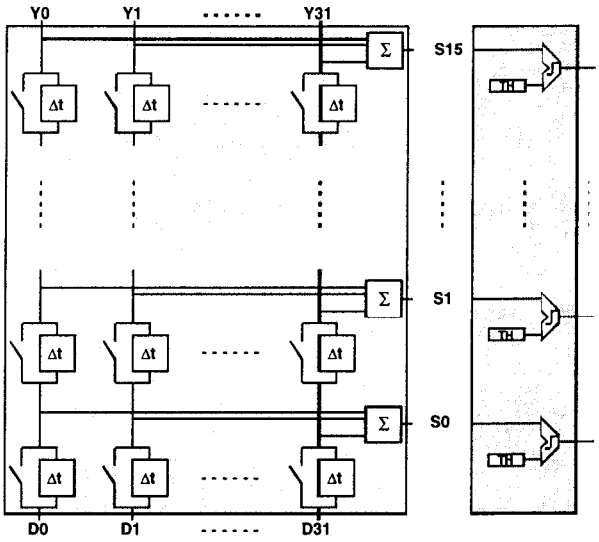


Figure 3.6: Threshold registers and comparators are available on chip next to the last adder stage.

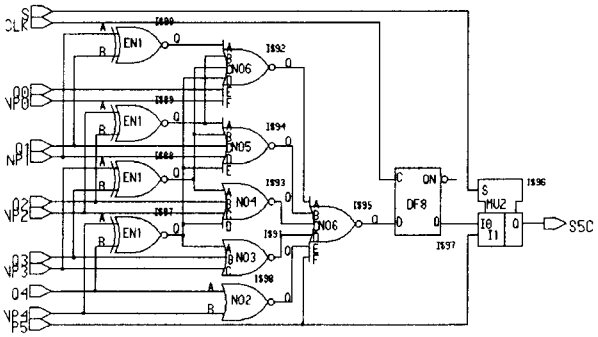


Figure 3.7: The threshold comparator's circuit diagram.

The comparator output is multiplexed with the sum in order to reduce pin-count. The mode of operation, sum or threshold detection output, is determined by an input pin. Similarly to the adder implementation, the discriminator logic was also optimized for the redundant code (figure 3.7).

4. Cascading of Huogh-Transform ASICs

The outputs of the top delay-cell row are available on pins, to allow easy cascading. Producing more columns is achieved simply by connecting these outputs to the input pins of the next stage. A pipeline register at the output allows such cascading without reduction of the operation frequency. One has to note though that the outputs of the second stage are one cycle delayed in comparison to the first stage. This has to be taken into account when producing the 3-D histogram. The result of such cascading is the doubling of the histogram width.

Extending the column width is however more complex. For each row, one has to add the two 6-bit sums. Using the internal threshold detection is not usable for such a configuration.

5. Performance

The technical data of the ASIC is summarised in table 5.1.

Table 5.1: The Hough-Transform ASIC data

Process	AMS 0.6µ 2M CMOS
Dimensions	6.8 x 7.4 mm
Clock Frequency	100 MHz
Array size	32 Columns
	16 Rows
Pipeline stages	5
Package	CPGA256
Prototype	Feb. 2000

The ASIC was implemented with AMS CUB technology (0.6µ CMOS with two metal layers). It is about 50 mm^2 in area and is pad limited as can be seen in figure 5.1.

It integrates 16 rows and 32 columns of delay elements, which allow the calculation of one matrix row (16 points wide) each clock cycle. It can operate at 100 MHz.

The first prototypes are expected in February 2000.

6. Conclusions

The Hough-transform ASIC will allow real time trace detection in systems like the OPAL detector in CERN. It will also enable accelerating by two to three orders of magnitude a wide spectrum of pattern recognition applications, which can be based on the Hough-transform.

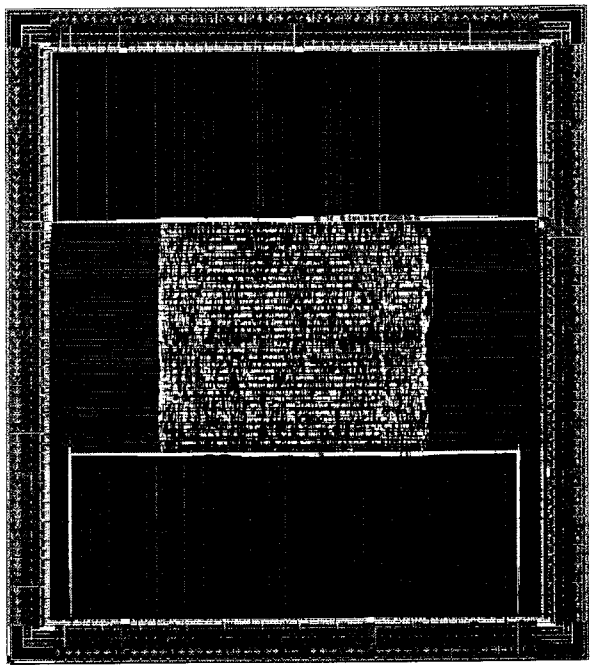


Figure 5.1: The layout of the Hough-Transform ASIC.

7. References

Ballard D. H. (1981). Generalizing the Hough Transform to detect arbitrary shapes, Pattern Recognition, Vol. 13, 111 - 122

Davis E. R. (1992). Modelling peak shapes obtained by Hough transform, IEE proceedings E, Vol. 139, 1

Duda E. O., Hart P. E. (1972). Use of the Hough transform to detect lines and curves in pictures, Comm. ACM (1972) 11 - 15

Durbin R., Mitchison G. (1990) A dimension reduction framework for understanding cortical maps. Nature, 343, 644 – 647

Hough P.V.C. (1959). Machine analysis of bubble chamber pictures, International Conference on High-Energy Accelerators and Instrumentation, CERN

Hough P.V.C. (1962). Method and means for recognizing complex patterns, US Patent 3069654

Ohlsson M., Peterson C., Yuille A. L. (1992). Track finding with deformable templates - the elastic arms approach, Comp. Phys. Comm., Vol. 71, 77 - 98

Van Swaaij M., Catthoor F., De Man H. (1990). Deriving ASIC architectures for the Hough transform, Parallel Computing 16, 113 - 121

CORRESPONDENCE

19. 02. 2000

Standardzell-Bibliotheken der Hersteller anhand von ausgewählten Beispielen

(Auszug aus der Vorlesung Entwurf und Layout Integrierter Schaltungen)

Prof. Dr. Kurt H. Schmidt,
Labor Entwurf Integrierter Schaltungen, Fachhochschule Furtwangen

Vorbemerkung: Im Nachgang zum Workshop am 28.Januar 2000 erfolgte die Bitte, den in 1999 entworfenen und über EUROPRACTICE gefertigten Chip Photozeilensensor-Kernel, aus der Diplomarbeit von Andreas Adler und Patrik Escher, im Studiengang Mikrosystemtechnik an der Fachhochschule Furtwangen, im Workshopband vorzustellen. Hierzu ist zu sagen: Der Chip wurde zum Jahresende 1999 ausgeliefert und es bedarf noch einiger Zeit für einen Testaufbau und das Testen seiner Funktionalität in einem System zusammen mit dem CMOS-Photosensorsystem FUGA. Dies muss noch abgeschlossen werden. Aus dem Blickwinkel der Verwendung von Standardzellen ist der Entwurf des Photozeilensensor-Kernels jedoch bereits in die Vorlesung „Entwurf und Layout Integrierter Schaltungen" als ein Beispiel für den Entwurf mit einer Standardzellen-Bibliothek im WS1999/2000 in Form eines Bildes aufgenommen worden. Der Verfasser hofft, die Thematik des Standardzellen-Entwurfs und einige dazu behandelte Überlegungen mögen für den/die eine/n oder andere/n Leser/in von Interesse sein und womöglich die Diskussion anregen.

Es folgt der Vorlesungsauszug einschließlich der Bilder zweier Beispielentwürfe von gefertigten Chips, wovon der zweite der Photozeilensensor-Kernelchip ist.

6.3 Standardzell-Bibliotheken der Hersteller
anhand von ausgewählten Beispielen

Die Standardzellen haben meist eine gleiche Höhe und weisen am oberen und unteren Rand die Versorgungsschienen für VDD und VSS in Metall auf. Der Umfang der Schaltung bestimmt das andere Maß, die Breite, der Zelle. Die Zellen können somit nebeneinander angordnet werden, weil sie dann Zellenzeilen mit überall gleicher Höhe und durchgezogenen Versorgungsschienen bilden. Für Verbindungen zwischen den Zellen dienen entweder freizulassende Verdrahtungskanäle zwischen den Zellenzeilen, oder wo mehrere Metallebenen zur Verfügung stehen, werden die Verbindungen in den höheren Metallebenen verlegt. Letzteres hat den Vorteil, dass die Zellenzeilen dicht an dicht liegen können, also für die Transistoren der Zellen jeder Teil der inneren Chipfläche nutzbar ist.

Die meist manuell entworfenen Standardzellen werden in Zellbibliotheken zusammengefasst und so dem Benutzer verfügbar gemacht. Bei der Highlevel-Synthese muß von den EDA Tools die synthetisierte Schaltung in den speziellen Bibliothekselementen eines gewünschten Herstellers implementiert, also automatisch darin abgebildet, werden. Zellenausgänge müssen gegebenenfalls über Treiberstufen leistungsmässig angehoben werden, um längere Verbindungsleitungen auf dem Chip und erforderlichenfalls viele Eingänge von Gattern, das ist eine hohe Last, treiben zu können.

Schließlich gibt es für den Padframe eines Chips Ausgangsverstärker und Eingangsschutz-schaltungen als sogenannte Padzellen.

EDA-Tools für das Placement und Routing der Zellen stehen zur Verfügung, so dass das Layout des Chips im wesentlichen automatisch aus den Standardzellen erstellt werden kann. Eingriffe sind dennoch immer wieder nötig, insbesondere bei Clocksignalverteilungen, bei Treiberverstärkern, und natürlich bei der elektrischen Anpassung des Peripherieinterfaces der Padschaltungen des Chips.

6.3.1 Beispiel: Zellen aus ES2 Solo1400

Sie standen anfangs der 90er Jahre zur Verfügung und sind in einer 1,5µm Technologie double-metal silicon-gate CMOS ausgeführt (ES2 European Silicon Structures, es handelte sich um eine gemeinsame Chipfoundry ca 10 europäischer Firmen. Der Zugang der Hochschulen erfolgte über das Programm EUROCHIP.) Für die Darstellung hier sind vier Zellen aus der nachstehenden Gesamtliste ausgewählt. Und es ist ein im FHF-Studiengang Mikroelektronik mit dieser Zellbibliothek entworfener Chip „Interruptfrontend" im Umfang von über 10000 Transistoren dargestellt. Dort sieht man sehr gut die zeilenweise Anordnung, wobei das EDA-Tool aus Gründen der Verdrahtung von sich aus drei Zeilen-„Spalten" bildete. Die Plots stellen diese Layer dar

CTOX	active diffusion,	CNWI	n-well,
CPOL	polysilicon,	CNPI	n+ diffusion,
CPPI	p+ diffusion,	CCON	contacts,
CME1	metal 1,	CVIA	via,
CME2	metal 2,		

sie sind in schwarz-weißer Darstellung des Skriptes kaum unterscheidbar.

Standardzellen-Bibliothek

Kombinatorische Schaltelemente und Buffer.

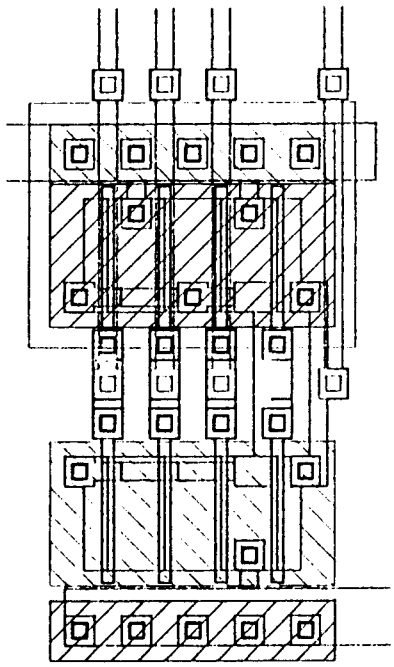
Funktion	Name
Layer	
Inverter	(not)
2 Input Nand	(nand2)
3 Input Nand	(nand3)
4 Input Nand	(nand4)
2 Input Nor	(nor2)
3 Input Nor	(nor3)
2 Input And	(and2)
3 Input And	(and3)
2 Input Or	(or2)
3 Input Or	(or3)
Exor	(xor)
Exnor	(eqv)
Full Adder	(sumcar)
Inverting Buffer * 2	(buffer2)
Inverting Buffer * 3	(buffer3)
Inverting Buffer * 4	(buffer4)
Tristate Buffer * 2	(tribuf2)
Tristate Buffer * 3	(tribuf3)
Tristate Buffer * 4	(tribuf4)
Tristate Inverter	(trinot)

Sequentielle Schaltelemente , Flip-Flops und Latches.

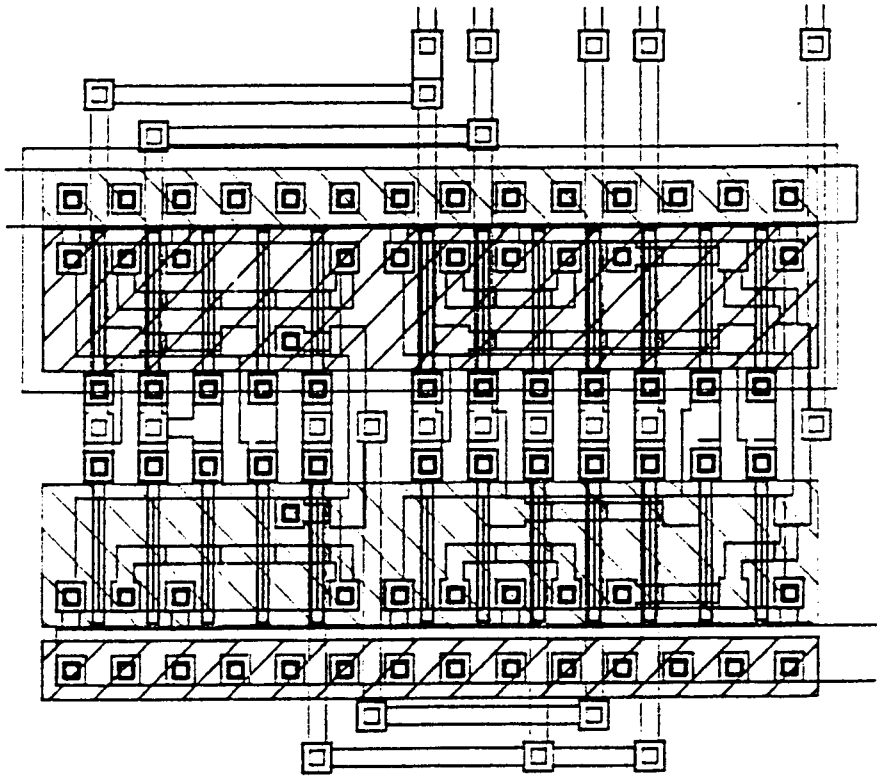
Funktion	Name
Layer	
Basic D-type FF	(bdff)
D-type FF with Set & Clear	(bdffn)
Transmission Gate D-type FF	(bdffct)
Scan-Path D-type FF	(bsdff)
Basic D-type Latch	(latch)
D-type Latch with Set & Clear	(latchn)

Padzellen

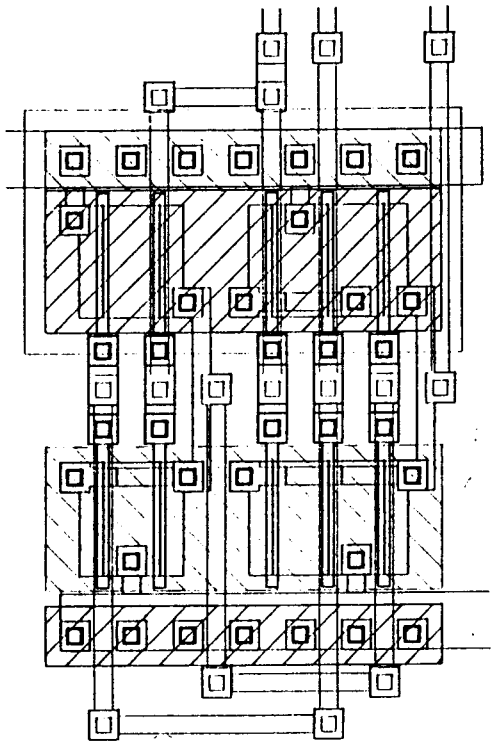
CMOS Input Pad	(ips8b)
TTL Input Pad	(ips8g)
Output Pad	(ops1u)
Tristate Output Pad	(ops1w)
Open Drain Output Pad	(ops1z)
Bidirectional Pad with CMOS Input	(ios1k)
Bidirectional Pad with TTL Input	(ios1p)
Power Pad	(vccc)
Ground Pad	(vsss)
Power Pad with Power-on Reset and Reference Pad for testing	(vccspor)
On-chip Chrystal Oscillator	(osc1)



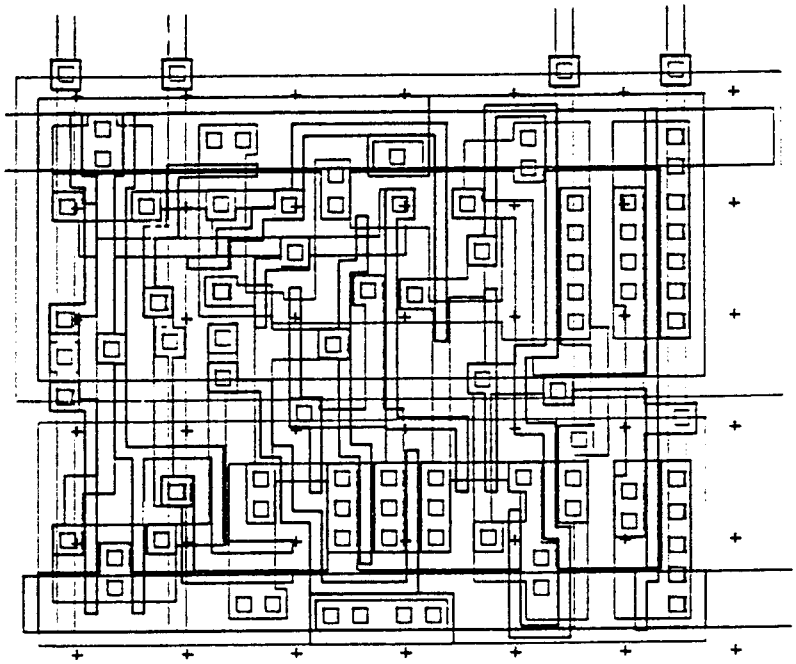
and3



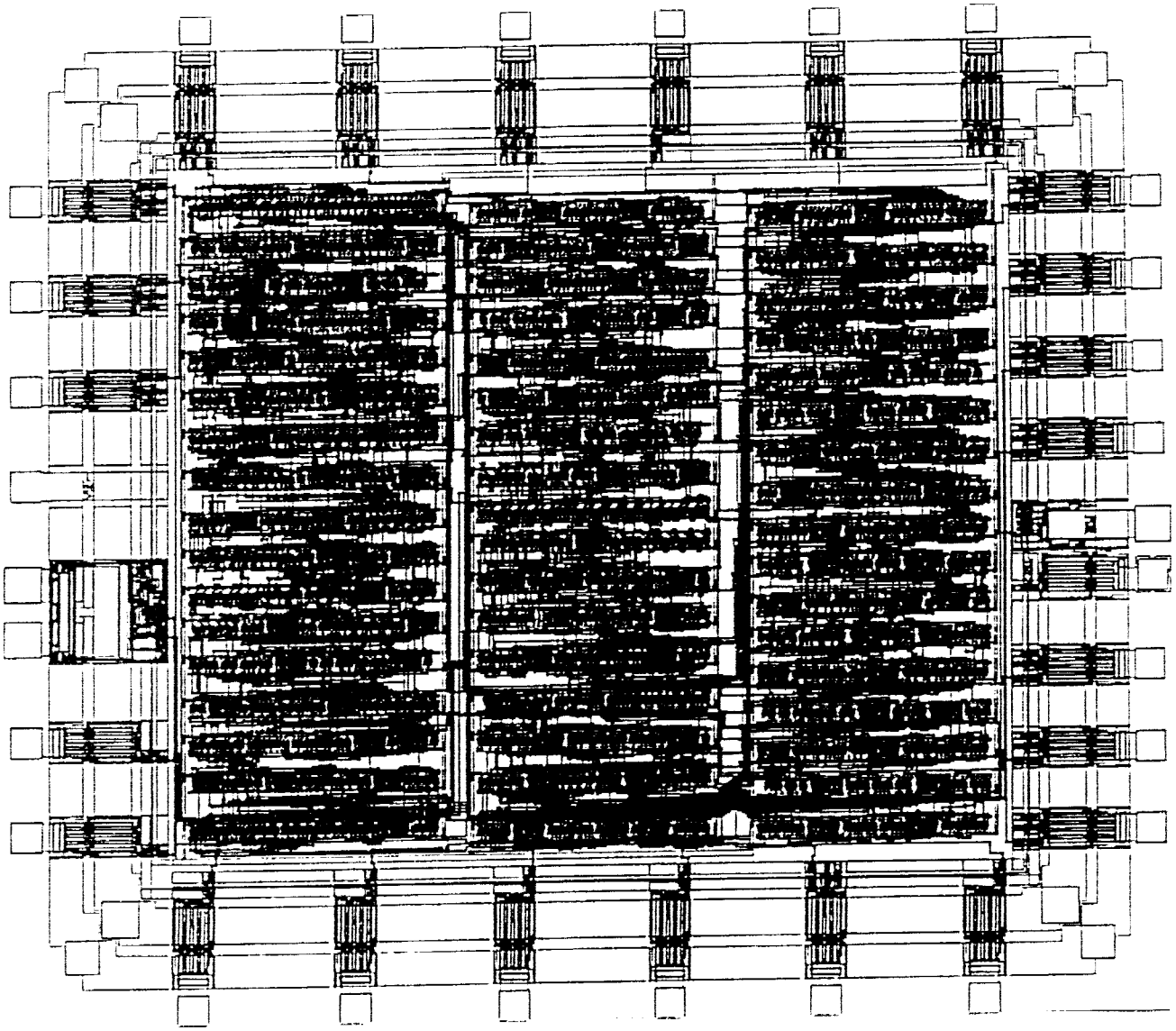
sumcar



xor



bdff



Interruptfrontend Chip
(DA B.Wichert, FHF 1991)

6.3.2 Beispiel: Zellen in Sub-µm-Technologie von MIETEC

Die Zellen sind in 0,5µm CMOS-Technologie für 3,3V entworfen worden und stehen seit 1997 zur Verfügung (Libraries MTC35000, 35100). Der Zugang der Hochschulen erfolgt über das von Brüssel geförderte europäische Programm EUROPRACTICE. Der Umfang der Zellenbibliothek ist im Vergleich zu Bibliotheken aus früheren Jahren enorm gestiegen. Allerdings kann man die zugehörigen Zellenlayouts nicht anschauen, sie sind geschützt, aus der Sicht des Herstellers eine notwendige Maßnahme.

Auch gelten die Unterlagen zu den aktuellen Bibliotheken als vertraulich. Man erfährt die Funktionalität der verfügbaren Zellen, ihren Footprint, einige Daten und die zugehörigen Zellennamen. Daher wird nachstehend lediglich eine etwas gekürzte Auswahlliste zur Veranschaulichung aufgeführt.

Des weiteren ist ein im FHF-Studiengang Mikrosystemtechnik mit dieser Zellbibliothek entworfener Chip „Photozeilensensor-Kernel" dargestellt. Er enthält außer den Logikteilen noch zwei RAM-Blöcke zur Datenspeicherung der digital gewandelten bytegroßen Pixelwerte. Die Anzahl der Transistoren des Chips liegt bei gut über 20000. Dennoch fällt auf, dass es noch viel freien Platz auf dem Chip gibt. Die Fläche wird vom Padframe bestimmt, denn den kann man bei einer gewünschten Pinanzahl nicht kleiner machen, weil man die Anschlüsse auf die Bondpads bonden muß. Da ist ein Abstand zwingend vorgegeben. Im Gegensatz dazu war noch bei der 1,2 µm Technologie die Chipfläche eher vom Chipcore bestimmt. In der 0,5 µm Technologie kann man aber sehr viel mehr Schaltungsumfang unterbringen.

Seit 1980 bei 6 µm konnte man das Gesetz einer Verdoppelung alle 2 Jahre des Gatterumfangs auf gleicher Fläche beobachten. Das heißt Vervierfachung alle 4 Jahre. Was die Kanallänge der Technologie anbetrifft, bedeutete das eine Halbierung etwa alle 4 Jahre.

Die Entwicklung für „random logic" verlief in etwa gemäß dieser Aufstellung (Speicherchips ausdrücklich ausgenommen):

Jahr	Kanallänge	Mögl. Schaltungsumfang bei gleicher Chipfläche	Spannung
1980	6µm	Bezugsgröße	5V
1984	3µm	4-fach	5V
1988	1,5µm	16-fach	5V
1992	0,75µm	64-fach	5V
1996	0,35µm	256-fach	3,3V
≥2000	0,18µm	1024-fach	≤2,5V

Stand ist 1999, dass die 0,25µm Technologie kommerziell geworden ist. Der weitere Fortschritt befindet sich in der Entwicklung. - Hatte man 1980 z.B. 3000 Transistoren auf einem Chip mittlerer Größe, können da im Jahr 2000 3 Millionen Transistoren auf die gleiche Fläche passen. Die Frage ist, wo liegt für Silizium die Grenze der Nanoelektronik (statt Mikroelektronik muß man im sub-µm Bereich eigentlich bereits Nanoelektronik sagen!). Vielleicht schon bei 30nm Kanallänge, wie es von einigen Fachleuten angenommen wird. Der Preis der Technologielinien, also der Foundries, steigt aber unverhältnismäßig stark an, bei der Umstellung von 0,25µm auf 0,18µm ist es beispielsweise um mehr als das 5-fache. Es werden daher wahrscheinlich mehr als zehn Jahre erforderlich, um auf eine 30nm

Technologie zu kommen, wenn man sie überhaupt noch finanzieren kann. Und ob es dann vom Markt her Anwendungen mit Stückzahlen im 100 Millionbereich/Chip gibt (Speicher ausgenommen), damit es sich rechnet, ist noch offen. Der Chip könnte dann ja ca 100 Millionen Transistoren auf der gleichen Fläche unterbringen, und sich nur für „Hard-Algorithmen", „Intelligenz", oder wie immer man das nennen will, mit mindestens diesem Schaltungsumfang lohnen. Bei dieser Überlegung fällt auf, dass man vielleicht nicht immer die vorangehenden Technologien völlig ersetzt, sondern – spekulativ gesagt - diese wegen ihrer besseren Preiswürdigkeit bei kleinerem Umfang parallel beibehält. Auch weiß man noch nichts genaues über die Verringerung der Zuverlässigkeit, wenn man noch weiter verkleinert.

Nun also die angekündigte gekürzte Zellenauswahlliste, s. die nächste Seite:
Besonders umfangreich sind die Flipflop Funktionen und eine Gruppe von komplexen Gattern, die sich AND-NOR- oder NAND-OR-Funktionen nennen.

Von einem Symbol (Zellnamen) gibt es grundsätzlich weitere Angaben
am Beispiel **AO11A**:

Area 440 μm^2, Power 0.925 $\mu W/MHz$,

sowie die Schaltzeiten tabellarisch bei unterschiedlichen Standardlasten (SL),

a to z tPLH 0.39ns, 1SL 0.55ns, 2SL 0.70ns usw
tPHL 0.42ns, 1SL 0.54ns, 2SL 0.67ns usw
b to z .....
.....
.....

sowie die Eingangskapazitäten normiert auf die Standardlast (SL) und output drive (fanout)

a 0.9SL, b 0.7SL, usw ..... ; Out z 4 SL

Solche Angaben kann der Benutzer präzise in der zugehörigen Dokumentation einer „cell-library" vorfinden.

AND Functions
AN2 2 inputs
AN2L low drive
AN2X4 4xdrive
AN3
AN4
AN5
.....
NAND Functions
ND2 2 inputs
ND2L low drive
ND2X4 4xdrive
ND3
ND4
ND5
ND8
.....
NOR Functions
NR2 2 inputs
NR2L low drive
NR2X4 4xdrive
NR3
NR4
NR5
NR8
.....
OR Functions
OR2 2 inputs
OR2L low drive
OR2X4 4xdrive
OR3
OR4
OR5
.....
Adder Functions
FA1A 1bit full adder

Multiplexer Functions
MUX21 2 to 1
MUX21X4 4xdrive
MUX41 4 to 1
MUX41X4 4xdrive
.....
D-FF Functions
FD1 pos.edge trig.clk
FD1P 2xdrive
FD1S with scan inputs
FD1SX4 4xdrive
FD1X4 4xdrive
FD2 with asyn.reset
FD3 asyn.reset and set

JK-FF Functions
FJK1 JK-FF
FJK1S with scan inputs
FJK1SX4 4xdrive
FJK1X4 4xdrive
FJK2 with asyn.reset
FJK2S with scan inputs
FJK2SX4 4xdrive
FJK2X4 4xdrive
FJK3 asyn.reset and set
FJK3S with scan inputs
FJK3SX4 4xdrive
FJK3X4 4xdrive
.....
Latch Functions
LD1 D-Latch
with active high clk
LD1P 2xdrive
LD1S with scan inputs
LD1SX4 4xdrive
LD1X4 4xdrive
LD2 active low clk
.....
LD3 act.high clk, reset
.....
LD4 act.low clk, reset
.....

Inverter Functions
IV Inverter
IV4 4xdrive
IV8 8xdrive
IV16 16xdrive
IVL low drive
IVP 2xdrive

Non-Inverting Buffer Functions
BF1T1 Buffer
BF1T2 2xdrive
BF1T4 4xdrive
BF2T8 8xdrive
BF2T16 16xdrive

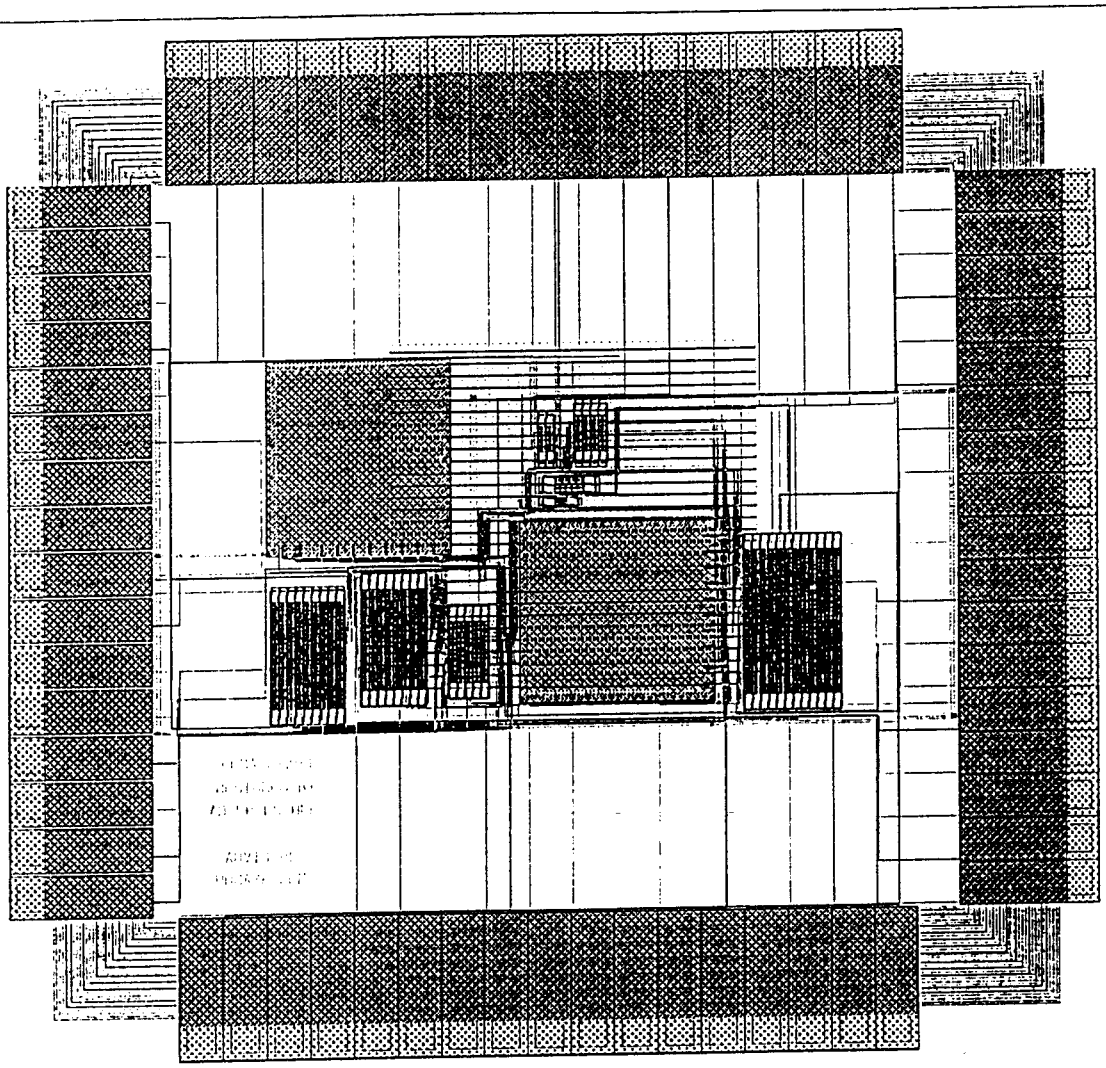
Internal Tristate Functions
BTSP pos.enable,2xdr.
BTSX4 4xdrive
BTSX8 8xdrive
BTSX16 16xdrive
BTSX32 32xdrive

AND-NOR or NAND-OR Functions
AO1
z=!((d+c+b)(d+c+a))
.....
AO11
z=!((ba)+(dc)+(fe))

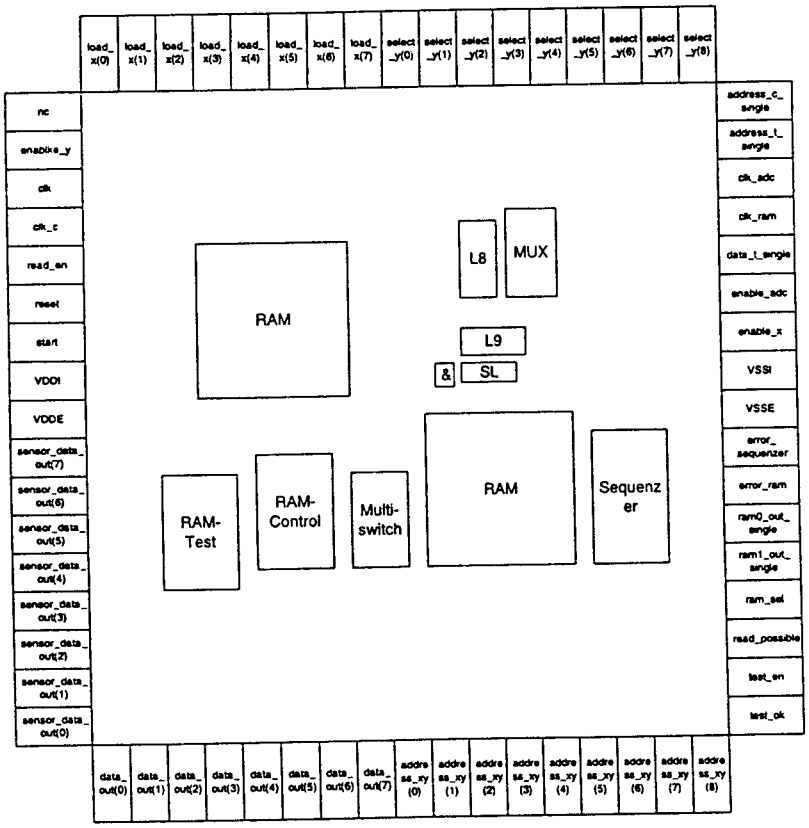
AO11N z=(ba)+(dc)+(fe)

AO11A
z=!((b&!a)+(dc)+(fe))
.....
AO12
z=!((h+g)(d+c)(b+a)(f+e))
.....
AO13
z=!((d+c+e)(b+a+e))
.....
AO14 z=!((cba)+(fed))
.....
AO15
z=!((f+e+d)(c+b+a))
.....
AO2 z=!((ba)+(dc))
.....
AO21 z=!(d (b+a+c))
.....
AO6 z=!((ab)+c)
.....
AO7 z=!((cb)+(ca))
.....
AO9 z=!((acb)+(de))
.....
EXOR-EXNOR Fctns.
EN 2input EXNOR
(with EXOR output)
ENX4 4xdrive
EN3 3input EXNOR
EN3X4 4xdrive
.....
Pad Cells
ca 150 Zellen !
mit TTL-, CMOS-
Interface, Tristate,
Schmitt-Trigger etc

Auszug aus einer Zellbibliothek zur Veranschaulichung

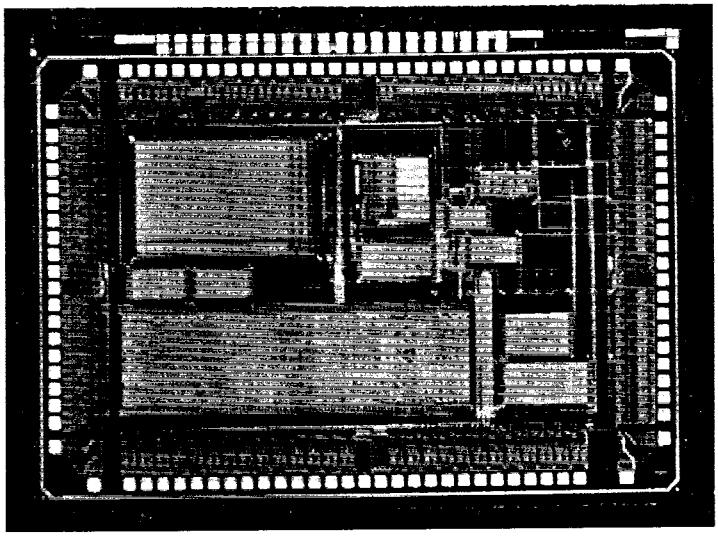


Photozeilensensor-Kernel Chip
(DA A.Adler, P.Escher, FHF 1999)



Layout-Floorplan

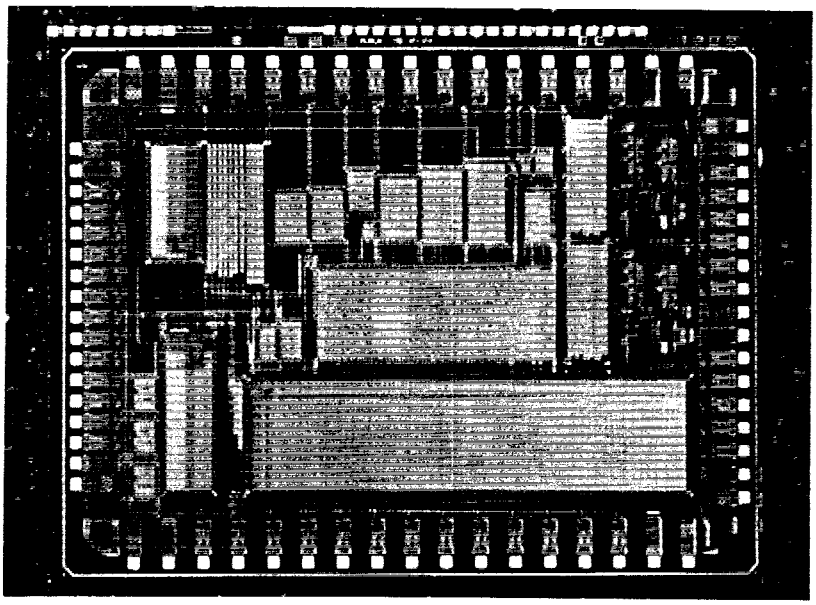
FHOP_MIKROCONTROLLER_V2



Entwurf:	Fachhochschule Offenburg Bearbeiter: Markus Fischer Betreuer: Prof. Dr.-Ing. Dirk Jansen
Layouterstellung:	Fachhochschule Offenburg (Standardzellenentwurf)
Technologie:	Alcatel Mietec 0.5µm CMOS_AD
Chipfertigung:	Europractice, Run 817
Herstelldatum:	Juni 1999
Kostenträger:	MPC-Mittel FH-Verbund Baden-Württemberg
Chipdaten:	Chipgröße: 3,8 x 5,6 mm^2 Gehäuse: CQFP120 Komplexität: ca. 60000 Transistoren
Funktion:	Das IC enthält einen kompletten Mikrocontroller, basierend auf dem, an der Fachhochschule Offenburg, weiterentwickelten FHOP_2 Mikroprozessor-Kernel. Als Peripheriemodule sind ein Buscontroller, eine Chipselect-Waitstate-Einheit, ein 2 KB-ROM, ein 2 KB-RAM, ein Watchdog, eine 16 Bit breite PIO, eine serielle Schnittstelle, zwei Timer, eine I^2C-Schnittstelle und ein Interruptcontroller vorhanden. Im ROM ist bereits ein BIOS implementiert, das Funktionen für die einzelnen Komponenten bereitstellt. Das Anwender-Programm wird über die I^2C-Schnittstelle aus einem EEPROM eingelesen.
Testergebnisse:	Der Mikrocontroller wurde erfolgreich getestet. Die Funktionen sämtlicher Peripheriemodule sind voll erfüllt. Die einzelnen Module können nun für andere Anwendungen verwendet werden.

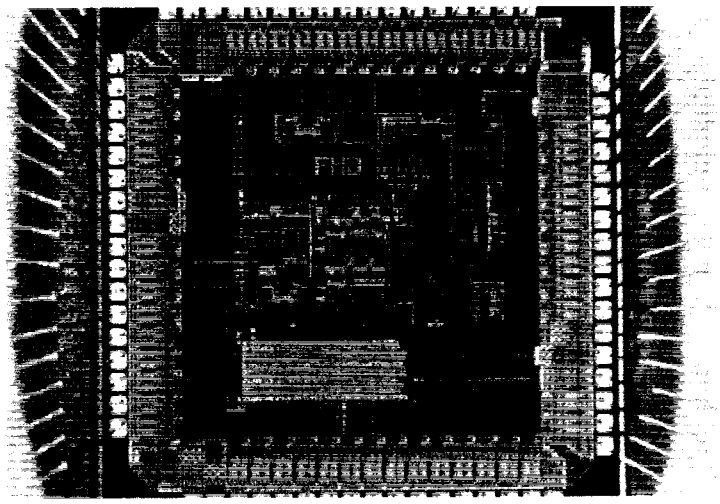
Max. Takt:	50 MHz
Leistungsaufnahme:	2.4 mW/MHz
MIPS:	12.5 (bei 50 MHz)

Dual SignalWavelet Processing Controller (DSWPC)



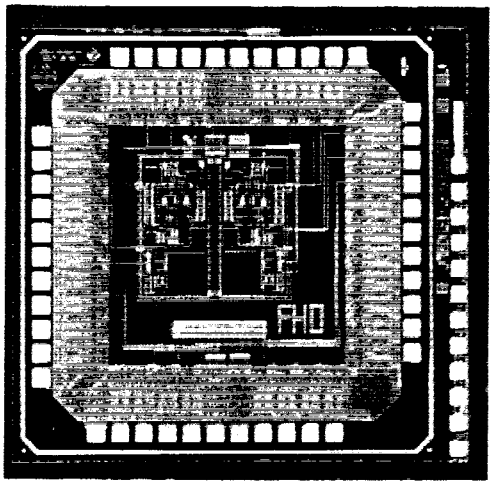
Entwurf:	Fachhochschule Offenburg Bearbeiter: Wolfgang Vollmer / Carsten Störk Betreuer: Prof. Dr.-Ing. Dirk Jansen
Layouterstellung:	Fachhochschule Offenburg (Standardzellenentwurf)
Technologie:	Alcatel Mietec 0.5µm CMOS_AD
Chipfertigung:	Europractice, Run 817
Herstelldatum:	Juni 1999
Kostenträger:	MPC-Mittel FH-Verbund Baden-Württemberg
Chipdaten:	Chipgröße: 6,6 x 4,9 mm^2 Gehäuse: JLCC68 Komplexität: ca. 80000 Transistoren
Funktion:	Bei diesem Chip handelt es sich um einen Mikrocontroller auf der Basis des FHOP Mikroprozessor-Kernel mit integrierter DSP-Einheit und zwei Sigma-Delta-Wandlern. Die DSP-Einheit arbeitet im Parallelbetrieb zum Prozessor und führt neben einer IIR- und FIR-Filterung der Eingangsdaten der ΣΔ-Wandler auch eine Waveletkomprimierung durch. Zur Kommunikation mit der Außenwelt sind neben drei 8 Bit breiten parallelen Schnittstellen auch eine serielle Schnittstelle, eine I^2C Schnittstelle und eine induktive Schnittstelle (PSK-Modem) integriert. Außerdem besitzt der Controller 4 KB ROM, 6 KB RAM, zwei 16 Bit Timer und einen Watchdog.
Testergebnisse:	Der Mikrocontroller wurde erfolgreich getestet. Die Funktionen sämtlicher Peripheriemodule sind voll erfüllt. Die einzelnen Peripheriemodule, die für diesen Chip auf die neue Zieltechnologie umgesetzt werden mußten, können nun auch für andere Anwendungen eingesetzt werden.

Sigma-Delta-Wandler 2.Ordnung



Entwurf:	Fachhochschule Offenburg Bearbeiter: Carsten Störk Betreuer: Prof. Dr.-Ing. Dirk Jansen
Layouterstellung:	Fachhochschule Offenburg (Mixed-Signal-Entwurf)
Technologie:	Alcatel Mietec 0.5µm CMOS-AD
Chipfertigung:	Europractice, Run 289
Herstelldatum:	Dezember 1998
Kostenträger:	MPC-Mittel FH-Verbund Baden-Württemberg
Chipdaten:	Chipgröße: 3,4 x 3,4 mm^2 Gehäuse: JLCC 68 Komplexität: ca. 5000 Transistoren
Funktion:	Bei diesem Chip handelt es sich um einen Analog/Digital-Wandler, der für das Projekt MINELOG entworfen wurde. Er beinhaltet einen Sigma-Delta-Modulator 2.Ordnung, welcher ein 1 Bit Digitalsignal liefert und einen SINC3-Filter, der hieraus ein 20 Bit Ausgangssignal erzeugt. Desweiteren wurden einige der für den ΣΔ-Wandler entworfenen Komponenten mit auf diesen Testchip integriert, unter anderem eine Band-Gap-Spannungsquelle, eine Referenzspannungserzeugung, ein Komparator, ein Differenzverstärker und ein Instrumentenverstärker.
Testergebnisse:	Eine vollständige Auswertung des Chips wird zur Zeit noch durchgeführt. Erste Messungen zeigten aber, daß der Sigma-Delta-Wandler bis zu einer Taktrate von f = 500 kHz betrieben werden kann. Hierbei beträgt die maximale Signalauflösung 16 Bit, bei einem Eingangsspannungsbereich von ± 1 V.

Temperaturzelle_v2

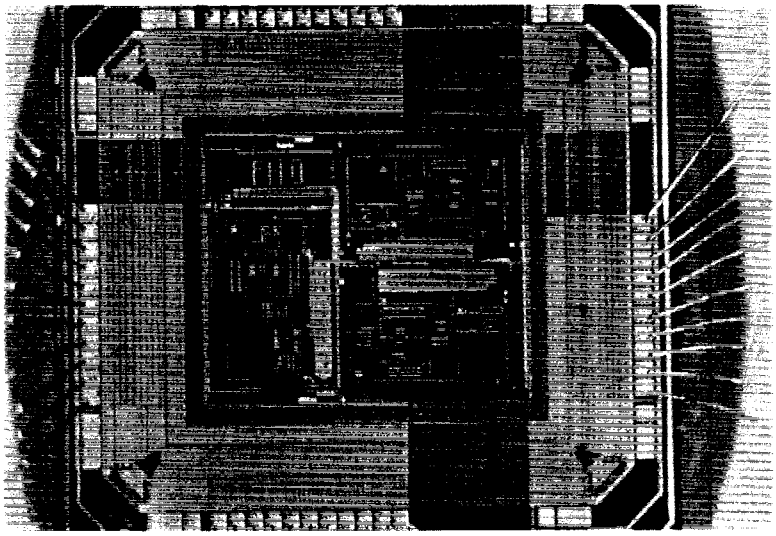


Entwurf:	Fachhochschule Offenburg Bearbeiter: Jürgen Hauser Betreuer: Prof. Dr.-Ing. Dirk Jansen
Layouterstellung:	Fachhochschule Offenburg (Standardzellenentwurf)
Technologie:	Alcatel Mietec 0.5µm CMOS_AD
Chipfertigung:	Europractice, Run 817
Herstelldatum:	Juni 1999
Kostenträger:	MPC-Mittel FH-Verbund Baden-Württemberg
Chipdaten:	Chipgröße: 2,5 x 2,5 mm^2 Gehäuse: JLCC44 Komplexität: ca. 2000 Transistoren

Funktion: In diesem Chip sind 2 Temperatursensoren sowie eine Auswerteelektronik enthalten. Der analoge Teil enthält Bandgapschaltung, die jedoch nicht als Spannungsreferenz arbeitet, sowie einen Sigma-Delta-Wandler. Die Bandgapschaltung liefert als Ausgangssignale einen Referenzstrom sowie einen temperaturabhängigen Strom. Der ΣΔ–Wandler arbeitet mit Strömen und liefert einen Temperatur-abhängigen Takt. Die Auswerte-elektronik zählt die Anzahl der Takte innerhalb einer bestimmten Zeit und liefert daraus einen Digitalen Wert. Analogteile und Digitalteil sind getrennt. Aus einem Analogteil sind Analoge Testsignale herausgeführt.

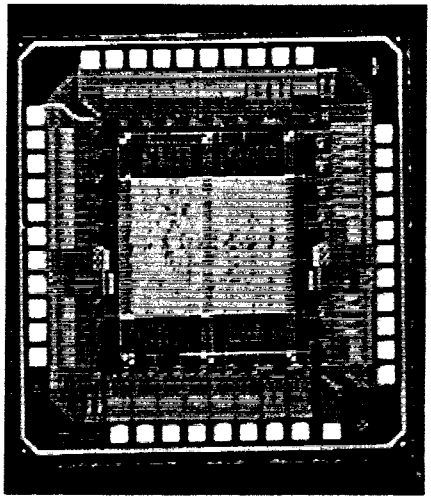
Testergebnisse: Bei diesem Entwurf wurden Analogteil und Digitalteil getrennt um Messungen durchführen zu können. Weiterhin wurden Padzellen einer anderen Bibliothek wie bei Version 1 verwendet. Sonst wurden keine Veränderungen vorgenommen.
Dieser Chip zeigt die gewünschte Funktion. Die genaue Vermessung des Chips hat keinen Einfluß der Spannung und der Taktfrequenz auf das Ergebnis gezeigt. Die Kennlinie ist über den gesammten Temperaturbereich nahezu linear. Mit einer Zweipunkteichung kann eine Genauigkeit von besser ±0,5°C erreicht werden.

Temperatursensor_V1



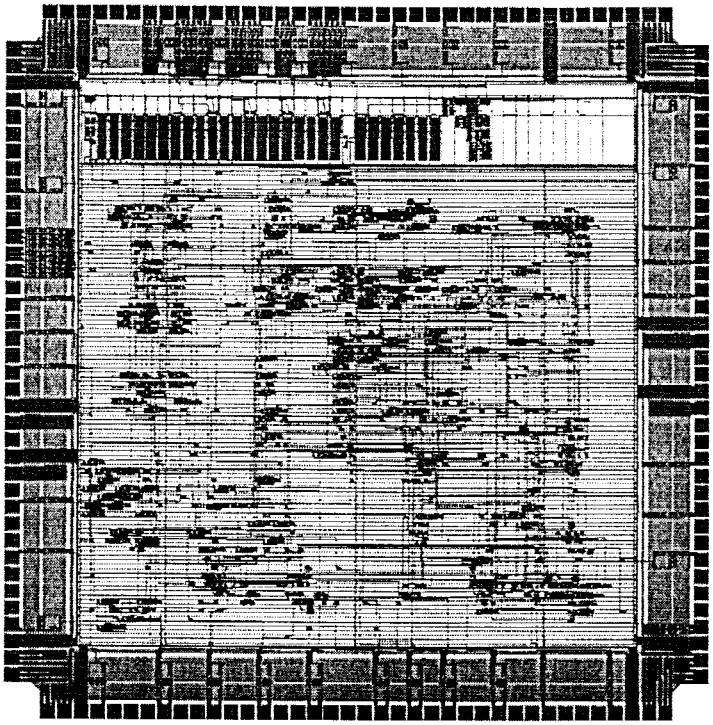
Entwurf:	Fachhochschule Offenburg Bearbeiter: Jürgen Hauser Betreuer: Prof. Dr. - Ing. Dirk Jansen
Layouterstellung:	Fachhochschule Offenburg (Mixed-Signal-Entwurf)
Technologie:	Alcatel Mietec 0.5µm CMOS-AD
Chipfertigung:	Europractice, Run 272
Herstelldatum:	Oktober 1998
Kostenträger:	MPC-Mittel FH-Verbund Baden_Württemberg
Chipdaten:	Chipgröße: 3,2 x 2,9 mm² Chipgehäuse: JLCC 44 Komplexität: ca 3000 Transistoren
Funktion:	In diesem Chip sind 3 Temperatursensoren mit digitaler Auswerte-elektronik enthalten. Der analoge Teil enthält Bandgapschaltung, die jedoch nicht als Spannungsreferenz arbeitet, sowie einen Sigma-Delta-Wandler. Die Bandgapschaltung liefert als Ausgangssignale einen Referenzstrom sowie einen temperaturabhängigen Strom. Der ΣΔ–Wandler arbeitet mit Strömen und liefert einen Temperatur-abhängigen Takt. Die Auswerte-elektronik zählt die Anzahl der Takte innerhalb einer bestimmten Zeit und liefert daraus einen Digitalen Wert. Die digitalen Auswerteelektroniken sind unterschiedlich geroutet und haben Unterschiedliche Busanschlüsse (8Bit und 16 Bit).
Testergebnisse:	Der Chip zeigte keine Funktion. Es wurde Festgestellt, dass Versorgungen für den Padbereich nicht gebondet waren. An einem Chip wurde dies nachgeholt. Trotzdem konnte keine Funktion festgestellt werden. Auf eine weitergehende Untersuchung wurde verzichtet.

Lotto_V3



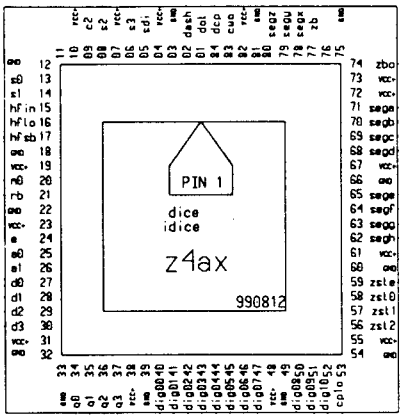
Entwurf:	Fachhochschule Offenburg Bearbeiter: Carsten Neumeister / Wofgang Vollmer Betreuer: Prof. Dr.-Ing. Dirk Jansen
Layouterstellung:	Fachhochschule Offenburg (Standardzellenentwurf)
Technologie:	Alcatel Mietec 0.5µm CMOS-AD
Chipfertigung:	Europractice, Run 817
Herstelldatum:	Juni 1999
Kostenträger:	MPC-Mittel FH-Verbund Baden-Württemberg
Chipdaten:	Chipgröße: 2,5 x 2,3 mm^2 Gehäuse: JLCC 44 Komplexität: ca. 10000 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 wurde mit Renoir in VHDL umgesetzt. Dabei wurden Fehler der früheren Versionen verbessert. Durch die Umsetzung ist eine weitgehende Technologie-unabhängikeit erreicht worden. Die Funktionalität der einzelnen Komponenten ist gegeben

BIDUEC2 – Übertragungschip 2 zur bidirektionalen Datenübertragung



Entwurf:	Fachhochschule Esslingen, Standort Göppingen Bearbeiter: Dipl.-Ing. B. Harrer, Betreuer: Prof. Dr. H. Töpfer
Layouterstellung:	IMS - Institut für Mikroelektronik Stuttgart
Technologie:	0,8µm CMOS Gate Forest (Mixed-Signal Gate Array)
Chipfertigung:	IMS - Institut für Mikroelektronik Stuttgart
Herstellungsdatum:	Oktober 1999
Kostenträger:	MPC-Mittel, FH-Verbund Baden-Württemberg
Chipdaten:	Master: GFN 012 mit 14 Analogfeldern Chipgröße: 4,8 x 4,8 mm^2 Gehäuse: CLCC44 Komplexität: ca. 9800 Transistoren (Digitalteil) ca. 70% der verfügbaren Widerstände
Funktion:	Analogteil: Verstärken, Abtasten und A/D-Umsetzen (8Bit-SAR) eines Messsignals. Das Messsignal liegt im Bereich −0,25V...+0.25V vor und wird durch einen Differenzverstärker auf 0,15V...4,8V verstärkt. Digitalteil: Der Digitalteil enthält Einheiten für ein bidirektionales Senden und Empfangen von Impulsfolgen, zum Erzeugen von Zündimpulsen für eine IGBT-Halbbrücke und zur Generierung der Steuersignale für den A/D-Umsetzer. Weiterhin enthält das Chip eine Synchronisiereinheit und die Paritybit-Generierung. Eine detailierte Funktionsbeschreibung ist dem MPC-Workshopband Juli 1999 zu entnehmen.
Testergebnisse:	Der Digitalteil arbeitet korrekt. Der Differenzverstärker funktioniert nicht, auch die A/D-Umsetzung arbeitet fehlerhaft. Ein korrigiertes ASIC wurde vom IMS zugesagt.

Frequenzzähler z4a



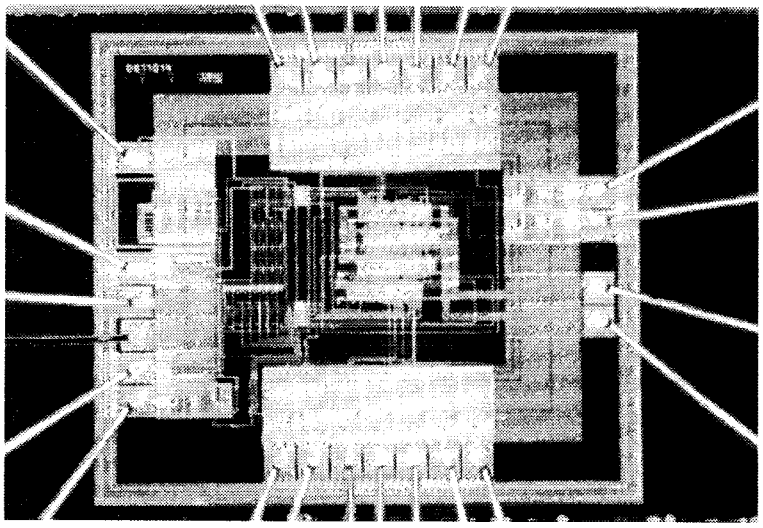
Der Schaltkreis z4a wurde im Rahmen des CAE-Kurses WS98/99 an der FH Ravensburg-Weingarten in Zusammenarbeit mit dem Institut für Mikroelektronik Stuttgart (IMS) entwickelt. Ziel: Frequenzzähler möglichst hoher Bandbreite und Empfindlichkeit, Uhrenfunktion und eine elektronische Morsetaste.

Die **Bandbreite von ca. 800MHz** liefert ein dynamischer CMOS-Vorteiler. Sein Eingangssignal wird durch vier geteilt und dekadisch synchronen Zählerstufen zugeführt, drei Eingangspins erwarten digitale, CMOS-kompatible Signale bis ca. 230MHz, weitere **empfindliche Eingangspins** treiben einen CMOS-Komparator. Die analogen Baugruppen und der dynamische Vorteiler stammen aus der Schaltungsbibliothek des IMS. Wichtige Merkmale in Kürze:

- Zeitnormal wählbar: als externe Quarzreferenz können die Frequenzen 1,2,4,8,16,32,64MHz und deren Fünffaches eingestellt werden.
- Anpassung Vorteiler: hochfrequente Signale bis ca. 800MHz werden mit einem dynamischen Vorteiler durch vier geteilt. Um eine „richtige" Anzeige zu erhalten ist die Torzeit bei der Frequenzmessung entsprechend zu verlängern. Sieben Teilerstufen sind vorhanden.
- Meßdaten werden an eine 7-Segment-Anzeige mit zehn Stellen und einer Statusstelle ausgegeben. Im PC-Mode kann man bei Bedarf alle Segment-Pins in transparente „ports" verwandeln. Damit lassen sich verbreitete LCD-Anzeigen ansteuern, ohne am Prozessor zusätzliche Pins zu belegen.
- Die Funktionskontrolle erfolgt durch eine Prozessor-Schnittstelle oder mit interner Stell-Logik. Drei Schalter stellen die Uhr, weitere 21 verwalten den Baustein, solange mit Pin $m0 = 0$ der Mini-Mode aktiv ist. Ist $m0 = 1$, so wird die Stell-Logik abgetrennt - Funktionen sind nun durch interne Registerbänke der PC-Schnittstelle definiert. Schalterstellungen können jederzeit gelesen und ausgewertet werden.

Komplexität:	ca. 35000 Transistoren
Technologie:	$0.8\mu m$ CMOS-gate-array
Chipfertigung	IMS Stuttgart, im Rahmen des MPC-Verbundes Baden-Württemberg
CAE-Software	Mentor V8, VHDL, Design-Architekt, Quicksim, ...
Entwurf:	CAE-Projekt 5. Semester Elektronik, WS98/99
Entwicklungsteam:	Kühl, Fessler, Dieng, Ostwald, Smolka, Niedermeier, Dumps, Spanninger, Essig, Monet, Heimpel, Schlierenzauer
Analogzellen	P.Gärtner (IMS)
CMOS-Vorteiler	B.Laquai (vorm.IMS)
Betreuung IMS	H.Richter
Betreuung FH	F.Förster, W.Ludescher
	Lu, FH Ravensburg-Weingarten, den 13.08.99

Digital programmierbarer Verstärker



Entwurf:	Fachhochschule Ulm Bearbeiter: Thomas Bückle, Thomas Rösch Betreuer: Prof. Dipl.-Phys. Gerhard Forster
Layouterstellung:	Fachhochschule Ulm (Mixed Signal-Entwurf) Analog-Teil: Full Custom Design Digital-Teil: Standardzellen
Technologie:	CYB 0,8 µm CMOS A/D, Fa. AMS
Chipfertigung:	Fa. AMS, Österreich, über Europractice
Herstelldatum:	II. Quartal 1999
Kostenträger:	MPC-Gruppe Baden-Württemberg
Chipdaten:	Chipfläche: 1,6 x 1,5 mm^2 Gehäuse LCC 44 Funktionsblöcke: Analogteil: Operartionsverstärker, Widerstandsnetzwerk, Schalter Digitalteil: ca. 150 Gatter
Funktion:	Es handelt sich um einen NF-Verstärker, dessen Spannungsverstärkung über eine digitale Schnittstelle in 3 dB-Schritten zwischen 0 dB und 45 dB programmierbar ist. Eine Digitalanzeige gibt den aktuellen Wert aus.