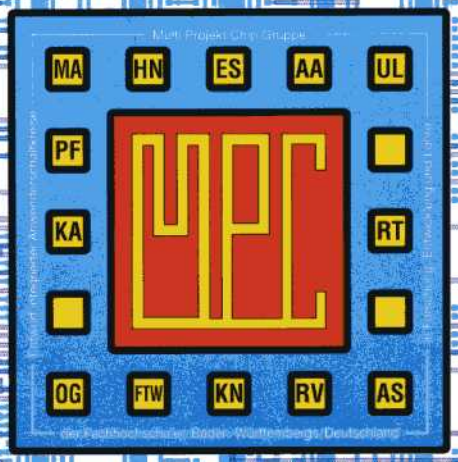
MULTIPROJEKTCHIP GRUPPE

BADEN-WÜRTTEMBERG

MPC-Workshop Juli 2004

Albstadt-Sigmaringen



MULTIPROJEKT CHIP-GRUPPE

BADEN - WÜRTTEMBERG

MPC-Workshop Juli 2004

Albstadt-Sigmaringen

Cooperating Organization
Solid-State Circuits Society Chapter
IEEE Germany Section

Herausgeber: Fachhochschule Ulm

© 2004 Fachhochschule Ulm

Das Werk und seine Teile sind urheberrechtlich geschützt. Jede Verwertung in anderen als den gesetzlich zugelassenen Fällen bedarf deshalb der vorherigen schriftlichen Einwilligung des Herausgebers Prof. A. Führer, Fachhochschule Ulm, Prittwitzstraße 10, 89075 Ulm.

Adressen der

MULTIPROJEKT-CHIP-GRUPPE (MPC-Gruppe)

BADEN - WÜRTTEMBERG

http://www.mpc.belwue.de

Fachhochschule Aalen
Prof. Dr. Bartel, Postfach 1728, 73428 Aalen
Tel.: 07361/576-107, Fax: -324, Email: manfred.bartel@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.: 07131/504400, Fax: /252470, Email: clauss@fh-heilbronn.de

Fachhochschule Karlsruhe
Prof. Dr. Koblitz, Postfach 2440, 76012 Karlsruhe
Tel.: 0721/925-2238, Fax: -2259, Email: koblitz@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. Ludescher, Postfach 1261, 88241 Weingarten
Tel.: 0751/501-9685, Fax: -9876, Email: ludescher@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

Entwurf des RISC-Kerns FHOENIX

zur Integration in SOC Designs

Dirk Jansen

ASIC Design Center der

University of Applied Sciences, Offenburg,

Badstrasse 24 d.jansen@fh-offenburg.de

Es wurde ein neuer Prozessorkern FHOENIX, in großen Teilen kompatibel mit dem Kern FHOP, entwickelt und auf FPGA verifiziert. Der Kern soll die Basis für zukünftige SOC-Designs bilden. Gegenüber dem bestehenden Design wurde die Performance um den Faktor 5 gesteigert. Die verwendete Harvard-Architektur ermöglicht gleichzeitigen Zugriff auf Programme wie Daten. Der Bootvorgang erfolgt durch Laden eines Images aus einem Flash- Memory über eine serielle SPI-Schnittstelle. Die zur Applikations-Entwicklung notwendige Integrierte Entwicklungsumgebung wurde durch Modifikation der FHOP- IDE gewonnen und verfügt jetzt auch über einen C-Compiler.

1. Einführung

Mikroprozessoren sind das Herz und Zentrum aktueller System On Chip Entwürfe. Sie ermöglichen die Intelligenz und Flexibilität, die ein moderner ASIC heute benötigt. Durch die Programmierbarkeit und Bereitstellung von universalen Schnittstellen sind ASICs damit für einen breiteren Einsatzbereich anwendbar. Fast alle Steuerfunktionen sind programmierbar.

Welchen Prozessorkern soll man integrieren? Die Auswahl ist auf den ersten Blick groß und reicht von den kleinen 8051-Typen mit 8 bit Architektur bis hin zu 32 bit Boliden wie den ARM-7TDI, ARM-9, LEON und weiteren firmenspezifischen Prozessorkernen. Hierbei sind neben der Leistungsfähigkeit der Kerne auch Fragen wie

- Unterstützung durch Betriebssysteme, Compiler und Debug-Tools,
- Speicherbedarf an lokalem wie globalem Speicher,
- Verfügbarkeit von Peripheriemodulen wie serielle Schnittstellen, USB usw.
- Aufwand für Bussystem,
- Komplexität und Verifikation durch verfügbare Entwurfstools

zu berücksichtigen.

Für die im Hochschulbereich machbaren Designs gilt dies in besonderem Maße. Das Wort von „den Kanonen, mit denen auf Spatzen geschossen wird", klingt in den Ohren. Der Einsatz von 32bit Kernen mit LINUX Performance ist in den meisten Industrie-Applikationen ein totaler Overkill. 32 bit Kerne, gleich welcher Art, erfordern hohen Speicherbedarf auf dem Chip und sind nur in den neuesten deep submicron CMOS-Technologien wirtschaftlich realisierbar. Hinzu kommen die hohen Lizenzkosten, die die Verwendung dieser Kerne für kleinindustrielle Anwendungen verschließt.

Offene Kerne wie der LEON [1] sind wiederum zu groß, um wirtschaftlich in kleine Anwendungen integriert zu werden (LEON ist eine SPARC – 32 bit Architektur) und erfordern zudem zu hohen Begleitaufwand. Der Vorteil der C – Programmierbarkeit kann nur genutzt werden, wenn entsprechende Speichervolumina (> 4 MByte) bereitgestellt werden. Andererseits ist die 8bit Architektur des 8051 Kerns trotz zahlreicher „revival" Konzepte nicht mehr zeitgemäß.

Am ASIC Design Center der Fachhochschule Offenburg wurde deshalb schon 1994 ein eigener 16 bit Prozessorkern „FHOP" entwickelt [2], der seitdem in zahlreichen Applikationen eingesetzt wurde und durch immer neue Erweiterungen im Peripheriebereich aktuell ist. Nicht zuletzt diente diese Entwicklung auch der Lehre und der Untersuchung der mit einem eigenen Kern verbundenen Gestaltungsmöglichkeiten.

Nach 10 Jahren erfolgreichem Einsatz des FHOPs steht eine Aktualisierung dringend an. Dabei sind die Erfahrungen aus 10 Jahren SOC-Integration und Systementwicklung einzubringen. Die technolo-gischen Randbedingungen haben sich stark geändert, ebenso die Entwurfsziele und Paradigmen, denen solche Entwürfe folgen müssen. Heute ist wichtig:

- Der Kern muß vollständig in einer synthetisierbaren Entwurfssprache (VHDL) beschrieben sein.
- Er sollte keine technologiebezogenen Macros enthalten und auf jede CMOS Technologie abbildbar sein.
- Das Steuerungskonzept muß RISC-artig sein mit hohem Durchsatz und einfachem Pipelining.
- Die Speicheranforderungen müssen moderat sein, da Speicher auf dem Chip weiterhin sehr viel Fläche benötigen.
- Der Kern muß selbst klein und einfach zu integrieren sein.
- Statischer Design und Optimierung auf niedrigen Leistungsverbrauch sind notwendig für mobile Applikationen.
- Der Kern muß kompatibel zu bestehenden Peripheriebaugruppen sein und ein einfaches Bussystem mit „Plug and Play" – Eigenschaften aufweisen.
- Es muß ein komplettes SW-Entwicklungstool einschließlich C Compiler effektiv zur Verfügung stehen.

Für die FH Offenburg stellte sich deshalb die Frage, entweder die FHOP- Linie ganz aufzugeben und auf einen kommerziellen bzw. offenen Kern umzusteigen (8051 Derivat, ARM oder LEON), oder durch ein „Facelifting" den bestehenden Entwurf zu aktualisieren und an die obigen Forderungen anzupassen. Im letzteren Fall waren zahlreiche Kompatibilitätsprobleme einzubeziehen, um die Entwicklungen der letzten Jahre einzubringen, insbesondere im Bereich der Peripheriemodule.

Das Argument der Lehre wie auch der Wunsch über die volle Kontrolle des Designs wiegen schwer. Der im folgenden näher beschriebene Kern FHOENIX, Synonym für „**FHO**- Processor with **En**hanced **I**nstruction **Ex**ecution", stellt deshalb einen Kompromiß dar zwischen den obigen Anforderungen und dem existierenden Konzept und ist spezifisch auf die Anforderungen kleinerer SOC – Designs ausgerichtet, die noch nicht die volle Funktionalität eines 32 bit LINUX erfordern.

2. Architekturkonzept

Die noch im FHOP verwendete John von Neumann Architektur ist zwar sehr günstig was den Bedarf an Speicher betrifft, erfordert aber wegen des Multiplexing von Daten und Instruktionen auf dem Bus eine komplexe Steuerung und deshalb prinzipiell mehrere Takte je Befehl. Ein RISC Konzept, das nur einen Befehl/Takt realisiert, läßt sich nur in einer Harvard Architektur verwirklichen, in der die Busse zum Zugriff auf Datenspeicher und Programmspeicher physikalisch getrennt sind Abb. 1. Ist der Programmspeicher selbst nicht als ROM ausgeführt, ist zusätzlich für den Boot-Vorgang sogar noch ein dritter Speicherblock, das BOOTROM, vorzusehen, der adressmäßig an der Stelle des Speicherplans anzuordnen ist, wo der Prozessor nach dem Reset den ersten Befehl ausführt.

Neben diesen 3 separierten Speichern sind die Peripheriebaugruppen wie SIO, PIO, IIC,USB... über einen Buscontroller anzuschließen. Wegen des vergleichsweisen niedrigen Durchsatzes reicht hier bei FHOENIX eine 8 bit breite Schnittstelle. Der Buskontroller, im klassischen Design mit Tristate-Gattern als echter Bus ausgeführt, wird bei FHOENIX durch Multiplexerstrukturen ersetzt. Dies hat viele Vorteile:

- Keine potentialmäßig undefinierten Netze und damit keine Buskeeper mehr,
- Deutlich niedrigerer Leistungsverbrauch,
- Bessere Abbildung in FPGA- Strukturen,
- Einfacher im Timing zu modellieren.

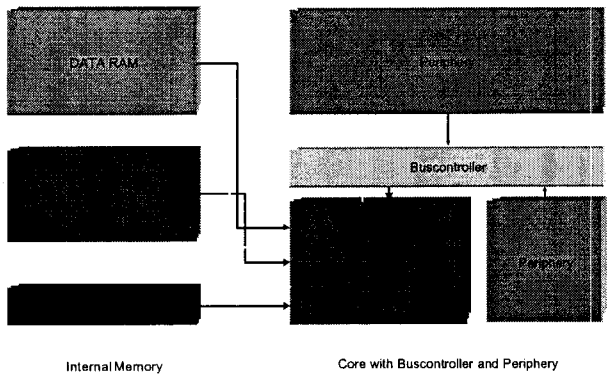


Abbildung 1: Harvard Architektur mit Buscontroller und 3 Speichern

Dies entspricht den modernen Buskonzepten wie z.B. beim AMBA-Bus, allerdings ohne die volle Komplexität zu übernehmen. Allerdings ist dies kein reines „Plug and Play"-Konzept mehr, da der Buskontroller bei Erweiterung angepaßt werden muß. Da dieser jedoch sowieso jeweils neu synthetisiert wird, stellt dies keine praktische Einschränkung dar. Man könnte sogar durch „generate" Funktionen den Vorgang automatisieren.

Die zahlreichen Verbindungsleitungen, die bei Multiplexerstrukturen benötigt werden, stellen innerhalb des Chips kein Problem dar. Ebenso können die Busse breit ausgelegt werden und müssen nicht im multiplex betrieben werden.

Ein größeres Problem stellt das RAM dar, welches nun die auszuführenden Befehle aufnimmt (Programm-RAM). Das Programm muß irgendwie in dieses RAM geladen werden. Klassisch bei RISC ist hier die Verwendung eines CACHE –Speichers, der sich automatisch mit dem aktuell ausgeführten Kode lädt. Allerdings ist hierfür ein erheblicher Steuerungsaufwand erforderlich. In REAL-Time Applikationen, die interruptgesteuert sind und geringe Latenzzeiten erfordern, ist bei Verwendung eines Cache keine Deterministik mehr gegeben, was in den meisten Fällen so stört, das Abschalten des Cache erforderlich wird. In unseren Anwendungsfällen ist dies typisch eigentlich immer der Fall, woraus sich die Verwendung eines Cache eigentlich von selbst verbietet.

Es bleibt nur das Konzept, den Programmspeicher als eine Art **Scratchpad-Memory** zu verwenden, der bei Bedarf programmgesteuert mit den aktuellen Routinen geladen wird. Dieses Laden soll über die Peripherie, z.B. über eine effektive SPI-Schnittstelle in serieller Form erfolgen. Hierfür sind entsprechende Mechanismen im Betriebssystem zu etablieren. Die Situation wird noch dadurch kompliziert, das gewöhnlich auch im Daten-Ram entsprechende Konstanten oder Variablen verwaltet werden müssen.

Andererseits gewinnt man den enormen Vorteil, das nun der Adreßraum des Prozessors, der mit 16 bit für heutige Verhältnisse als ziemlich eingeschränkt gelten muß, an Bedeutung verliert und praktisch beliebig große Programme bearbeitet werden können. Serielle Speicher mit SPI- Schnittstelle sind heute in winzigen Gehäusen mit wenigen Anschlußpins in Megabyte Dimensionen preiswert verfügbar.

Eine Einschränkung gibt es allerdings bezüglich großer Datenobjekte, insbesondere Bilder, hierfür reicht der Adreßraum definitiv nicht aus und eine 32 bit Verarbeitung ist unabdingbar. Dies ist aber nicht der vorgesehene Einsatzbereich.

Der fest im IC zu speichernde Programmkode läßt sich damit auf einen „Loader" beschränken, der auch während des Bootvorgangs verwendet wird und in dem BOOTROM angeordnet wird. Dieses wird wiederum so klein, daß auf eine echte ROM-Struktur verzichtet werden kann, d.h. das ROM wird ersetzt durch ein synthetisiertes Gatternetz. Der Kern bleibt dabei „soft", d.h. für die Synthese werden keine besonderen Macros benötigt.

Die RAMs, sowohl das Programm- Ram wie das Daten-Ram, sind als klassische synchrone Single-Port- Rams ausgeführt. Erste Konzeptüberlegungen, die Harvard-Struktur mit einem Dual-Port RAM zu realisieren, wurden aufgegeben, nachdem sich zeigte, daß DP-RAMs nahezu die 2,5 – Fache Fläche im Chip benötigen wie SP-RAMs und deshalb extrem unwirtschaftlich sind (in ASICs, in FPGAs gibt es kaum Einschränkungen). Die vom FHOP aus Gründen

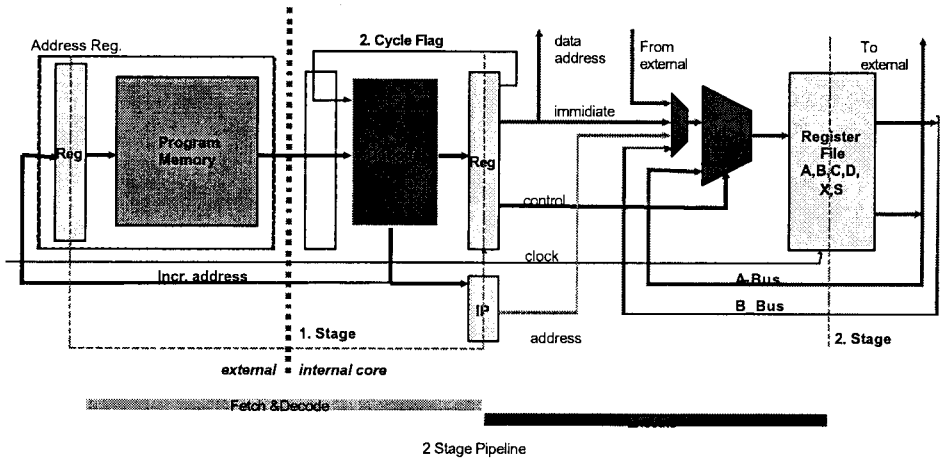


Abbildung 2: Befehls-Pipelining bei FHOENIX

der Kompatibilität übernommene variable Befehlslänge von 1 Byte-, 2 Byte- und 3 Byte- Befehlen erfordert allerdings eine besondere Auslegung des Programm-RAMs als eine aus 2 einzelnen Byte-breiten SP- RAMs bestehende Struktur, vergleiche Abbildung 8. Hierauf wird später noch einmal eingegangen.

Die Befehle sollen in nur einem Takt abgearbeitet werden. Dies erfordert eine Pipeline – Struktur Abb. 2, die aus einem Speicherzugriff für die Phase *Fetch and Decode* und einer Verarbeitungsphase *Execute* besteht. Dies ist die kleinste Steuerungsstruktur, die möglich ist. Sie ist geeignet für Register-Register-Befehle wie MOV, ADD, SUB..., die damit in einem Takt ohne Rücksicht auf Datenabhängigkeiten verarbeitet werden können.

Die genauere Analyse des Befehlssatzes ergab allerdings, daß eine ganze Klasse von *Load –Store* Befehlen, insbesondere wenn die Adressen aus Registern genommen werden sollen, mit dieser Simpelstruktur nicht auskommt. Für diese Befehlsklassen wird ein zusätzlicher Takt benötigt, da die Adreßinformation in dem zuvor bearbeiteten Befehl erst in der „Execute" Phase berechnet wird und deshalb nicht zeitgleich zur Verfügung steht. Dies gilt insbesondere für komplexe Befehle wie CAL, RET, PSH, POP..., die sowohl Daten zum/vom Speicher transportieren wie auch Register manipulieren. Man hätte natürlich in RISC-Manier auf diese Befehle verzichten können, allerdings hätte das erheblichen Software-Overhead bedeutet. Nach einigen Abwägungen haben wir uns entschieden, jegliche Beschränkung in der Verwendung von Befehlen (instruction dependency) zu vermeiden. Andernfalls wären hohe Forderungen an den Compiler bzw bei Handprogrammierung an den Assembler zu stellen.

In der Hardware bedeutet das, daß bei den betroffenen Befehlen ein 2.Cycle Flag gesetzt wird, was zu einer weiteren Taktphase führt. Hierdurch wird der Kodierungsraum allerdings verdoppelt, was den Aufwand im Decoder erhöht. Andererseits stellt dies eine extrem einfache und effektive Mikroprogrammierung aus 2 Mikroinstruktionen dar, die im VHDL-Kode durch 2 aufeinanderfolgende Zeilen der Kodierungstabelle leicht zu handhaben sind.

Der Datenpfad ist wie beim FHOP klassisch aus Flipflop-Registern und einer ALU aufgebaut Abb. 3. Es gibt 4 Universalregister A,B,C,D von 16 bit, zusätzlich ein dediziertes Indexregister X und ein Stackregister S. Die Indexfunktion des B- Registers mit Autoindex ist aus Kompatibilitätsgründen übernommen worden, wie auch alle anderen FHOP-Befehle mit gleicher Wirkung übernommen worden sind. Bestehende Assemblerprogramme sind damit in den meisten Fällen direkt lauffähig, auch wenn dabei die besonderen Fähigkeiten des erweiterten Instruktionssatzes nicht zur Geltung kommen. Damit kann ein großer Teil der bereits bestehenden Software übernommen bzw. angepaßt werden.

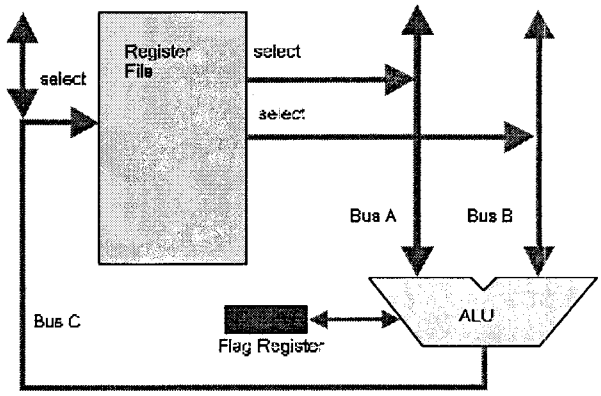


Abbildung 3 Datenpfad

Für den Zugriff zum Datenspeicher ist ebenfalls eine Pipeline-Stufe zu durchlaufen, da das externe RAM als synchroner Speicher über ein Eingangsregister für die Adresse verfügt, welches zusammen mit den getakteten Flipflops der Register zwei Pipelinestufen bildet. Der Ablauf des Zugriffs wie auch des Abspeicherns läuft dabei parallel zur Befehlsverarbeitung. Es ist deshalb prinzipiell nicht möglich, Programme aus dem Datenspeicher zu verarbeiten

Es ist auch nicht möglich, Daten aus dem Programmspeicher zu lesen, da nicht vom Steuerpfad und vom Datenpfad gleichzeitig auf den Single Port RAM zugegriffen werden kann. Allerdings ist es möglich, auf den Programmspeicher zu schreiben (ev. auch zu lesen), wenn das Programm aus dem BOOTROM ausgeführt wird, da in diesem Fall die entsprechenden Busse frei sind.

Weitere Einzelheiten der Architektur zeigt Abbildung 4. Die ALU wird noch ergänzt um einen 16b x 16b Multiplizierer. Die Datenpfade werden durch Multiplexer geschaltet, die wiederum von der Dekodiereinheit direkt angesteuert werden. Diese besteht im wesentlichen aus einem (großen) Dekodiernetzwerk mit 8 bit Input und daraus abgeleiteten und über ein Register synchronisierten Steuersignalen, die die Tore und die ALU direkt steuern.

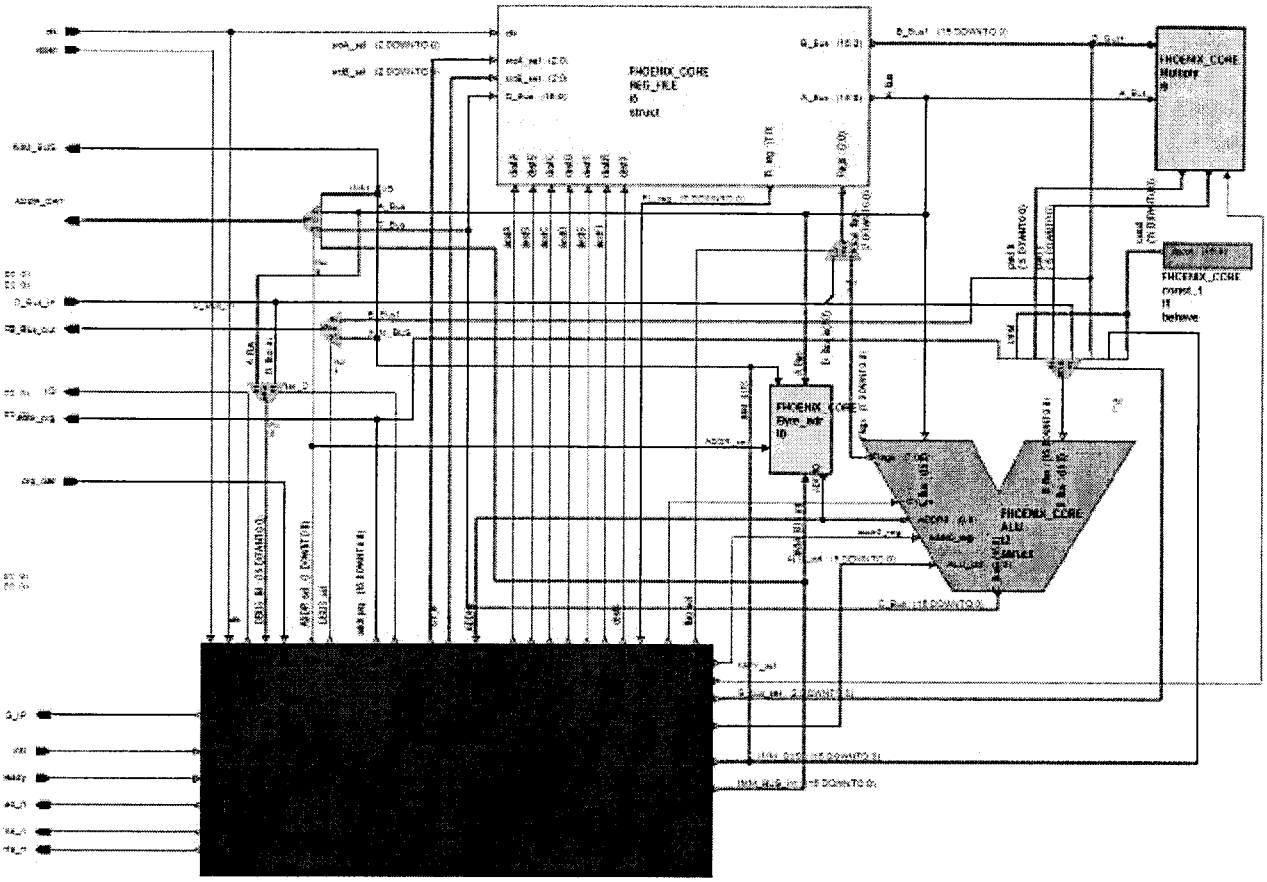


Abbildung 4: Details der Kernarchitektur

3. Architektur des Befehlssatzes

Die Forderung nach einer Kompatibilität mit dem bestehenden FHOP- Befehlssatz, zumindest in der mnemonischen Formulierung und Wirkung, hat viele Konsequenzen. Auf der Ebene des Objektkodes war keine Kompatibilität erforderlich, hier konnte frei kodiert werden. Allerdings sind folgende Randbedingungen zu beachten:

- Alle Instruktionen sind in 8 bit zu kodieren, damit ergeben sich 256 mögliche Befehle (Kodierraum).
- Der 8 bit OperationsKode soll auch, wie bei FHOP, die Registeradressierung beinhalten.
- Alle Register-Register-Befehle kodieren damit in nur 1 Byte.
- Für *immediate*- Werte wie Adressen und Konstanten wird eine 16 bit *Extension* verwendet, damit ergibt sich ein 3 Byte Befehl.
- Für die IO-Befehle PIN und POT sowie den Befehl SWI werden 2 Byte benötigt.

Damit ist das Befehlsformat, wie bei RISC-Prozessoren sonst üblich, nicht fest, sondern kann als 1-Byte, 2-Byte oder 3-Byte Befehl auftreten. Die Instruktionen müssen aber in allen Bestandteilen gleichzeitig aus dem Programmspeicher ausgelesen werden, was somit eine 48 bit Busschnittstelle erfordert. Da 48 keine Potenz von 2 ist, erfordert dies eine komplizierte Architektur des Programmspeichers aus 2 unabhängigen SP - RAMs von jeweils 16 bit Breite, die über Multiplexer geeignet kombiniert werden. Dabei kann es sein, daß beide Speicher mit der gleichen Adresse oder mit unterschiedlichen Adressen ausgelesen werden, was einen Adreßaddierer erfordert Abb. 6. Abgesehen vom zusätzlichen Aufwand, der sich noch in Grenzen hält, bedeutet dieses Konzept eine zusätzliche Verzögerung im Speicherzugriff auf Daten.

Andererseits ist der extrem kompakte Befehlssatz mit 169 Ein-Byte-Befehlen von insgesamt 249 Befehlen

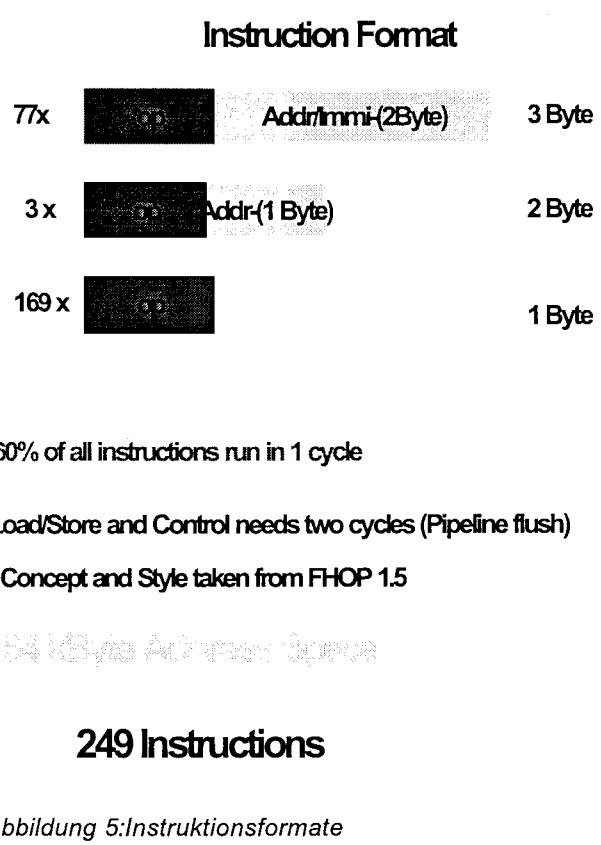


Abbildung 5:Instruktionsformate

sehr effektiv und führt zu sehr kompaktem Kode, der mit einer einheitlichen 16 bit Befehlsstruktur in RISC-Manier nicht zu erreichen gewesen wäre. 60% aller Befehle werden in nur einem Takt verarbeitet.

Eine weitere wichtige Entscheidung betrifft die Beibehaltung der absoluten Adressierung auch bei Sprung- und CAL – Befehlen, was die Handprogrammierung vereinfacht, allerdings keinen verschiebbaren Kode ermöglicht. Das Programm muß also für die entsprechende Position, wo es bei Laufzeit ausgeführt werden soll, zuvor compiliert werden. Dies wurde von FHOP so übernommen. Es ist jedoch vorgesehen, entsprechende Relativ-Sprünge noch in den Befehlssatz aufzunehmen, um echten *relocatable code* erzeugen zu können. Dies entspricht eher dem angestrebten *Scratchpad-Memory* Konzept.

Der bestehende Befehlssatz wurde durch mehrere wichtige Adressierungsmöglichkeiten und Erweiterungen ergänzt:

- Indirekte Adressierung sowohl für lesen als auch schreiben ohne Autoindex (LDX,STX),
- Stack- relative Adressierung zur einfachen Verwendung von lokalen Variablen.
- Wichtige Befehle wirken jetzt auf alle Register, d.h. die meisten Register sind gleichwertig.

Das *Load-Store*- Prinzip ist weitgehend umgesetzt. Nur für die bestehenden Befehle wurden Kombinationen von Register und Memory noch zugelassen (ADD adrr, A), diese sollen aber in den Programmiertools vermieden werden.

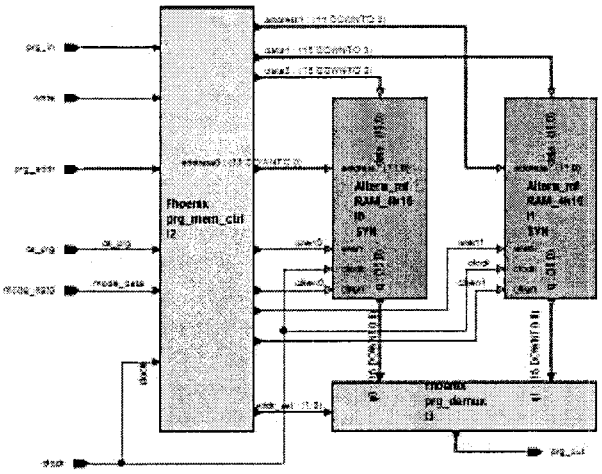


Abbildung 6: Speicherorganisation des Programmspeichers

Ein großer Teil der Befehle wurde eingeführt, um einen effektiven C-Compiler zu ermöglichen, der in wesentlichen Teilen bereits besteht. Wichtigste Vorraussetzung war dafür die Ermöglichung von lokalen Variablen auf dem Stack sowie eine effektive Pointerbehandlung durch das Indexregister. Ein vollständig orthogonaler Befehlssatz war wegen des beschränkten Kodierraums von 8 bit nicht realisierbar. Tabelle 1 zeigt den Kodierraum und alle kodierten Befehle.

1.sel		D,A	A,B	A,C	A,D	A,X	A,S	D,C	C,D	B,A	C,B	B,C	B,D	C,A	D,B		M
2.sel		A	B	C	D	X	S	PC	F	A	B	C	D	X	S	addr	M
Type	hex	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
CTRL	0x	NOC	NOP	INV C	INV D	INV B	ENI	DIS	CBI C	MPL A,B	MPL A,C	MPL A,D	MPL B,C	MPL B,D	MPL C,D	HLT	
ARITH	1x	DEC A	DEC B	DEC C	DEC D	DEC X	ORI c16	CBI B	CBI A	INC A	INC B	INC C	INC D	INC X	CBI D	XRA c16	XRA M
LOGIC 1	2x	CLR B	ANA B	ANA C	ANA D	CLR C	ANI c16	RAL	RAR	CLR A	XRA B	XRA C	XRA D	INV A	SWP	ANA c16	ANA M
LOGIC 2	3x	CLR D	ORA B	ORA C	ORA D	ORA X	XRI c16	RLC	RRC	MPH A,B	MPH A,C	MPH A,D	MPH B,C	MPH B,D	MPH C,D	ORA c16	ORA M
LOAD 1	4x	LDA c16,A	LDA C16,B	LDA C16,C	LDA C16,D	LDA C16,X	POP A	POP B	POP C	LDI c16,A	LDI c16,B	LDI c16,C	LDI c16,D	LDI c16,X	LBA C16,A	PIN port	LDA M
LOAD 2	5x	LDX A	LDX B	LDX C	LDX D	POP X	POP D	POP X	POP F	LDS C16,A	LDS C16,B	LDS C16,C	LDS C16,D	LDS C16,X	LBS C16,A	LBX A	
STORE1	6x	STA A,c16	STA B,c16	STA C,c16	STA D,c16	STA X,c16				PSH A	PSH B	PSH C	PSH D	PSH X	STB A,c16		STA M
STORE2	7x	STX A	STX B	STX C	STX D	STX X	SSX A	SUB D,X	PSH F	STS A,C16	STS B,C16	STS C,C16	STS D,C16	STS X,C16	SSB A,c16	POT port	SUB B,X
MOV 1	8x	MOV D,A	MOV A,B	MOV A,C	MOV A,D	MOV A,X	MOV A,S	MOV D,C	MOV C,D	MOV B,A	MOV C,B	MOV B,C	MOV B,D	MOV C,A	MOV D,B	MOV X,A	ADD B,X
MOV 2	9x	SUI C16,A	SUI C16,B	SUI C16,C	SUI C16,D	SUI C16,X	SUI C16,S	MOV S,A	MOV F,A	ADI C16,A	ADI C16,B	ADI C16,C	ADI C16,D	ADI C16,X	ADI C16,S	MOV IP,A	ADD D,X
ADD 1	Ax	ADD D,A	ADD A,B	ADD A,C	ADD A,D	ADD A,X	ACI c16	ADD D,C	ADD C,D	ADD B,A	ADD C,B	ADD B,C	ADD B,D	ADD C,A	ADD D,B	ADD c16	ADD M
ADD 2	Bx	ADX A	ADX B	ADX C	ADX D	ADD X,A	ADC B	ADC C	ADC D	ADS C8,A	ADS C8,B	ADS C8,C	ADS C8,D	ADS C8,X	ADD C,X	ADC c16	ADC M
SUB 1	Cx	SUB D,A	SUB A,B	SUB A,C	SUB A,D	SUB A,X	SCI c16	SUB D,C	SUB C,D	SUB B,A	SUB C,B	SUB B,C	SUB B,D	SUB C,A	SUB D,B	SUB c16	SUB M
SUB 2	Dx	SBX A	SBX B	SBX C	SBX D	SUB X,A	SUC B	SUC C	SUC D	SBS C8,A	SBS C8,B	SBS C8,C	SBS C8,D	SBS C8,X	SUB C,X	SUC c16	SUC M
CMP 1	Ex	CMP D,A	CMP A,B	CMP A,C	CMP A,D	CMP A,X	CMI c16	CMP D,C	CMP C,D	CMP B,A	CMP C,B	CMP B,C	CMP B,D	CMP C,A	CMP D,B	CMP c16	CMP M
JMP	Fx	JNC c16	JNZ c16	JNO c16	JCY c16	JEQ c16	JGE c16	JGT c16		MOV A,IP	RET cn8	RET	SWI cn8	INTR	CAL addr	JMP c16	NOC

Tabelle 1: Instruktionen im Kodierraum von 8 bit

Die Architektur ermöglicht Interrupts, die wie normale Befehle abgearbeitet werden. Hold- und Ready Funktionalität sind nicht realisiert, da gewöhnlich nur auf den internen Speicher mit definierten Zugriffszeiten gearbeitet wird. Eine Erweiterung der Architektur um einen weiteren Bus- Slave macht ebenfalls keinen Sinn. DMA erfolgt sowieso programmgesteuert über die SPI-Schnittstelle. Ein externer Speicheranschluß zur Erweiterung des Programmspeichers ist zwar grundsätzlich möglich, widerspricht aber den Intentionen des Designs und ist auch nicht notwendig. Mit 16 kByte ist der Programmspeicher als Scratchpad-Memory bereits ziemlich fett angelegt.

4. Boot Vorgang

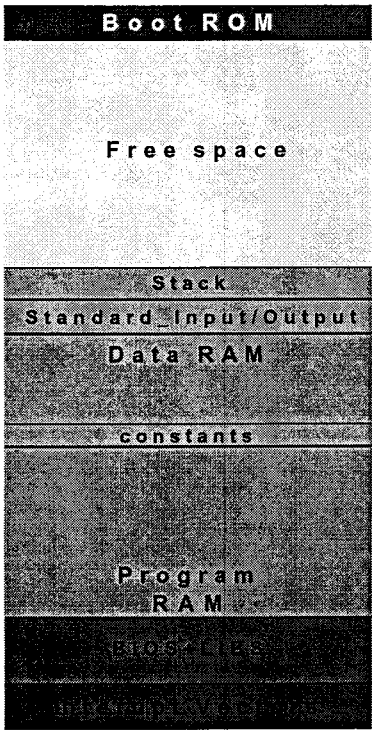
Der Boot-Vorgang soll hier noch einmal besonders erläutert werden. Nach einem *Reset* wird der erste Befehl aus der Adresse $FF80 ausgeführt. Diese Adresse liegt damit am Beginn des BOOTROMS, welches den Bereich von $FF80 ... $FFFF einnimmt. Damit stehen maximal 128 Byte Programmspeicher zur Verfügung, um einen LOADER und eine BLOCKCOPY-Routine zu programmieren. Beide Routinen können auch vom laufenden Programm als Betriebssystemroutinen aufgerufen werden, wobei die Parameter in den Registern übergeben werden.

Der LOADER steuert in der Standardversion das SPI-Modul an, welches mit einem externen seriellen FLASH-Speicher großer Kapazität verbunden ist. Dieser Flash wird dabei wie ein Plattenlaufwerk behandelt, das die einzelnen Programmsequenzen als (quasi) Files enthält. Die ersten beiden Files bilden Initialisierungssequenzen, die als Image in das Scratchpad-Memory kopiert werden. Anschießend wird auf den Anfang dieser Sequenz gesprungen und diese ausgeführt. Dabei werden gewöhnlich weitere Images geladen, eventuell auch Konstanten in den Datenspeicher.

Der LOADER kann auch während normaler Programmausführung angesprochen werden und entsprechende Pages laden bzw. transferieren.

Es ist vorgesehen, den Loader noch in Richtung eines Ladevorgangs über die serielle Schnittstelle (COM) zu erweitern, um auch von der SPI – Schnittstelle unabhängig werden zu können bzw. einen eleganten Debug- Betrieb zu ermöglichen.

Die Reseteinsprungadresse wurde auf $FF80 gelegt, um den unteren Memorybereich für das *Scratchpad*



freizuhalten. Hier liegt auch die Interrupt Vektor Tabelle, die ebenfalls beim Bootvorgang geladen werden muß.

Die Speicheraufteilung zeigt Abb. 7. Der freie Bereich kann entweder mit Daten- oder Programm-RAM bei Bedarf belegt werden. Dies macht aber erst dann Sinn, wenn sehr kompakte Speichermacros zur Verfügung stehen, was erst in 0.13 CMOS und darunter der Fall ist. Für die aktuellen Designs erscheint der Speicher absehbar ausreichend.

Abbildung 7: Speicherplan

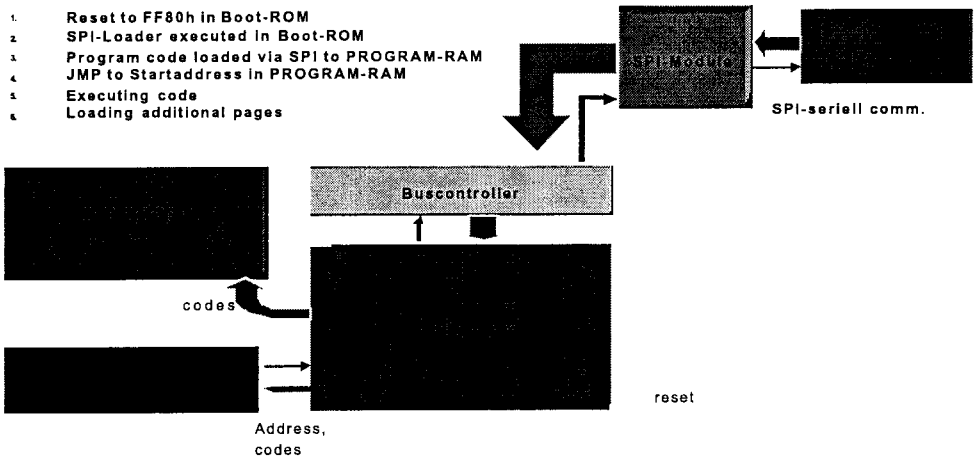


Abbildung 8:Bootvorgang

5. Software –Entwicklung für FHOENIX

Der FHOENIX-Core kann nur dann in Applikationen verwendet werden, wenn die notwendige Softwareunterstützung verfügbar ist. Die bestehende, in weiten Teilen modifizierbare FHOP-IDE wurde deshalb so umgestaltet, daß sie als FHOENIX –IDE verwendet werden kann. Dies beinhaltete einen neuen Assembler, wobei der CRASH-Assembler von *Daniel Vogel*, bei einigen kleinen Erweiterungen, durch einen neuen Konfigurationsfile (FOEHNIX.bef) in die Lage versetzt wurde, FHOENIX-Kodes zu übersetzen. Ebenfalls angepaßt wurde der GCompiler (auf Basis LCC), der nun in der Lage ist, einfache C-Programme in Assemblerkode zu compilieren Abb. 10. Die Arbeiten hieran sind allerdings noch nicht zum Abschluß gekommen und es gibt noch Fehlermeldungen in einigen Fällen. Es ist beabsichtigt, das der Compiler die komplette ANSI-C Spezifikation unterstützen soll.

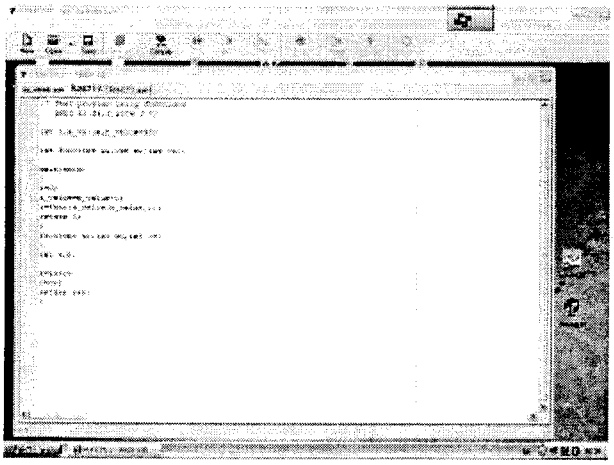


Abbildung 9: Compiler

Wichtig für die Entwicklung ist auch ein Debugger, der ebenfalls auf die FHOP-Version zurückgeht und sich dort sehr bewährt hat (Abb. 11). Debugger/Simulator sind mit der Hardware-Simulation auf VHDL- Ebene abgeglichen. Die Tools sollen noch um Download-Werkzeuge ergänzt werden, um entwickelte Programme auf einer Emulationsplatine ablaufen lassen zu können. Hierfür ist jedoch ein minimales Betriebssystem erforderlich, welches noch programmiert werden muß.

Der Simulator verfügt über eine DDE- Schnittstelle, die mit Softmodulen, die als eigenständige Programme auf dem PC laufen, kommuniziert. Solche Softmodule können Hardware-Schnittstellen wie PIO, SIO oder

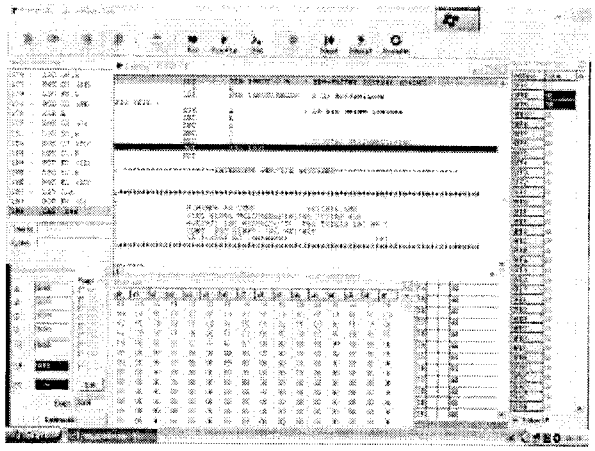


Abbildung 10: Simulator

TIMER- Modelle sein und ermöglichen die Simulation eines kompletten Systems auf dem PC in Verbindung mit dem Simulator. Die bestehenden Softmodule sind auch mit der FHOENIX-IDE lauffähig. Eine Terminal-artige Schnittstelle wurde noch ergänzt.

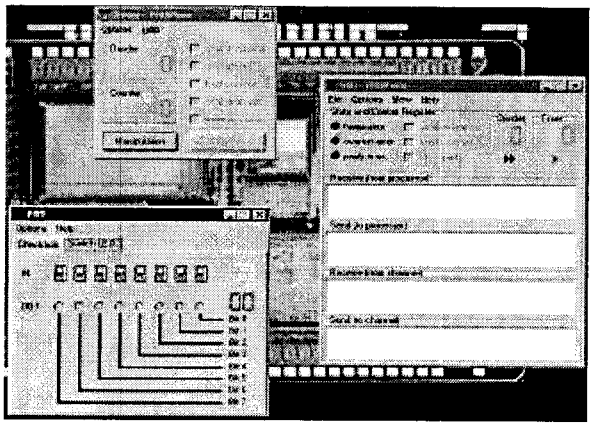


Abbildung 11: Softmodule für FHOENIX (DDE)

6. Stand der Entwicklung

Für den Kern wie auch alle Peripheriemodule liegen inzwischen synthesefähige VHDL- Kodes vor, die auf FPGA funktional emuliert und verifiziert wurden. Derzeit wird an einem Qualifikations-Chip für eine 0.35 CMOS AMI Technologie gearbeitet, in dem der Kern und alle verfügbaren Module enthalten sind,. Nach den ersten, nicht optimierten Synthese-ergebnissen ist mit einer garantierten Taktfrequenz von etwa 20 MHz im ersten Durchlauf zu rechnen. Der Kern selbst ist für etwa 50 MHz ausgelegt, so daß eine Performance von 50 MIPS erreicht werden wird. Gegenüber dem bestehenden FHOP Kern bedeutet das eine Verbesserung um den Faktor 5, wobei der Kern nur unwesentlich größer ist (was auf den Multiplizierer zurückzuführen ist).

Zur weiteren Ausreifung des Designs sind noch zahlreiche Arbeiten im Hardware- (Synthese) wie Softwarebereich (Compiler, Betriebssystem) durchzuführen, ein breites Feld für Studenten, die sich für Prozessorarchitektur und Real Time Operating Systeme interessieren.

7. Referenzen:

[1] *LEON*: http://www.estec.esa.nl/wsmwww/leon/leon.html

[2] *Vollmer, W*: Aufbau eines Mikrokontroller-systems auf der Basis des Mikroprozessorkerns FHOP und Entwurf der zugehörigen Peripheriemodule als integrierter Anwenderschaltkreis (ASIC). Diplomarbeit an der FH-Offenburg, 1996.

Verifikation eines Mikrokontrollersystems durch Emulation auf FPGA und darauf folgende Synthese und Routing in einer 0,35µm Technologie

Daniel Bau, Prof. Dr. Dirk Jansen, ASIC Design Center
Fachhochschule-Offenburg, Badstrasse 24, D-77652 Offenburg
Tel. 0781/205-274

Einen neu entwickelter Prozessorkern FHOENIX gilt in ein Mikrocontrollersystem einzubinden. Das Mikrocontrollersystem ist als Haward Architektur ausgelegt. Alle verfügbare Peripheriemodule aus bestehenden Entwürfen sind an das neue Buskonzept anzupassen und durch Emulation zu verifizieren.

1. Einleitung

Im Jahre 1994 wurde an der Fachhochschule Offenburg im ASIC Design Center der Grundstein für das langjährige Projekt FHOP (First Homemade Operational Prozessor) von Prof. Dr. Dipl.-Ing D. Jansen gelegt. Zuerst entstand ein reiner Mikroprozessorkern, der mit seiner 16Bit Architektur und seinem Befehlssatz mit dem häufig in der Industrie eingesetzten 80C51 Mikrocontroller vergleichbar ist. In zahlreichen Studien- und Diplomarbeiten wurde der Prozessorkern FHOP zu einem kompletten Mikrocontrollersystem weiter-entwickelt. Das Mikrocontrollersystem FHOP weist eine J.v. Neumann Architektur auf, indem die Daten und das Programm in einem Speicher vereint sind. Das Betriebssystem (BIOS) der Architektur befindet sch in einem 2kByte großem ROM. Die Verarbeitung eines Befehls des Prozessors FHOP erfolgte durchschnittlich in 8 Takten und dessen Befehlssatz war im Vergleich mit heutigen Prozessoren recht uneffizient.

Um den Anforderungen der Industrie und der Wirtschaftlichkeit gerecht zu werden, wurde ein neuer Prozessor entwickelt namens FHOENIX (FHO-Processor with Enhanced Instruction Execution). Hier wurde eine Harvard Architektur eingesetzt, dessen Struktur einen getrennten Programm- und Datenspeicher beinhaltet. Dadurch ist eine schnellere Verarbeitungsgeschwindigkeit möglich, verwirklicht durch den gleichzeitigen Datenaustausch auf Programm- und Datenbus. Das ROM wurde in diesem System durch ein rein kombinatorisches Boot-ROM ersetzt, welches die Treiberroutinen enthält, um einen seriellen Bus (SPI) anzusteuern, welcher über einen externen Flash das BIOS in den Programm-RAM herunterlädt. Durch eine Pipeline-Struktur innerhalb des Prozessorkerns erfolgt die Befehlsver-arbeitung in einem Taktzyklus. Der Befehlssatz ist im Vergleich mit den FHOP-Instructions umfangreicher und flexibler gestaltet. Hauptziel der Arbeit war, den Prozessor FHOENIX zu verifizieren und in ein Mikrocontrollersystem einzubinden. Dadurch mussten die vorhandenen Peripherie-einheiten an das neue Buskonzept angepasst werden. Nach erfolgreichen FPGA-Tests soll dann ein ASIC in einer 0,35µm Technologie entstehen.

Das Mikrocontrollersystem FHOENIX weist folgende Komponente auf:

- Prozessorkern FHOENIX,
- 8kBx16 Programm- und 4kBx16 Daten-RAM,
- 128 Byte BOOT-ROM,
- Buscontroller,
- Interrupt Controller,
- Watchdog,
- 3 Timer und Real-Time-Clock,
- 3 parallele und 1 serielle Schnittstelle,
- Accustic Human Interface,
- I²C- Schnittstelle,
- Serial Peripheral Interface (SPI),
- USB Controller,
- induktive Q-PSK und RFID-Schnittstelle.

2. Mikrocontrollersystem FHOENIX

Das Mikrocontrollersystem FHOENIX wurde als Haward Architektur realisiert, d.h. das System hat einen Programmspeicher, indem die Anwendungsprogramme abgelegt sind und ausgeführt werden, und einen Datenspeicher, in welchem die Konstanten und spezifische, je nach Anwendung benötigte Daten abgelegt werden. Über ein rein kombinatorisches Bootrom, welches eine Treiberroutine für die Ansteuerung des SPI-Interface enthält, wird bei einem Systemstart über einen externen SPI-Datenflash der Programmspeicher mit Daten bzw. dem BIOS des Mikrocontrollersystems beschrieben.

In dem alten Mikrocontrollersystem FHOP befand sich ein ROM, welches das BIOS des Systems enthielt. In dem System war das Fertigungsrisiko größer, da ein Fehler im BIOS durch das fixe ROM nicht mehr korrigierbar war. Mit dem externen Flash kann das BIOS jederzeit angepasst und erweitert werden.

Der Buszugriff im alten System FHOP wurde über Tristates gesteuert, welche im Mikrocontrollersystem FHOENIX durch Multiplexer ersetzt wurden. Gründe für den Austausch der Tristates waren:

- Multiplexer hat im Vergleich mit einem Tristate einen geringeren Stromverbrauch,
- Multiplexer kann eins zu eins auf einem FPGA abgebildet werden,
- Mux-Ansteuerung ähnelt einem Speicher,
- Mux-Durchlaufzeiten können einfacher modelliert werden.

Die Peripherieeinheiten besitzen einen 8Bit breiten und die beiden Speicher einen 16Bit breiten Datenbus. Der Programmspeicher liefert einen 24Bit breiten Datenstrom, welcher direkt in den Kern zur Befehlsverarbeitung weitergegeben wird.

Das in Abbildung 2.0 gezeigte Blockschaltbild gibt Aufschluss über das Mikrocontrollersystem FHOENIX.

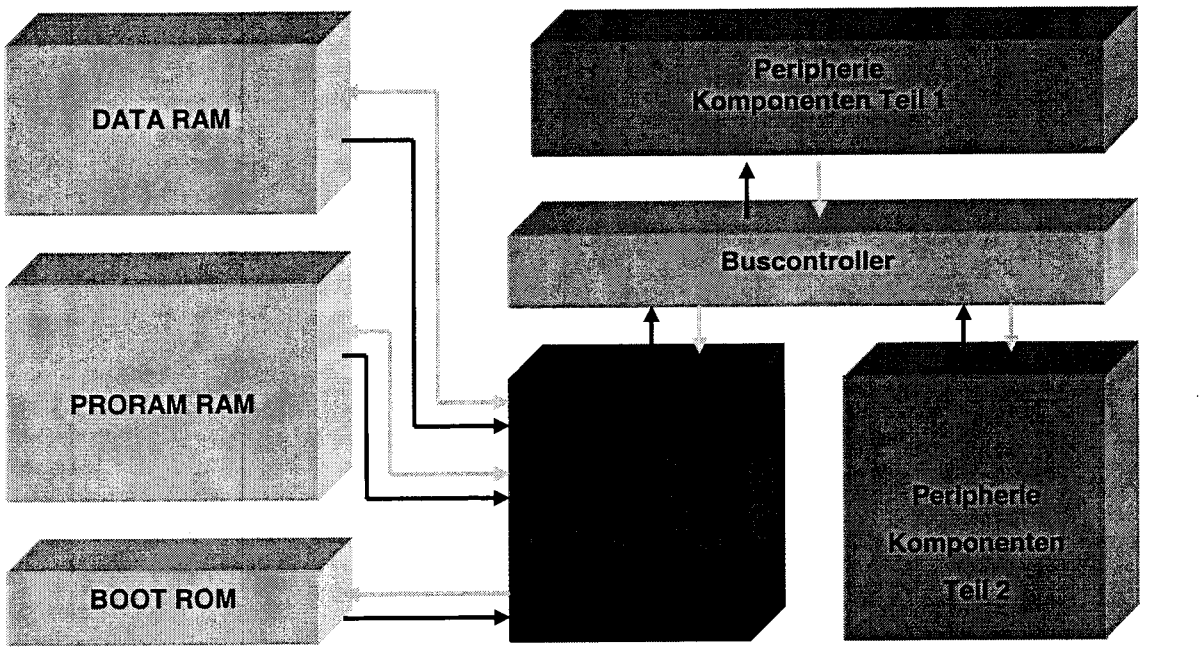


Abb.2.0: Mikrocontrollersystem FHOENIX

3. BOOT-Vorgang des Mikrocontrollersystem FHOENIX

Über das 128 Byte große BOOT-ROM wird bei einem Systemstart das SPI-Interface angesteuert, welches über einen externen SPI-Daten-Flash das BIOS des Systems in den Programmspeicher kopiert.Der Flash wird bei der Initialisierung ab Adresse 0 ausgelesen. Nachdem 4 Pages heruntergeladen und in den Programmspeicher geschrieben wurden, wird an die Adresse 3E0h gesprungen. An der Adresse 3E0h befindet sich das Initialisierungsprogramm für die Perpheriekomponenten.

Nach der Initialisierung kann erneut in das BOOT-ROM gesprungen werden, worüber man Datensegment in den Datenspeicher kopieren kann oder Anwendungsprogramme in den Programmspeicher überträgt. Die BOOT-ROM Routine wurde variabel gestaltet, man bestimmt ab welcher Adresse vom Flash gelesen wird, wieviel Bytes man lesen möchte und an welche Stelle man die Daten in dem Programm-speicher ablegen will.

Im BOOT-ROM befindet sich noch eine Copy-Block-Routine (Einsprungadresse 0FFEEh), mit der es möglich ist, Daten von einer Adresse im Speicher an eine andere Adresse im Speicher zu kopieren. Abbidung 3.0 zeigt den Bootvorgang im Mikrocontrollersystem FHOENIX.

4. Speicherkonzept des Mikrocontrollersystem FHOENIX

Das Programm-RAM des Mikrocontrollersystems FHOENIX weist einen Speicherbereich von 8KBx16 auf. In dem RAM wird das BIOS des Systems und die Anwendungsprogramme abgelegt. Ab Adresse 0400h (Beginn des Anwendungsprogramms) wird die erste Instruktion des Anwenders erwartet.

Das BIOS lässt sich in folgende Bereiche unterteilen:

- Interruptsprungtabelle:
  Wird ein Hardware-Interrupt ausgelöst, wird über den Interruptcontroller die Adresse im BIOS des entsprechenden Interrupts eingelesen und dann in die dazugehörige Interruptroutine gesprungen.

- BIOS Konstanten:
  Definition der Portadressen der Peripherie-einheiten und Sprungadressdefinitionen.

- Subroutinen:
  Interuptserviceroutinen\Treiberroutinen für die Peripherieeinheiten

- Initialisierung:
  Initialisierung der Hardware des Mikrocontroller-systems FHOENIX

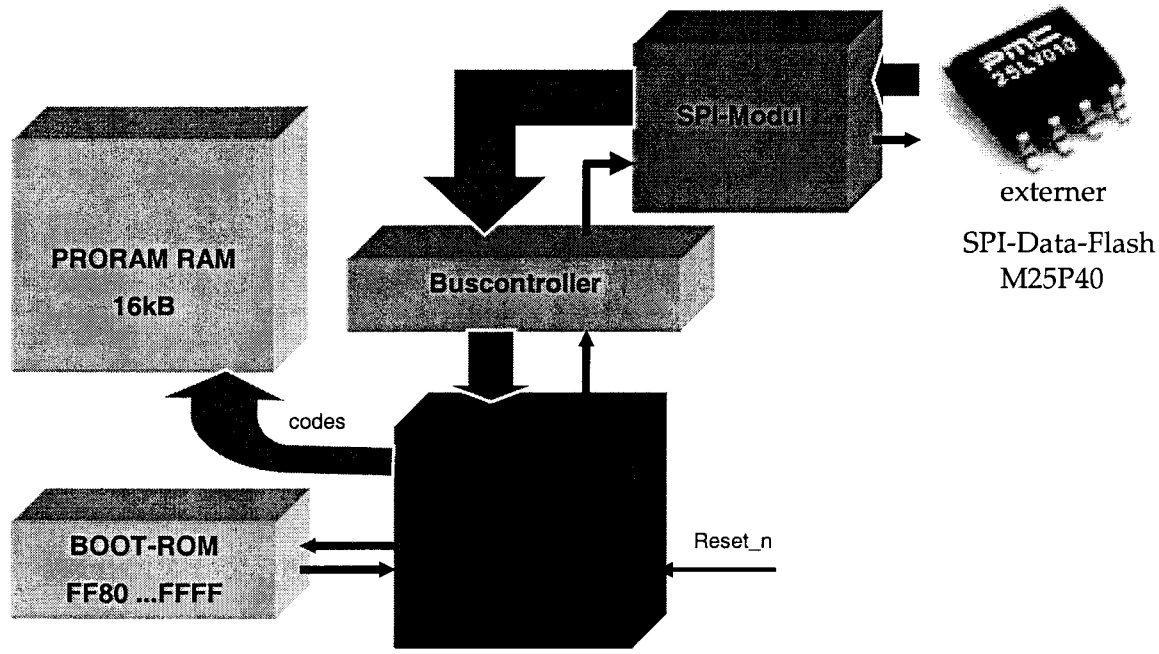


Abb. 3.0: BOOT-VORGANG im µC-System

Abbildung 4.0 zeigt das Speicherbild des Programm-Rams und gibt Aufschluss über die Adressaufteilung. Das Daten-RAM des Mikrocontrollersystem FHOENIX weist einen Speicherbereich von 4KBx16 auf. Der RAM dient hauptsächlich zum ablegen Anwendungsprogrammdaten.

Im unteren Bereich des Daten-RAMs liegen für die im BIOS verwendeten Subroutinen Variablen und Konstanten. Ab dem Bereich 4100h kann der Anwender seine Daten ablegen und weiterverarbeiten.

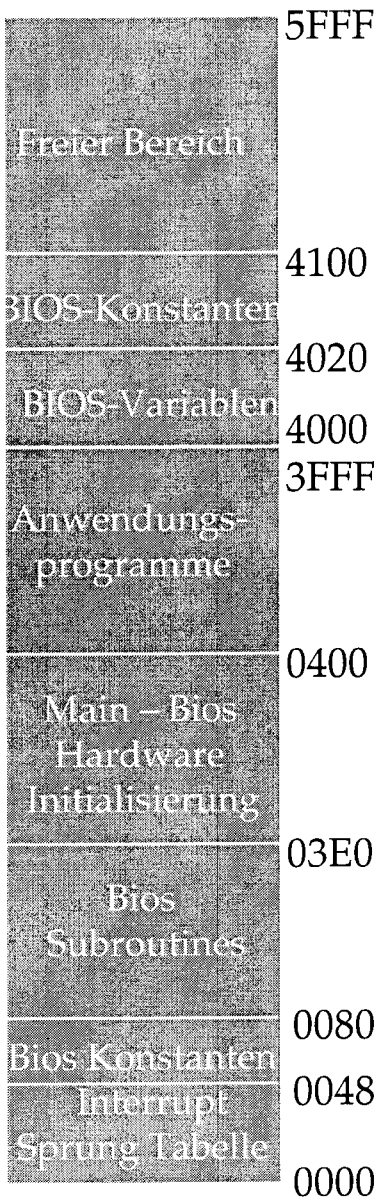


Abb. 4.0: Speicheraufteilung des Programmspeichers und des Datenspeichers

5. Syntheseergebnisse

Nach erfolgreicher Simulation wurde der VHDL-Code mit dem Syntheseprogramm „Leonardo Spectrum" synthetisiert und mit dem Programm Quartus platziert und geroutet auf dem FPGÁ. Über den HDL-Designer kann man direkt die Synthese starten, welche dann voll automatisch ihr Syntheseskript erstellt. Man wählt nur die gewünschte Zieltechnologie aus. Als Zieltechnologie wurde hier der FPGA-Chip der Firma Altera EP20K1500EBC652-3 eingesetzt, welcher mit 1,5 Mio. Gatter einer der größten dieser Art ist. Der FPGA befindet sich auf dem Entwicklungsboard der Firma EL Camino. Das Board beinhaltet alle nötigen Komponenten für eine komfortable Entwicklung.

FPGA-Syntheseresultate:

Beschreibung	Verwendet	Verfügbar
IOs (IN/OUT-Ports)	69	488
LCs (Logic Cells)	6.457	51.840
Memory Bits	217.344	442.368

Tabelle 5.0: Syntheseergebnisse mit dem FPGA APEX20KE

Das Design wurde auf eine Systemfrequenz von 24MHz optimiert. Von dem im FPGA APEX20KE verfügbaren 1,5 Millionen Gatter (ca. 50000 Altera spezifischen Logik Zellen) wurden nur 12 % benutzt.

Die ASIC-Synthese wurde mit dem Programm Design Analyzer von Synopsys durchgeführt. Zur Synthesedurchführung gibt es zwei Möglichkeiten. Man ruft das Programm Synopsys auf und führt über die entsprechenden Menüpunkte die verschiedenen Synthese-Schritte manuell durch. Viel effizienter ist es, den Synthesevorgang zu automatisieren. Die Syntheseanweisungen werden dabei in eine Skript-Datei geschrieben. Im Syntheseprogramm wird dann lediglich das TCL-Skipt aufgerufen und die Anweisungen werden dann automatisch abgearbeitet.

Tabelle 5.1 zeigt die Syntheseergebnisse des kompletten Mikrocontrollersystems in der 0,35µm Mietec-Technologie.

ASIC-Syntheseresultate:

Beschreibung	Benutzt
Number of Ports	84
Number of Cells	9.416
Number of nets	10.349

Tabelle. 5.1:
Synopsys Syntheseergebnis
in der 0,35µm Technologie

6. Zusammenfassung

In dieser Arbeit wurde ein komplettes Mikrocontrollersystem zusammengestellt, simuliert, synthetisiert und in einem FPGA emuliert. Der Schwerpunkt lag im Prozessorkern FHOENIX, welcher sich gerade in der Entwicklungsphase befindet. Über Testprogramme wurde der komplette Befehlssatz des Prozessors verifiziert. Das Assembler-Testprogramm wurde so programmiert, das es bei einem nicht erwünschten Ergebnis in eine Error-Routine springt. Zuerst wurde die Prozessorfunktionalität mit dem Programm Modelsim in der Simulation nachgewiesen und zuletzt in einem FGA realisiert.

Nach erfolgreichen Prozessortests wurden die verschiedenen Peripherieeinheiten schrittweise an das neue Buskonzept angepasst und jede Komponente simuliert und in einem FPGA im Zusammenspiel mit dem Prozessor emuliert. Jede Komponente wurde angesteuert und deren Funktion nachgewiesen.

Als FPGA-Zieltechnologie wurde der FPGA-Chip der Firma Altera EP20K1500EBC652-3 eingesetzt. Die ASIC-Synthese in der 0,35µm Mietec-Technologie benötigt 9.416 logische Zellen. Zum Abschluß der Arbeit muss noch das Routing mit dem Programm Encounter von der Firma Cadence durchgeführt werden. Nach erfolgreicher Nachsimulation wird der Chip zur Produktion freigegeben.

7. Literaturverzeichnis

[1] Handbuch der Electronic Design Automation
Herausgegeben von Dirk Jansen

[2] Nidal Fawaz, „Development of CP-DQPSK Modulator and Demodulator using VHDL for Inductive Data Transmission",

[3] Frank Baier, „Entwicklung einer digitalen bidirektionalen Schnittstelle nach dem induktiven Gütemodulationsverfahren in VHDL gemäß ISO/IEC 14443-A"

[4] Artur Kurz, „Redesign eines USB-Controllers in VHDL und Erstellung der zugehörigen Treiber-routinen zum Emulieren mit einem Mikroprozessorkern auf FGPA"

Entwurf eines LVDS-Leitungstreibers für On-Chip-Kommunikation

Immanuel Kurz, Prof. Dr.-Ing. Heinz-Peter Bürkle

Fachhochschule Aalen, EDA-Zentrum, Beethovenstraße 1, 73430 Aalen

Tel. 07361 / 576 – 247, Fax 07361 / 576 – 324

ikurz@rz.fh-aalen.de, heinz-peter.buerkle@fh-aalen.de

Im Rahmen einer Studienarbeit wurden zwei Schaltungskonzepte für LVDS-Leitungstreiber in einer 0,25µm-SiGe:C-BiCMOS – Technologie untersucht. Hauptanwendungsgebiet ist die hochbitratige Datenübertragung zwischen Subsystemen auf dem Chip („On-Chip-Kommunikation").

Das erste Schaltungskonzept beruht auf einer Stromquelle mit umschaltbarer Polarität. Es wurde als Schaltung mit MOS-Transistoren ausgeführt.

Das zweite Schaltungskonzept hat zwei gesteuerte Spannungsquellen als Grundlage. Bei der schaltungstechnischen Umsetzung wurden ausschließlich Bipolartransistoren verwendet.

Es stellte sich heraus, dass bei kleinen Leitungslängen oder geringen Datenraten beide Schaltungskonzepte möglich sind, das erste Konzept jedoch vorteilhafter ist. Für größere Leitungslängen und höhere Datenraten muss man das zweite Schaltungskonzept verwenden.

Aufgrund des größeren Einsatzbereichs wurde das zweite Konzept in ein IC-Layout überführt.

1 Einführung

1.1 Motivation für serielle und differenzielle On-Chip-Datenübertragung

Die ständig steigende Komplexität integrierter Schaltkreise, die bereits heute so genannte „Systems on Chip" (SoC) ermöglicht, und der Trend zu Mixed-Signal-ICs führen zu zwei Schwierigkeiten:

- Immer mehr Daten müssen zwischen Subsystemen auf dem Chip ausgetauscht werden. Dazu verwendete parallele Bussysteme benötigen daher einen zunehmenden Anteil der Chipfläche.
- Höhere Taktfrequenzen mit steileren Flanken auf den Signalleitungen sorgen dafür, dass an die Signalleitungen angrenzende analoge Schaltungsteile immer größeren elektromagnetischen Störungen ausgesetzt sind.

Für diese Probleme sind folgende Lösungsansätze denkbar:

- Der Verbrauch an Chipfläche durch eine Vielzahl paralleler Leitungen lässt sich reduzieren, indem man die Daten seriell mit entsprechend höherer Taktfrequenz überträgt.

  Man benötigt dann Multiplexer und Demultiplexer, um parallele Datenströme in serielle Datenströme und wieder zurück zu wandeln.

  Multiplexer und Demultiplexer benötigen ihrerseits Chipfläche. Man muss also im Einzelfall entscheiden, ob man durch die eingesparten Signalleitungen mehr Chipfläche gewinnt, als man für die Multiplexer und Demultiplexer benötigt.

  Durch die höhere Taktfrequenz wird außerdem das Problem, dass andere Schaltungsteile gestört werden, verstärkt.
- Die Störungen, die die Signalleitungen aussenden, lassen sich vermindern, indem man für die Datenübertragung differenzielle Signale verwendet.

  Man benötigt hierfür Schaltungen, die nichtdifferenzielle Signale in differenzielle Signale umsetzen, und Leitungstreiber mit genügend kleinem Ausgangswiderstand, um die parasitären Kapazitäten der Signalleitungen hinreichend schnell umzuladen. Am Ende von Signalleitungen sind außerdem Leitungsempfänger notwendig.

1.2 Differenzielle Datenübertragung

Bei differenziellen Signalen wird nicht die Spannung einer Signalleitung gegen Masse, sondern die Differenzspannung zwischen zwei Signalleitungen ausgewertet. Die Differenzspannung wird auch als Gegentaktanteil des Signals oder als Gegentaktspannung bezeichnet.

Man versucht, die Gleichtaktspannung[1] möglichst konstant zu halten, was aufgrund von Unsymmetrien der Leitungstreiber und der Leitungsführung in der Regel jedoch nicht ganz gelingt.

Wenn die Gleichtaktspannung aber weitgehend konstant ist, werden kaum Störungen ausgesendet.

Ein Nachteil differenzieller gegenüber nicht-differenzieller Datenübertragung besteht darin, dass man die doppelte Anzahl an Signalleitungen sowie umfangreichere Treiberschaltungen benötigt.

Ein Vorteil ist neben der geringeren Störaussendung auch die geringere Störempfindlichkeit:

Elektromagnetische Störungen von außen wirken sich meist gleichmäßig auf beide Signalleitungen aus. Die Differenzspannung zwischen den Signalleitungen wird erheblich weniger gestört als die Spannung einer Signalleitung gegen Masse.

Daher reicht es aus, wenn man für das differenzielle Signal einen wesentlich geringeren Spannungshub vorsieht, als man bei einem nicht-differenziellen Signal verwenden müsste.

Der geringere Spannungshub bewirkt zudem, dass die durch Unsymmetrien der Leitungstreiber und der Leitungsführung verursachten Schwankungen der Gleichtaktspannung geringer werden. Dadurch sinkt die Störaussendung nochmals.

Im Zusammenhang mit differenzieller Datenübertragung wird heute häufig der Begriff LVDS verwendet. LVDS bedeutet „Low Voltage Differential Signaling", d. h. Verwendung differenzieller Signale mit geringen Spannungspegeln.

Es gibt mehrere konkurrierende Normen, die LVDS spezifizieren. Zwei wichtige Normen sind:

- IEEE 1596.3–1996 „IEEE Standard for Low Voltage Differential Signals (LVDS) for Scalable Coherent Interface (SCI)"
- ANSI/TIA/EIA-644 „Electrical Characteristics of Low Voltage Differential Signaling (LVDS) Interface Circuits"

Beide Normen wurden im Jahr 1996 veröffentlicht.

ANSI/TIA/EIA-644 wurde 2001 für den Anschluss mehrerer Leitungsempfänger an denselben Leitungstreiber erweitert (ANSI/TIA/EIA-644-A).

IEEE 1596.3-1996 wurde im Jahr 2002 vom IEEE zurückgezogen und wird vom IEEE nicht mehr unterstützt. Im Folgenden wird daher auf IEEE 1596.3-1996 nicht weiter eingegangen.

[1] Die Gleichtaktspannung ist der Mittelwert der Spannungen der beiden Signalleitungen gegen Masse.

ANSI/TIA/EIA-644 spezifiziert zwei Signalleitungen, die im Folgenden als LVDS + und LVDS - bezeichnet werden. Die Differenzspannung zwischen LVDS + und LVDS – (die Gegentaktspannung) muss betragsmäßig im Bereich von 0,247 V bis 0,454 V liegen, die Gleichtaktspannung im Bereich von 1,125 V bis 1,375 V. LVDS + und LVDS - werden auf der Empfängerseite mit einem Widerstand von 100 Ω abgeschlossen.

ANSI/TIA/EIA-644 wird im Folgenden als LVDS-Spezifikation bezeichnet.

Die in den nachfolgenden Kapiteln beschriebenen Leitungstreiber wurden daraufhin untersucht, ob Gegentakt- und Gleichtaktspannung am Ausgang des Leitungstreibers die LVDS-Spezifikation erfüllen.

1.3 Verwendete Halbleitertechnologie

Der LVDS-Leitungstreiber soll in der 0,25µm-SiGe:C-BiCMOS-Technologie von IHP (Innovations for High Performance microelectronics, ehemals: Institut für Halbleiterphysik, Sitz: Frankfurt (Oder), [1]) realisiert werden.

Der verwendete Prozess beinhaltet Silizium-Germanium-npn-Transistoren mit einer Transitfrequenz von bis zu 80 GHz sowie Silizium-MOS-Transistoren.
Pnp-Transistoren stehen nicht zur Verfügung.

1.4 Ziel und Einordnung der Arbeit

Die hier beschriebene Arbeit stellt eine einführende Grundlagenuntersuchung im Rahmen eines Gesamtprojekts dar.

Inhalt der Arbeit sind vor allem Schaltungskonzepte für den LVDS-Leitungstreiber mit dem Schwerpunkt der On-Chip-Kommunikation.

Weitere Arbeiten befassen sich

- mit der Extraktion der On-Chip-Leitungskapazitäten,
- mit Empfängerschaltungen,
- mit hochbitratigen, Strom sparenden und selbst synchronisierenden Multiplexer- und Demultiplexerschaltungen
- sowie mit einer weiteren Optimierung der Leitungstreiber.

Ziel des Gesamtprojekts ist die Erstellung einer Bibliothek, die LVDS-Zellen sowohl für On-Chip- als auch für Off-Chip-Kommunikation enthält. Die Realisierung eines Testchips und dessen messtechnische Evaluierung sollen das Gesamtprojekt abschließen.

2 Schaltungskonzepte für den Leitungstreiber

2.1 Stromquelle mit umschaltbarer Polarität

Prinzip

Ein mögliches Konzept für den Leitungstreiber ist eine Stromquelle mit umschaltbarer Polarität. Je nach logischem Wert des zu übertragenden Signals wird der Strom in der einen oder in der anderen Richtung durch die angeschlossene Last geleitet. Dabei bleibt zunächst unberücksichtigt, dass in der LVDS-Spezifikation auch die Gleichtaktspannung festgelegt ist. Abbildung 1 veranschaulicht dieses Konzept.

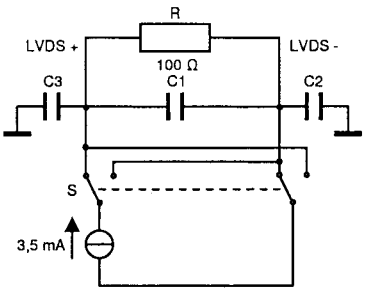


Abbildung 1 Stromquelle mit umschaltbarer Polarität

C1, C2 und C3 in Abbildung 1 stellen die parasitären Kapazitäten zwischen den beiden Signalleitungen bzw. zwischen jeweils einer Signalleitung und Masse dar.

Da die Signalleitungen nach Möglichkeit nebeneinander verlegt werden, gilt: $C_3 \approx C_2$.

Es ist zu beachten, dass die Gegentaktspannung[2] während der Abtastung durch den Leitungsempfänger bitmusterunabhängig sein sollte. Vernachlässigt man die Setup- und Hold-Zeiten des Leitungsempfängers, so sollte sich die Gegentaktspannung in einer Taktperiode annähernd auf den Endwert einstellen.

Geht man z.B. davon aus, dass sich die Gegentaktspannung nach drei RC-Zeitkonstanten annähernd auf den Endwert eingestellt hat, so muss in einer ersten Abschätzung gelten:

$$f_{Takt,\max} < \frac{1}{3 \cdot R \cdot C}$$

$$\text{mit } R \approx 100\ \Omega,\ C \approx C_1 + \frac{C_2}{2} (\text{da } C_2 \approx C_3)$$

$f_{Takt,\max}$ ist die maximale Taktfrequenz, mit der der Leitungstreiber betrieben werden kann.

[2] Differenzspannung zwischen LVDS + und LVDS -

C_1 und C_2 sind unter anderem proportional zur Länge der Signalleitungen.

Die Länge der Signalleitungen begrenzt somit unmittelbar die maximal mögliche Taktfrequenz.

On-chip dürften C_1 und C_2 Werte im Femto- oder Picofaradbereich aufweisen.

Umsetzung in eine Schaltung

Abbildung 2 zeigt eine mögliche Realisierung des Schaltungskonzeptes. Dabei wurde ein Schaltungsvorschlag aus [2] abgeändert: Die H-Brücke enthält nun auch p-Kanal-MOSFETs. Der Strom durch den Abschlusswiderstand wird nicht über eine Spannungsreferenz eingestellt; die Gleichtaktspannung am Ausgang wird nicht geregelt.

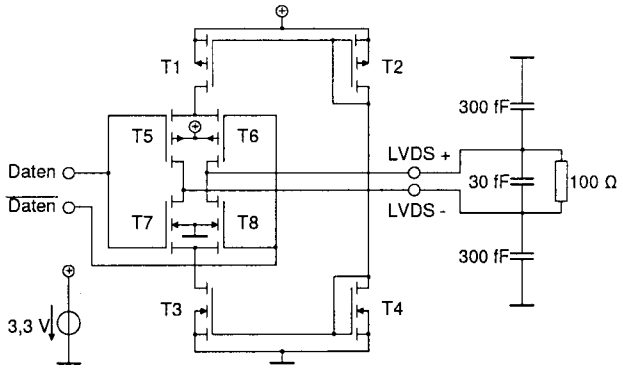


Abbildung 2 Realisierung mit H-Brücke aus n-Kanal- und p-Kanal-MOSFETs

In Abbildung 2 werden die Eingangssignale als Daten und $\overline{\text{Daten}}$ bezeichnet. LVDS + und LVDS - bilden das Ausgangssignal des Leitungstreibers.

Die Gatelänge ist bei allen Transistoren die minimale Gatelänge des verwendeten Prozesses.

Die Gatebreiten betragen nach Optimierung:

T1:	58	µm
T2:	2,9	µm
T3:	20	µm
T4:	1	µm
T5, T6:	58	µm
T7, T8 :	20	µm

Die Stromquelle in Abbildung 1 wurde durch die beiden Stromspiegel T1/T2 und T3/T4 in Abbildung 2 realisiert. Dem Schalter S in Abbildung 1 entspricht die H-Brücke T5 - T8 in Abbildung 2.

Die Stromspiegel stellen bei dieser Dimensionierung der Transistoren einen Strom von etwa 3,5 mA durch T1 bzw. durch T3 ein. Aufgrund des Faktors 20 zwischen den Gatebreiten von T1 und T2 bzw. zwischen den Gatebreiten von T3 und T4 wird dafür nur ein Strom von 0,19 mA durch T2 bzw. T4 benötigt.

T5 – T8 prägen den Strom von etwa 3,5 mA dann in wechselnder Polarität der Last am Ausgang ein:

- Bei Daten[3] = 1 leiten T6 und T7. T5 und T8 sperren. Durch die Ausgangslast fließt ein Strom von etwa 3,5 mA. Es stellt sich bei der angegebenen Beschaltung des Ausgangs nach dem Umladen der parasitären Leitungskapazitäten eine Spannung von + 350 mV zwischen LVDS + und LVDS - ein.
- Bei Daten = 0 leiten T5 und T8. T6 und T7 sperren. Durch die Ausgangslast fließt wieder ein Strom von etwa 3,5 mA, nun jedoch in umgekehrter Richtung. Es stellt sich bei der angegebenen Beschaltung des Ausgangs nach dem Umladen der parasitären Leitungskapazitäten eine Spannung von - 350 mV zwischen LVDS + und LVDS - ein.

Vorteil dieser Schaltung ist, dass der von ihr aufgenommene Strom nahezu in vollem Umfang dazu verwendet wird, die Last zu treiben. Lediglich ein Strom von 190 µA wird zur Einstellung des Arbeitspunktes benötigt.

Vorstehende Schaltungsbeschreibung geht von einer Betriebsspannung von 3,3 V und einer Temperatur von 27 °C aus.

Bei anderen Betriebsspannungen oder anderen Temperaturen stellen sich andere Ströme und Spannungen ein; in der Simulation erfüllt die Gegentaktspannung am Ausgang im eingeschwungenen Zustand jedoch für Betriebsspannungen von 3,0 V bis 3,6 V und Temperaturen von 20 °C bis 60 °C die LVDS-Spezifikation.

Lässt man Abweichungen der Gegentaktspannung von der LVDS-Spezifikation von maximal 10 mV zu - was bei On-Chip-Kommunikation unkritisch ist, wenn man den Leitungsempfänger entsprechend auslegt -, so erhält man bei gleichem Spannungsbereich einen Temperaturbereich von 10 °C bis 80 °C. Erlaubt man Abweichungen von maximal 15 mV, so erhält man bei gleichem Spannungsbereich einen Temperaturbereich von 0 °C bis 90 °C.

Zur Ermittlung dieser und aller folgenden Betriebsspannungs- und Temperaturbereiche wurden Monte-Carlo-Simulationen durchgeführt, um Prozessschwankungen zwischen verschiedenen ICs und den Mismatch innerhalb eines ICs zu berücksichtigen.

Werte, die sich auf genau eine Betriebsspannung und Temperatur (3,3V; 27 °C) beziehen, wurden hingegen immer ohne Berücksichtigung von Prozessschwankungen und Mismatch ermittelt.

[3] Daten, $\overline{\text{Daten}}$: Logischer Wert 0 ≙ 0 V gegen Masse
Logischer Wert 1 ≙ 3,3 V gegen Masse

Durch Variation der Gatebreiten der Transistoren kann einfach ein anderer Betriebsspannungs- oder Temperaturbereich eingestellt werden, in dem die Gegentaktspannung am Ausgang der LVDS-Spezifikation genügt oder von dieser nur leicht abweicht.

Der Strom durch T2 und T4 wird nicht über eine Referenzstrom- oder Referenzspannungsquelle eingestellt, da dies zu einem zusätzlichen Verbrauch an Strom und Chipfläche führen würde. Der Strom durch T2 und T4 bestimmt über die beiden Stromspiegel und die H-Brücke den Betrag der sich einstellenden Gegentaktspannung.

Die Gleichtaktspannung am Ausgang wird bei diesem Leitungstreiber nicht geregelt. Sie liegt bei einer Betriebsspannung von 3,3 V und einer Temperatur von 27 °C im eingeschwungenen Zustand leicht oberhalb des nach der LVDS-Spezifikation zulässigen Bereichs.

Für On-Chip-Kommunikation ist dies unkritisch, da hier der Leitungsempfänger entsprechend ausgelegt werden kann. Eine Regelung oder Anpassung der Gleichtaktspannung würde zu einem zusätzlichen Verbrauch an Strom und Chipfläche führen.

Für Off-Chip-Kommunikation ist es notwendig, dass die Gleichtaktspannung die LVDS-Spezifikation erfüllt. Dies kann durch Anpassung der Schaltung leicht erreicht werden. Die Schaltung benötigt dann aber mehr Strom und Chipfläche.

Simulation

Die nachstehende Simulation wurde für eine Betriebsspannung von 3,3 V und eine Temperatur von 27 °C durchgeführt.

Der Ausgang wurde dabei mit dem Abschlusswiderstand von 100 Ω, mit parasitären Kapazitäten der Signalleitungen gegen Masse von jeweils 300 fF und mit einer Koppelkapazität zwischen den Signalleitungen von 30 fF belastet (siehe Abbildung 2).

Untersuchungen zu den parasitären Leitungskapazitäten des verwendeten Halbleiterprozesses werden zur Zeit im EDA-Zentrum der FH Aalen durchgeführt.

Die Eingangssignale (Daten und $\overline{\text{Daten}}$) haben in der Simulation folgende Eigenschaften:

- Logischer Wert 0 ≙ 0 V gegen Masse
- Logischer Wert 1 ≙ 3,3 V gegen Masse
- Datenrate: 5 GBit/s
- Anstiegs-/Abfallzeit: 40 ps

Abbildung 3 zeigt das Ergebnis der Simulation. Dargestellt sind die Spannungen von Daten, LVDS + und LVDS - gegen Masse sowie die Gleichtakt- und Gegentaktspannung am Ausgang.

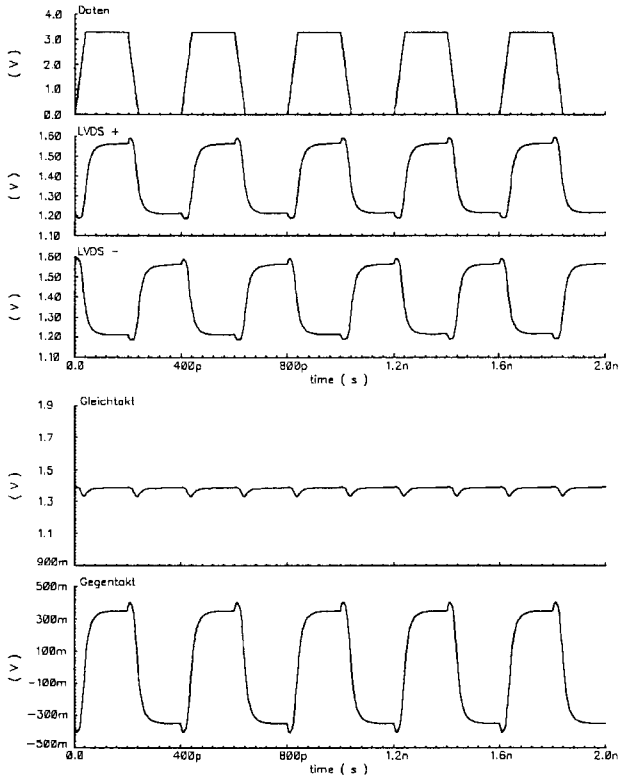


Abbildung 3 Simulationsergebnisse des H-Brücken-Leitungstreibers

Ergebnisse

Der Leitungstreiber hat einen geschätzten Flächenverbrauch von 1500 µm².

Bei einer Betriebsspannung von 3,0 V bis 3,6 V und einer Temperatur von 20 °C bis 60 °C erfüllt die Gegentaktspannung am Ausgang die LVDS-Spezifikation. Lässt man leichte Abweichungen der Gegentaktspannung von der LVDS-Spezifikation zu, so erhält man bei gleichem Betriebsspannungsbereich einen Temperaturbereich von z.B. 0 °C bis 90 °C.

Die Gleichtaktspannung am Ausgang erfüllt die LVDS-Spezifikation nicht für alle Betriebsspannungen von 3,0 V bis 3,6 V und Temperaturen von 20 °C bis 60 °C. Die Schaltung kann für Off-Chip-Kommunikation aber leicht so angepasst werden, dass die Gleichtaktspannung die LVDS-Spezifikation im gesamten Temperatur- und Betriebsspannungsbereich erfüllt.

Bei einer Betriebsspannung von 3,3 V und einer Temperatur von 27 °C ergeben sich folgende Werte:
Der Leitungstreiber besitzt eine Stromaufnahme von 3,7 mA. Die Gegentaktamplitude beträgt 350 mV. Beim Signalwechsel weist die Gegentaktspannung im Bereich von 10 % bis 90 % ihres Hubs[4] eine Anstiegsgeschwindigkeit von 13,8 V/ns auf. Es treten Gleichtaktschwankungen von 60 mV (Spitze – Spitze) auf.

Alternative Realisierung in Bipolar-Technik

Neben MOS-Transistoren stehen auch Bipolartransistoren (npn) im verwendeten Prozess zur Verfügung.

Eine Realisierung der H-Brückenschaltung unter Verwendung von Bipolartransistoren ist zwar prinzipell möglich, jedoch aus folgendem Grund sehr ungünstig:

Da die H-Brücken-Transistoren als Schalter arbeiten (siehe Abbildung 1), müssten die Bipolartransistoren in der Sättigung betrieben werden. Dies ist für sehr schnelle Schaltungen aufgrund der großen Ausschaltverzögerung nicht akzeptabel. Das betrachtete Konzept kann für die angestrebte Anwendung somit nur mit MOS-Transistoren realisiert werden.

2.2 Gesteuerte Spannungsquellen

Prinzip

Hier wird nicht wie bei der H-Brücke der Strom durch den Abschlusswiderstand eingeprägt, sondern es werden die Spannungen an LVDS + und LVDS – eingeprägt. Abbildung 4 veranschaulicht dieses Konzept.

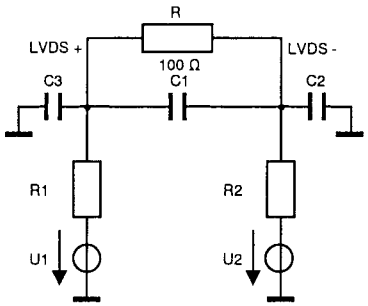


Abbildung 4 Gesteuerte Spannungsquellen

C1, C2 und C3 stellen wie in Abbildung 1 die parasitären Kapazitäten zwischen den beiden Signalleitungen bzw. zwischen jeweils einer Signalleitung und Masse dar. Es gilt: $C_3 \approx C_2$.

[4] Der Hub beträgt 700 mV, die Gegentaktspannung steigt von - 350 mV (≙0 %) auf + 350 mV (≙100 %) bzw. fällt von + 350 mV (≙ 100 %) auf - 350 mV (≙ 0 %).
10 % entsprechen - 280 mV; 90 % entsprechen + 280 mV.

U1 und U2 werden in Abhängigkeit des Eingangssignals eingestellt. R1 und R2 sind die Innenwiderstände der Spannungsquellen U1 bzw. U2 und sollten möglichst gering sein.

Mit genügend kleinen Innenwiderständen R1 und R2 erhält man bei gleichen Leitungskapazitäten eine geringere Zeitkonstante R·C als beim vorherigen Schaltungskonzept.

Es ist:

$$R = (R_1 + R_2) \| 100\ \Omega,\ C \approx C_1 + \frac{C_2}{2} (\text{da } C_2 \approx C_3)$$

Ist die maximale Taktfrequenz vorgegeben, mit der der Leitungstreiber betrieben werden soll, so kann man mit diesem Schaltungskonzept höhere Leitungskapazitäten und damit längere Leitungen treiben (da R geringer ist).

Umsetzung in eine Schaltung

Abbildung 5 zeigt eine mögliche Realisierung des Schaltungskonzepts. Den Spannungsquellen U1 und U2 in Abbildung 4 entspricht dabei in Abbildung 5 im Wesentlichen ein Stromschalter mit zwei nachgeschalteten Emitterfolgern.

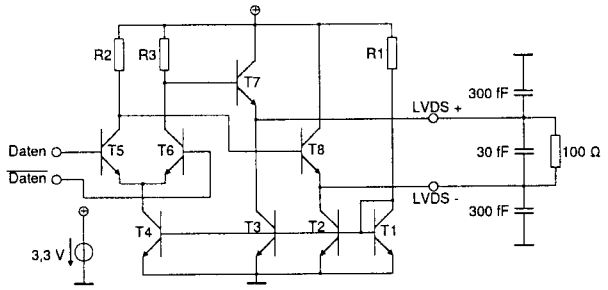


Abbildung 5 Realisierung mit Stromschalter und Emitterfolgern

Wie in Abbildung 2 sind Daten und $\overline{\text{Daten}}$ die Eingangssignale des Leitungstreibers. Das Ausgangssignal wird von LVDS + und LVDS – gebildet.

Die Widerstände haben folgende Werte:

R1:	7100	Ω
R2, R3:	120	Ω

Die Schaltung besteht aus einem Stromspiegel (T1 - T4) mit drei gesteuerten Zweigen, einem Stromschalter (T5/T6) und zwei Emitterfolgern (T7 bzw. T8).

Der Stromspiegel prägt den Strom durch den Stromschalter ein und stellt den Arbeitspunkt der Emitterfolger ein.

Der Strom in die Kollektoren von T2 und T3 beträgt jeweils 3,80 mA, in den Kollektor von T4 fließen 3,77 mA. Der Strom durch R1 beträgt 0,35 mA.

Die Spannungen U1 und U2 werden vom Stromschalter und den Emitterfolgern gebildet:

- Bei Daten[5] = 1 fließt in den Kollektor von T6 praktisch kein Strom. T5 ist im aktiven Bereich.

  Am Kollektor von T6 liegt eine Spannung von 3,29 V gegen Masse an, am Kollektor von T5 eine Spannung von 2,85 V gegen Masse.

  Die Basis-Emitter-Spannungen an T7 und T8 betragen 0,87 V bzw. 0,77 V.

  Es stellt sich somit eine Spannung von 2,42 V gegen Masse an LVDS + sowie von 2,08 V gegen Masse an LVDS - ein.

  Zwischen LVDS + und LVDS - ergibt sich eine Spannung von + 0,34 V.

- Bei Daten = 0 fließt in den Kollektor von T5 praktisch kein Strom. T6 ist im aktiven Bereich.

  Am Kollektor von T5 liegt eine Spannung von 3,29 V gegen Masse an, am Kollektor von T6 eine Spannung von 2,85 V gegen Masse.

  Die Basis-Emitter-Spannungen an T7 und T8 betragen 0,77 V bzw. 0,87 V.

  Es stellt sich somit eine Spannung von 2,08 V gegen Masse an LVDS + sowie von 2,42 V gegen Masse an LVDS - ein.

  Zwischen LVDS + und LVDS - ergibt sich eine Spannung von - 0,34 V.

Durch die Emitterfolger T7 und T8 erreicht man niedrige Ausgangswiderstände der Schaltung.

Der Ausgangswiderstand an LVDS + ist etwa umgekehrt proportional zum Strom in den Kollektor von T7; der Ausgangswiderstand an LVDS - ist etwa umgekehrt proportional zum Strom in den Kollektor von T8.

Man kann durch entsprechende Dimensionierung der Schaltung relativ einfach höhere Ströme in die Kollektoren von T7 und T8 erreichen. Dadurch erreicht man geringere Ausgangswiderstände und kann somit höhere Leitungskapazitäten bei gleicher Datenrate treiben.

Vorstehende Schaltungsbeschreibung geht von einer Betriebsspannung von 3,3 V und einer Temperatur von 27 °C aus.

Bei anderen Betriebsspannungen oder anderen Temperaturen stellen sich in der Simulation andere

[5] Daten, $\overline{\text{Daten}}$: Logischer Wert 0 ≙ 1,5 V gegen Masse
Logischer Wert 1 ≙ 2 V gegen Masse

Ströme und Spannungen ein; die Gegentaktspannung am Ausgang erfüllt jedoch für Betriebsspannungen von 3,0 V bis 3,6 V und Temperaturen von - 20 °C bis + 60 °C die LVDS-Spezifikation.

Lässt man Abweichungen der Gegentaktspannung von der LVDS-Spezifikation von maximal 10 mV zu, so erhält man bei diesem Leitungstreiber bei gleichem Spannungsbereich einen Temperaturbereich von - 40 °C bis + 90 °C. Erlaubt man Abweichungen von maximal 15 mV, so erhält man bei gleichem Spannungsbereich sogar einen Temperaturbereich von – 50 °C bis + 100 °C.

Durch eine andere Dimensionierung der Transistoren und Widerstände lassen sich auch andere Betriebsspannungs- oder Temperaturbereiche einstellen, in denen die Gegentaktspannung am Ausgang der LVDS-Spezifikation genügt oder von dieser nur leicht abweicht.

Die Gleichtaktspannung am Ausgang liegt bei einer Betriebsspannung von 3,3 V und einer Temperatur von 27 °C im eingeschwungenen Zustand etwa 0,87 V oberhalb des nach der LVDS-Spezifikation zulässigen Bereichs. Sie kann für Off-Chip-Kommunikation leicht an die LVDS-Spezifikation angepasst werden. Dies bedeutet aber auch hier einen zusätzlichen Verbrauch an Strom und Chipfläche.

Simulation

Die Simulation erfolgt für dieselbe Betriebsspannung und Temperatur und mit derselben Ausgangslast wie bei der H-Brücken-Schaltung.

Die Eingangssignale (Daten und $\overline{\text{Daten}}$) haben in der Simulation folgende Eigenschaften:

- Logischer Wert 0 ≙ 1,5 V gegen Masse
- Logischer Wert 1 ≙ 2 V gegen Masse
- Datenrate: 5 GBit/s
- Anstiegs-/Abfallzeit: 40 ps

Für die logischen Werte 0 und 1 werden hier andere Spannungen gegen Masse als beim H-Brücken-Leitungstreiber verwendet, um die Transistoren des Stromschalters (T5 und T6) nicht zu übersteuern.

Abbildung 6 zeigt das Ergebnis der Simulation. Dargestellt sind wie in Abbildung 3 die Spannungen von Daten, LVDS + und LVDS - gegen Masse sowie die Gleichtakt- und Gegentaktspannung am Ausgang.

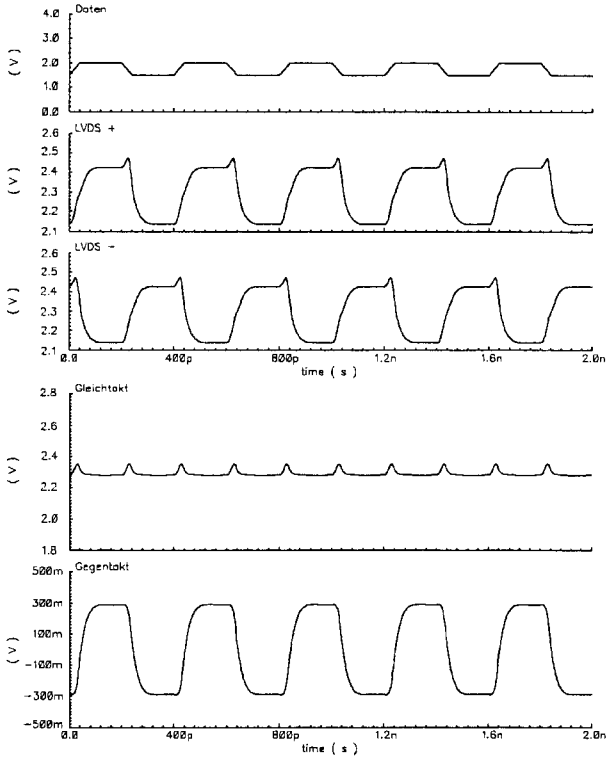


Abbildung 6 Simulationsergebnisse des Leitungstreibers mit Stromschalter und Emitterfolgern

Ergebnisse

Der Leitungstreiber hat einen Flächenverbrauch von etwa 8600 µm². Der Flächenverbrauch lässt sich ohne nennenswerte Geschwindigkeitseinbußen durch Verwendung kleinerer Transistoren auf etwa 5250 µm² verringern.

Bei einer Betriebsspannung von 3,0 V bis 3,6 V und einer Temperatur von -20 °C bis + 60 °C erfüllt die Gegentaktspannung am Ausgang die LVDS-Spezifikation. Lässt man leichte Abweichungen der Gegentaktspannung von der LVDS-Spezifikation zu, so erhält man bei gleichem Betriebsspannungsbereich einen Temperaturbereich von z.B. – 50 °C bis + 100 °C.

Die Gleichtaktspannung am Ausgang erfüllt die LVDS-Spezifikation bei einer Betriebsspannung von 3,0 V bis 3,6 V und einer Temperatur von – 20 °C bis + 60 °C nicht. Die Schaltung kann aber für Off-Chip-Kommunikation leicht so angepasst werden, dass die Gleichtaktspannung die LVDS-Spezifikation erfüllt.

Bei einer Betriebsspannung von 3,3 V und einer Temperatur von 27 °C ergeben sich folgende Werte: Der Leitungstreiber hat eine Stromaufnahme von 11,7 mA. Die Gegentaktamplitude beträgt ungefähr 340 mV. Beim Signalwechsel weist die Gegentakt-

spannung im Bereich von 10 % bis 90 % ihres Hubs[6] eine Anstiegsgeschwindigkeit von 11,5 V/ns auf. Es treten Gleichtaktschwankungen von 76 mV (Spitze – Spitze) auf.

Die Anstiegsgeschwindigkeit der Gegentaktspannung beim Signalwechsel ist etwas geringer als beim H-Brücken-Leitungstreiber. Vergrößert man bei der Betriebsspannung von 3,3 V und der Temperatur von 27 °C durch eine andere Dimensionierung der Schaltung den Strom in die Kollektoren von T2 und T3 jedoch auf etwa 7,8 mA, so erhält man eine größere Anstiegsgeschwindigkeit von 18,4 V/ns. Die Schaltung kann also ihren Vorteil nur ausspielen, wenn die Ströme genügend groß sind. Durch weitere Stromerhöhungen lassen sich auch noch größere Anstiegsgeschwindigkeiten einstellen.

Alternative Realisierung in MOS-Technik

Prinzipiell ist es denkbar, dieses Schaltungskonzept auch in MOS-Technik zu realisieren. Dies wurde im Rahmen der hier beschriebenen Arbeit jedoch nicht durchgeführt.

Bipolartransistoren haben gegenüber MOS-Transistoren den Vorteil, dass ihre Steilheit proportional zum Kollektorstrom ist, während die Steilheit der MOS-Transistoren proportional zur Wurzel des Betrags des Drainstroms ist.

Der Ausgangsleitwert eines Emitterfolgers oder Sourcefolgers entspricht wiederum in etwa der Steilheit des eingesetzten Transistors.

Man benötigt also ab einem bestimmten Punkt beim Bipolartransistor geringere Ströme als beim MOS-Transistor, um denselben Ausgangsleitwert zu erreichen.

3 Ergebnis

In der hier beschriebenen Arbeit wurden zwei Schaltungskonzepte für einen LVDS-Leitungstreiber mit dem Schwerpunkt der On-Chip-Kommunikation untersucht.

Dabei stellte sich heraus, dass für kleine Leitungskapazitäten oder geringe Datenraten der mit MOS-Transistoren realisierte H-Brücken-Leitungstreiber besser geeignet ist, weil er eine geringere Stromaufnahme und einen geringeren Flächenverbrauch aufweist.

Hat man Daten mit hoher Taktfrequenz über Leitungen mit großen Kapazitäten zu übertragen, so muss der mit Bipolartransistoren realisierte Leitungstreiber mit Stromschalter und Emitterfolgern verwendet werden. Dies ist besonders beim Entwurf von LVDS-Treibern für Off-Chip-Kommunikation zu beachten.

Für den Leitungstreiber mit Stromschalter und Emitterfolgern wurde ein Layout erstellt. Dieses soll als Ausgangspunkt zur Entwicklung einer LVDS-Zellen-Bibliothek verwendet werden.

4 Literaturverzeichnis

[1] http://www.ihp-ffo.de

[2] IEEE Journal of Solid-State Circuits, Vol. 36, No. 4, April 2001

[6] 10% entsprechen hier – 272 mV; 90% entsprechen + 272 mV.

FPGA Implementierung eines Frame Buffer Controllers

Dipl.-Ing. Josef Hahn-Dambacher, Dipl.-Ing. Christian Kirschenlohr, Prof. Dr. Manfred Bartel

Fachhochschule Aalen, EDA-Zentrum, Beethovenstrasse 1, 73430 Aalen

Tel. 07361 / 576-247, Fax. 07361 /576-324

jhahn@rzws.fh-aalen.de, christian@kirsche.org, manfred.bartel@fh-aalen.de

Im Rahmen einer Diplomarbeit und einer parallel laufenden Projektarbeit wurde das Konzept und die Implementierung eines FPGA basierten Frame Buffer Controllers (FBC) entwickelt.

Die vorgegebene FBC-Spezifikation wurde prototypisch auf einem ALTERA NIOS Developmentboard, das mit einem FPGA Cyclone 1C20 bestückt war, implementiert und getestet. Die Vorgaben konnten annähernd erfüllt werden, womit einer vollständigen Implementierung nichts mehr im Wege steht.

1 Einführung

1.1 Wofür wird ein Frame Buffer Controller benötigt?

Frame-Buffer-Controller finden heutzutage überall dort Verwendung, wo es darum geht digitale Daten auf einem Bildausgabegerät wie zum Beispiel Monitore, Beamer, LED-Leinwände usw. auszugeben.

Es wird hierbei zwischen zwei Arten unterschieden:

1. die z.B. in einem PC zum Einsatz kommen und Bestandteil einer Graphikkarte sind und
2. die zur Umkonvertierung von Bilddatenströmen gedacht sind. Diese Konvertierung dient dazu, den Bilddatenstrom auf ein spezielles Ausgabegerät anzupassen.

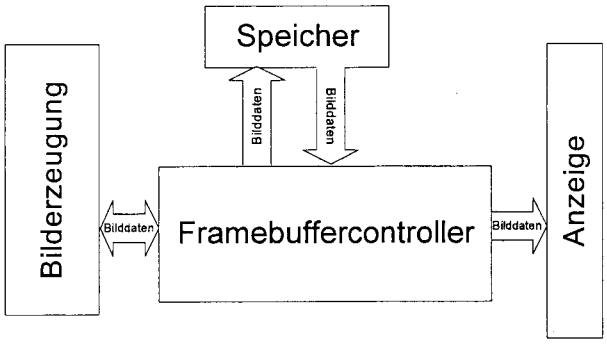
Der Funktionsumfang von käuflich erhältlichen Controllern reicht von einer simplen Bildwiederholratenkonvertierung bis hin zur komplexen Auflösungsanpassung dem so genannten "Scaling".

1.2 Was ist ein Frame Buffer Controller?

Unter einem Frame-Buffer versteht man einen Zwischenspeicher in dem digitale Bilddaten abgelegt werden. Der Zugriff auf diese Bilddaten erfolgt über ein Steuergerät, den so genannten Frame-Buffer-Controller (im weiteren FBC genannt).

Dieser FBC hat folgende grundlegende Aufgaben:

1. Ablegen der Bildinformation im Speicher
2. Abrufen der Bildinformation aus dem Speicher
3. Ausgeben der Bilddaten mit einem für das Display passenden Timing



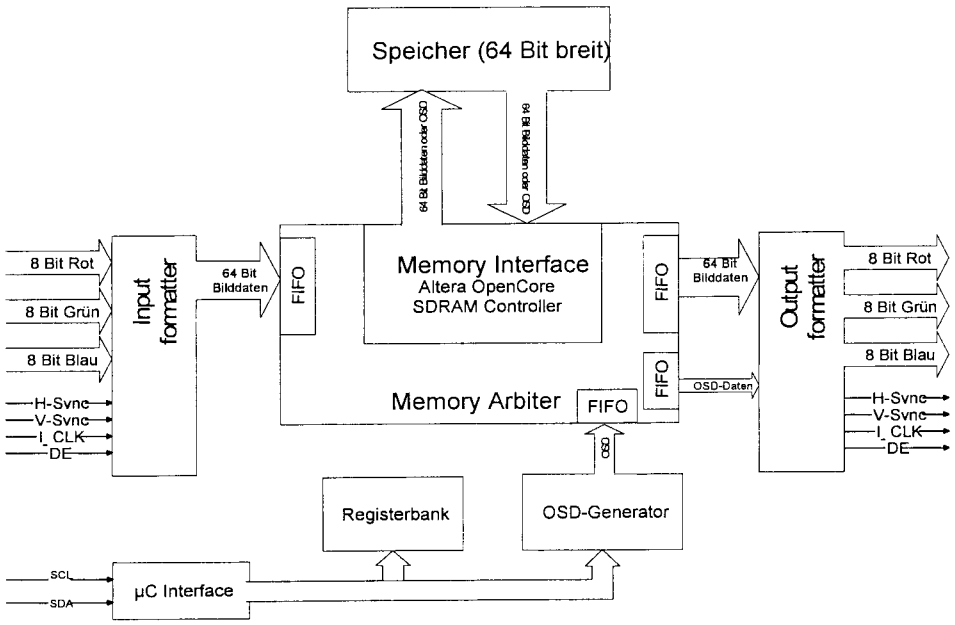
Im Speziellen können bestimmte FBC auch noch folgende Funktion übernehmen:

1. Scaling: Das bedeutet das Anpassen der Eingangsauflösung auf die Ausgangsauflösung
2. OSD (On Screen Display) -Funktionalität: Darstellung von Statusinformation auf dem aktuellen Bildinhalt

1.3 Aufbau eines FBC

Ein FBC lässt sich in Datenflussrichtung, vom Eingang zum Ausgang, in folgende Komponenten zerlegen:

1. Inputformatter
   Der Inputformatter entnimmt die relevanten Bilddaten aus dem Eingangsbild und passt diese der Speicherbreite an.
2. Scaler (optional)
   Anpassung des Eingangs- auf das Ausgansbild, d.h. Vergrößerung oder Verkleinerung des Eingangsbilds.



3. Memory Arbiter und Memory Controller
   Diese Einheit sorgt dafür, dass alle Funktionsblöcke, entsprechend ihrer Priorität, Zugriff auf den Speicher erhalten und steuert den Zugriff.
4. OSD Generator (optional)
   Erzeugt ein On Screen Display, welches über das Ursprungsbild projiziert und ausgegeben wird.
5. Outputformatter mit Display Timing Generator (DTG)
   Erzeugt die Synchronisationssignale und generiert aus den Bild- und OSD-Daten das vollständige Ausgangsvideobild.

1.4 Warum eine Eigenentwicklung?

Selbst die "einfachen" FBC sind heutzutage so leistungsstark, dass sie für viele Anwendungen überdimensioniert sind. Aus folgenden Gründen kann es Sinn machen, einen FBC selbst zu entwickeln:

1. Kein auf dem Markt verfügbarer Schaltkreis ist in der Lage die gewünschten Leistungsmerkmale zu einem vernünftigen Preis zu bieten.
2. Durch die rasante Entwicklung auf dem Markt der Grafik- und Frame-Buffer-Controller ist man gezwungen bewährte Platinenlayouts zu verwerfen, da der verwendete Chip nicht mehr lieferbar ist.

Um aus der Abhängigkeit zu einzelnen Halbleiterproduzenten zu kommen, wurde dieser FBC entwickelt.

2 Der Frame Buffer Controller

2.1 Spezifikation

Der zu entwickelnde FBC soll in einem so genannten KVM Extender eingesetzt werden. KVM steht für Keyboard, Video und Maus. Dabei wird der Arbeitsplatz z.B. nur über eine Glasfaser an einen PC oder Server angebunden. Dies wird oft bei der Serverwartung, aber auch in kritischen Bereichen (EMV, Lärm) eingesetzt.

Folgende technische Spezifikation soll der zu entwickelte FBC erfüllen:

- Maximal UXGA Auflösung
  - d.h. 1600*1200 sichtbare Bildpunkte
  - bzw. 2160*1250 Gesamtbildpunkte
  - bei 60 Hz Bildwiederholrate
- 24 Bit RGB Farbtiefe
- Keine Skalierfunktionalität
- Crop und Pan Funktionalität
- Voll parametrierbar über ein µC Interface
- OSD Generator
  - 640*480 Bildpunkte
  - 16 Farben (Farbpalette 32 Bit, zusätzlicher Alpha Kanal)
- Datenaustausch über ein µC Interface

2.2 Vorüberlegungen

Arbeitsfrequenz:

Für die Auswahl des benötigen FPGA's ist unter anderem die maximale Arbeitsfrequenz der Eingangs- und Ausgangsstufen wichtig. Diese hängt direkt von dem maximalen Pixeltakt ab, welcher aus den Daten der Spezifikation berechnet werden kann.

Pixeltakt: 2160*1250*60 Hz = 162 MHz

Speicher:

Der Speicher muss ausreichend groß sein um mindestens zwei komplette Bilder plus OSD Bild speichern zu können.

1 Bild:	1600*1200*24 Bit	≈ 44 Mbit
OSD:	640*480*4 Bit	≈ 1,2 Mbit
Gesamt:	44 + 44 + 1,2 Mbit	≈ 89,2 Mbit

Es wird mindestens ein Speicher mit 11,1 MByte Größe benötigt.

Wichtig für die Auswahl des Speichers ist die so genannte Speicherbandbreite, d.h. wie viele Daten pro Zeit gespeichert, bzw. gelesen werden. Dazu wird die Berechnung im kritischen Bereich, d.h. alle Einheiten liefern oder fordern Daten an, durchgeführt.

1 Bild:	162 MHz * 24 Bit	≈ 3,9 GBit / s
OSD:	1,2 Mbit * 60 Hz	≈ 70 MBit / s
Gesamt:	3,9 + 3,9 + 0,07 GBit / s	≈ 7,9 GBit / s

2.3 Bausteinauswahl

2.3.1 Speicher

Um das unter Kosten-/Nutzenaspekten günstigste Speicherinterface zu ermitteln wurden Technologien wie SDRAM bzw. DDR-SDRAM unter folgenden Gesichtspunkten betrachtet.

- Implementierungsaufwand
- Kosten
- Verfügbarkeit der Speicherbausteine

Dabei fiel die Entscheidung auf ein 64 Bit breites SDRAM Speicherinterface mit einem Speichertakt von 166 MHz. Dabei werden zwei 64 MBit (16 MByte) große SDRAM Speicher in einer so genannten 2M x 32 Konfiguration verwendet.

Gründe:

1. Mit deutlich geringerem Logikaufwand ist ein SDRAM Interface im Vergleich zu einem DDR-RAM Interface, zu implementieren. Laut Altera Datenblatt liegt das Verhältnis SDRAM 64 Bit / DDR-RAM 32 Bit bei 180/1000 LE´s
2. Die Verfügbarkeit eines kostenlosen und quelloffenen IP-Core von Altera.
3. Die Verfügbarkeit von Standardspeicherbausteinen in der benötigten Konfiguration. Ein weiteres Problem stellt sich dadurch, dass die Speicherbausteine nicht in der benötigten, relativ kleinen Kapazität, in jeder Bauform verfügbar sind.

Die gewählte Konfiguration hat aber auch eine Reihe von Nachteilen die nicht verschwiegen werden sollen:

1. Es ist eine relativ aufwändige Umkonvertierung der Bilddaten von 24 Bit auf 64 Bit notwendig. Bitbreiten von vielfachen von 24, wie z.B. 48, 72, 96 wären für die Implementierung einfacher.
2. Großer I/O Count des verwendeten FPGA's wirkt sich negativ auf den Preis des FPGA's aus.
3. Durch die Verwendung von 2 Speicherbausteinen erhöht sich der Flächenbedarf auf der Platine. Die Folge ist eine Erhöhung der Produktionskosten.

Der SDRAM wird im so genannten Full-Page-Mode betrieben um die Datenrate zu maximieren. Hierbei treten folgende Verluste auf:

1. Overhead von 8 % bei einer Page Breite von 256 Worten bei Lese- und Schreibzugriffen
2. Refresh Zyklen bewirken einen Verlust von 4%

Daraus ergibt sich nun ein Speicherbandbreitenbedarf von:

7,9 GBit / s * 1,12 ≈ 9 GBit / s

Für den ausgewählte SDRAM mit einem Speichertakt von 166 MHz ergibt sich eine maximale Speicherbandbreite von 10,62 GBit/ s.

Somit ergibt sich eine Bandbreitenreserve von 1,62 GBit/s bzw. 15 %.

2.3.2 FPGA

Für die Auswahl des FPGA wurden folgende Randbedingungen betrachtet:

- Die Taktfrequenz der implementierten Schaltwerke muss mindestens den Ansprüchen des Videotaktes bzw. Speichertaktes genügen.
- On-Chip-Memory für FIFOs
- IP Core für das SDRAM Interface soll verfügbar sein
- PLLs für SDRAM Interface und Outputformatter
- ca. 5000 Logikelemente
- Eventuell ein Soft Core Prozessor verfügbar
- Kosten von ca. 50 Euro pro Einheit (FPGA + SDRAM)
- ca. 150-160 I/Os
- Für eine spätere Serienfertigung soll eine HardCopy Option (FPGA zu ASIC) verfügbar sein.
- Verfügbarkeit eines Developmentboards

Die Auswahl für dieses Projekt fiel auf den Cyclone 1C6 der Firma Altera.

2.4 Implementierung

2.4.1 Inputformatter

Der Inputformatter bestimmt die relevanten Bildpunkte im Eingangsvideobild mittels der Synchronisationssignalen V-Sync und H-Sync bzw. DE (Display Enable). Die Bildpunkte mit einem 24 Bit Farbwert müssen nun an das 64 Bit Speicherinterface angepasst werden. Dazu werden jeweils 8 Bildpunkte zu einem 3 mal 8 Byte Paket zusammengefasst, welches an den Memory Arbiter / Controller weitergereicht wird. In nachfolgender Tabelle ist ein Packet dargestellt, in den Zellen ist die Platzierung der RGB Farbwerte der 8 Bildpunkte eingetragen.

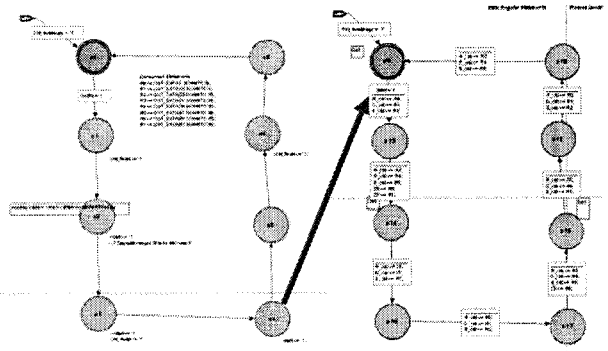
	64 Bit / 8 Byte Speicherbreite							
	0	1	2	3	4	5	6	7
0	R0	G0	B0	R1	G1	B1	R2	G2
1	R3	G3	B3	R4	G4	B4	R5	B2
2	R6	G6	B6	R7	G7	B7	G5	B5

2.4.2 Outputformatter

Der Outputformatter erzeugt nach einstellbaren Parametern das Timing und die Synchronisationssignale des Ausgangsvideobildes. Die vom Memory Arbiter angeforderten Bildsignale werden wieder in das 24 Bit RGB Format zurückkonvertiert und dem Videosignal zugeführt. Zusätzlich werden OSD Daten, wenn dieses dargestellt werden sollen, angefordert und den Bilddaten überlagert. Die Bilddaten des OSD sind 32 Bit breit implementiert, d.h. sie bestehen aus einem 24 Bit RGB Farbanteil und einem zusätzlichen 8 Bit Alpha Kanal, mit dessen Hilfe teiltransparente Farben definiert werden können.

Der Outputformatter wurde in einer Pipeline Architektur aufgebaut, da einerseits eine zeitliche Verzögerung von 3 Taktperioden zwischen Arbiter und Outputformatter und zusätzlich noch 1 Taktperiode für die OSD Überlagerung bei der Teiltransparenzberechnung benötigt wird.

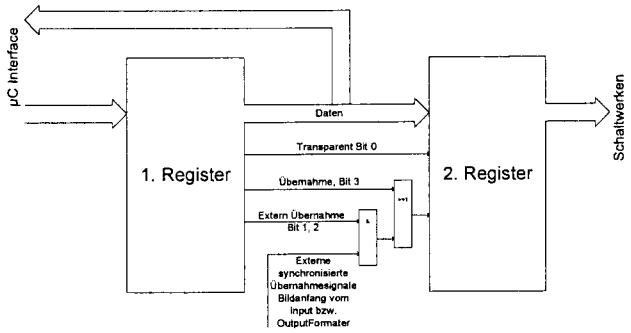
In nachfolgender Abbildung sind exemplarisch zwei Automaten für die Pipelinesteuerung dargestellt, wobei der erste Automat den zweiten steuert. Beide Automaten bearbeiten pro Durchgang jeweils 8 Bildpunkte, d.h. jeweils ein komplettes Packet.



2.4.3 Registerbank

Die verschiedenen Einheiten des FBC sollen über ein Mikrocontroller Interface parametrierbar sein. Die Parameter werden in einer Registerbank gespeichert. Da jedoch immer ein kompletter Parametersatz in die Registerbank geladen werden soll, bevor dieser aktiviert wird, um z. B. ungültige Zustände zu vermeiden oder nicht VESA konforme Videosignale zu erzeugen, wurden zwei identische Registerbänke implementiert. Auf die erste Registerbank, dem sogenannten Shadow-Register, wird über das Interface zugegriffen. Erst nachdem alle Parameter aktualisiert wurden, werden diese an die zweite Registerbank weitergereicht. Dies kann auch noch

zusätzlich mit dem Bildanfang des Inputformatters bzw. des Outputformatters synchronisiert werden um Bildstörungen zu vermeiden.



2.4.4 Memory Interface

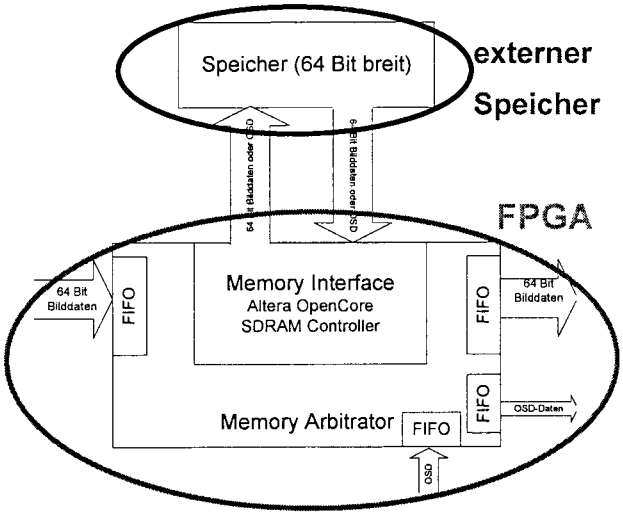
Für das Memory Interface standen folgende drei Alternativen zur Verfügung:

1. Eigentwicklung
   Nachteilig ist hierbei der sehr große Arbeitsaufwand, da viele verschiedene SDRAM Timings unterschiedlichster Hersteller beachtet werden müssen. Außerdem muss das Interface auf der Hardware validiert werden.
2. Kauf eines IP Cores
   Diese käuflichen Cores sind vom Hersteller validiert und müssen normalerweise nur noch für den jeweiligen SDRAM Baustein und Betriebsmodi parametriert werden. Sie bieten eine komfortable Schnittstelle für den Datenaustausch mit dem SDRAM und somit ist der Implementierungsaufwand als relativ niedrig anzusehen.
3. Altera OpenCore SDRAM Controller
   Dies ist ein freier und im Quellcode vorliegender IP Core. Dieser muss bei der Implementierung parametriert werden und bietet eine transparente Schnittstelle zum SDRAM. D.h. grundlegende Operationen zum Datenaustausch müssen von einer höheren Ebene erledigt werden. Dies äußert sich in einem etwas höheren, aber vertretbaren Arbeitsaufwand.

2.4.5 Memory Arbiter

Da kein gleichzeitiger Zugriff der konkurrierenden Schaltwerke auf die Ressource Speicher möglich ist, muss der Arbiter zu diesen Schaltwerken durch FIFOs gekapselt werden die genügende Kapazität/ Elastizität haben um Sperrzeiten beim Speicherzugriff zu überwinden.

Der Arbiter muss dabei in der Lage sein ein bevorstehendes Leerlaufen, bzw. Vollaufen der FIFOs rechtzeitig zu erkennen und den entsprechend lesenden bzw. schreibenden Speicherzugriff auszuführen. Dadurch sorgt der Arbiter dafür, dass es für die Ein- und Ausgänge so aussieht als ob sie permanent exklusiven Zugriff auf den Speicher haben.



Aus dem Anforderungsprofil ergibt sich ein weiteres Problem. Da sich die Taktraten sowohl am Eingang als auch am Ausgang je nach Videogröße und Wiederholrate in weiten Bereichen ändern können, müssen die Schaltwerke untereinander synchronisiert werden. Man spricht hierbei vom so genannten Clock Domain Crossing. Da auf Grund der Systemarchitektur sowieso FIFOs verwendet werden müssen, können diese FIFOs auch zum Synchronisieren der Clock Domains verwendet werden. Dabei müssen die FIFOs aber als so genannte Dual-Clocked FIFOs ausgeführt werden, bei denen der Lesetakt sich vom Schreibtakt unterscheidet.

3 Ergebnis und Ausblick

Der FBC wurde auf einem NIOS Developmentboard von Altera mit dem FPGA Cyclone 1C20 aufgebaut und getestet. Wobei bezüglich des Speichers (32 Bit Breite und 133 MHz) Kompromisse eingegangen werden mussten.

Nachfolgend sind die derzeitigen Synthese und Simulationsergebnisse aufgeführt:

1. Nach Place&Route Angaben fmax. 148 MHz
2. Design mit 125 MHz getestet
3. Schaltung mit 800*600 Bildpunkten bei 60 bis 85 Hz getestet
4. ca. 3000 LE's wurden benötigt

Wobei beachtet werden muss, dass der spezifizierte Baustein Cyclone 1C6 in einem höheren Speedgrade sowie in einem kleineren Gehäuse verwendet wird. Die Syntheseergebnisse kommen hierbei auf eine maximale Taktfrequenz der Schaltung knapp unter den geforderten 162 MHz.

Nach den viel versprechenden Ergebnissen wird nun der FBC weiterentwickelt. Insbesondere wird, zur Zeit, ein Board entwickelt, welche die spezifizierten Bauelemente sowie die gesamte weitere Hardware des KVM Extender Zielsystems beinhaltet.

Bei den, die Taktfrequenz limitierenden, In- und Outputformatter soll weiterhin untersucht werden, ob sich mit weiteren Pipe-Line-Architekturen eine Erhöhung der maximalen Arbeitsfrequenz erreichen lässt.

Realisierung einer Direct Digital Synthesis (DDS) auf einem FPGA

Stefan Widmann
FH Ulm, Eberhard-Fink-Strasse 11
Telefon: 0731 / 50 – 28338 (Prof. Führer)

Ziel der Diplomarbeit war es, ein System zur digitalen Frequenzsynthese auf einem FPGA zu implementieren welches Ausgangsfrequenzen zwischen 0 und 140 MHz erzeugt.

Das Wechseln der Frequenz sollte schnell und ohne Phasensprung möglich sein.

Basis der Implementierung war das Virtex-II Pro FPGA der Firma Xilinx, untergebracht auf dem Evaluationboard aus dem Hause Memec.

1. Aufgaben und Motivation

1.1. Aufgabenstellung

Es galt während dieser Diplomarbeit ein System zur digitalen Frequenzsynthese auf einem FPGA zu realisieren, in diesem Fall auf dem Virtex-II Pro FPGA von Xilinx.

Das System sollte in der Lage sein, Ausgangsfrequenzen zwischen 0 und 140 MHz zu erzeugen, und der Wechsel der Frequenz sollte schnell und ohne Phasensprung möglich sein.

Aus den drei zur Verfügung stehenden Digital Analog Convertern (DAC) war der für die Arbeit am besten geeignete auszuwählen und die physikalische Verbindung zur Übertragung der Daten zwischen FPGA und DAC herzustellen.

Das Design des Systems auf dem FPGA sollte in VHDL unter Verwendung des Softwarepakets ISE von Xilinx erfolgen.

Zum Abschluss sollte durch Messung des Phasenrauschens die Eignung des entstandenen Systems für Hochfrequenzanwendungen verifiziert werden.

1.2. Motivation

Motivation der Arbeit war das sogenannte Software Defined Radio (SDR). Ziel ist es dabei, möglichst grosse Teile der Signalerzeugung in den digitalen Bereich des Systems zu verlagern.

Dadurch ist es möglich, Problemen wie Toleranzen und Alterung der Bauteile im Analogteil aus dem Weg zu gehen.

Es wird auch möglich, einheitliche Hardwareplattformen für unterschiedliche Einsatzzwecke zu entwerfen, bei denen sich dann nur die Software zur Signalerzeugung unterscheidet.

Eine Änderung der Signalerzeugung muss dann auch nicht mehr einen Austausch des Systems bedeuten, sondern im Idealfall ist lediglich ein Software-Update nötig. FPGAs eigenen sich besonders wegen der Möglichkeit der Reprogrammierung für Software Defined Radio Systeme.

2. Grundlagen der DDS

Bei der Direct Digital Synthesis handelt es sich um ein Verfahren zur digitalen Erzeugung sinusähnlicher Signale.

Ein DDS System besteht dabei im Wesentlichen aus zwei Komponenten: dem Phasenakkumulator und dem Phasen-Amplituden-Wandler.

2.1. Phasenakkumulator

Der Phasenakkumulator ist ein Ringzähler mit variabler Schrittweite, der mit einem festen Takt gespeist wird. Die Schrittweite wird durch das so genannte Tuning Word definiert.

Jeder Zählerstand des Akkumulators entspricht einem bestimmten Phasenwinkel, welcher an den Phasen-Amplituden-Wandler weitergegeben wird.

2.2. Phasen-Amplituden-Wandler

Der Phasen-Amplituden-Wandler setzt den vom Phasenakkumulator gelieferten Phasenwinkel in die zugehörige Amplitude für einen oder mehrere Kanäle um. Dazu werden meist Look-Up-Tabellen verwendet, in denen die Amplitudenwerte zu den Winkeln fest gespeichert sind. Der Phasenwinkel dient dann als Index innerhalb der Tabellen.

2.3. Ausgangsfrequenz

Von der Taktfrequenz f_{Takt} der DDS und der Bitbreite n des Phasenakkumulators ist die Auflösung der Frequenz des Systems abhängig. Der kleinste mögliche Frequenzschritt Δf berechnet sich dabei wie folgt:

$$\Delta f = \frac{f_{Takt}}{2^n} \qquad (2.1)$$

Mit dem Tuning Word x_t wird die Ausgangsfrequenz eingestellt. Die Frequenz des Nutzsignals f_n ergibt sich nach Formel 2.2:

$$f_n = \frac{x_t * f_{Takt}}{2^n} \qquad (2.2)$$

Stellt man diese Formel um, so erhält man das für eine bestimmte Ausgangsfrequenz einzustellende Tuning Word x_t:

$$x_t = 2^n \frac{f_n}{f_{Takt}} \qquad (2.3)$$

3. Lösung der Aufgabenstellung

3.1. Auswahl des DAC

Zur Einbindung in das entstehende System standen 3 Digital Analog Converter zur Auswahl:

- AD9786 von Analog Devices
- AD9726 von Analog Devices
- MB86064 von Fujitsu

Die Wahl fiel auf den AD9786, da das Evaluationboard einfach zu kontaktieren ist und der Baustein Interpolations- und Modulationsstufen bietet.

3.2. Implementierung der DDS

Zur Realisierung der Aufgabenstellung wurde der DDS-IP-Core von Xilinx ausgewählt. Der fertige und getestete Funktionsblock enthält zusätzlich zu den im zweiten Absatz erwähnten Grundkomponenten eines DDS-Systems Methoden zur Rauschoptimierung (Phase-Dithering und Taylor-Näherungen).

Die Einbettung in das eigene Design ist schnell und einfach möglich. Nach der Parametrisierung des Cores im sogenannten Core-Generator wird ein Blocksymbol erzeugt, welches im Schematic Editor in das Design eingefügt werden kann.

Im Bild 1 zu sehen ist das beschriebene Blocksymbol. Auf der linken Seite befinden sich der Takteingang und die Dateneingänge zur Einstellung von Tuning Word und Phasenoffset.

Auf der rechten Seite des Symbols sind die Ausgänge für das komplexe Signal zu sehen.

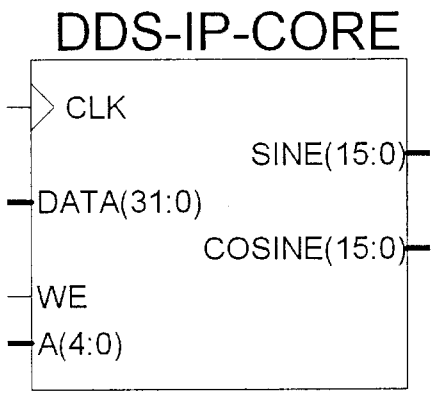


Bild 3.1: Blocksymbol des DDS-Cores

4. Ergebnis

Getaktet wird die DDS mit 100 MHz, die höheren Ausgangsfrequenzen bis zu 149 MHz lassen sich durch Hochmischen des Nutzsignals mit Hilfe der Modulations- und Interpolationsstufen des AD9786 gewinnen.

4.1. Ausgangssignal im Zeitbereich

In Bild 4.1 ist das Ausgangssignal bei einer Frequenz f_n von 6,25 MHz zu sehen.

Schön zu erkennen sind sinusähnliche Form und Quantisierungsstufen im Signalverlauf.

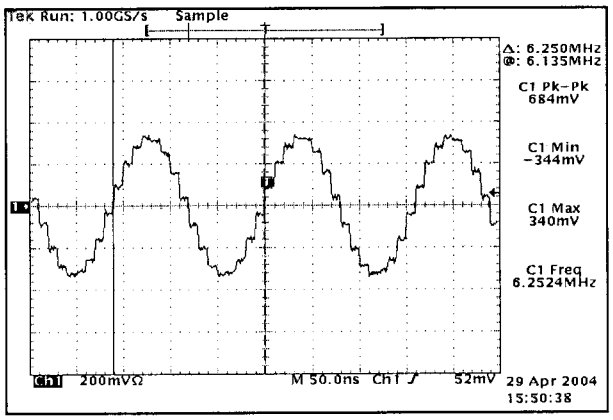


Bild 4.1: Oszillogramm

4.2. Spektrum des Ausgangssignals

Das Spektrum des Ausgangssignals bei einer Frequenz von 140 MHz zeigt Bild 4.2.

Neben dem Nutzsignal (in der Mitte) mit einer Ausgangsleistung von fast 0 dBm sind auch Nebenaussendungen, die so genannten Spurs oder Spurious zu erkennen. Diese liegen im Bild unterhalb von -65 dBm. Der Pegelabstand zwischen Haupt- und Nebenaussendungen sollte möglichst groß sein. Mindestens 30 bis 40 dB sind notwendig, um das Phasenrauschen (siehe 4.3.) messen zu können, da sonst die PLL des Phase-Noise-Interfaces nicht auf die eigentliche Nutzfrequenz aufsynchronisieren kann.

Da, wie im Bild zu sehen, der Abstand über 60 dB betrug, war eine Messung des Phasenrauschens möglich.

4.3. Phasenrauschen

Bild 4.3 zeigt die Messergebnisse bei einer Ausgangsfrequenz der DDS von 125 MHz.

Vier für verschiedene Anwendungsfälle wichtige Pegelgrenzen sind im Bild markiert.

Die erste Linie (1) bei -70 dBc / Hz markiert den Pegel, oberhalb dem keine Störungen für Mobilfunkanwendungen zulässig sind. Besser ist es, wenn es weniger als -80 dBc / Hz (Linie 2) sind. Als ideal sind Störungspegel kleiner als -90 dBc / Hz anzusehen (Linie 3). Wie im Bild zu erkennen, liegen alle Störungen noch unterhalb der dritten Linie, bis auf eine harmonische Aussendung 25 MHz vom Trägersignal entfernt. Wichtig ist vor allem, dass der Noise Floor möglichst tief liegt, da Nutzsignale, deren Pegel darunter liegen nicht auszuwerten sind und somit verloren gehen.

Das System ist also gut für Mobilfunkanwendungen geeignet.

Die letzte Linie (Linie 4) bei -130 dBc / Hz kennzeichnet den Pegel, ab dem eine sinnvolle Signalauswertung für Dopplerradaranwendungen möglich ist.

Wie im Bild zu sehen liegen viele Störungen deutlich oberhalb der vierten Linie, es sind zum Teil unvermeidbare Störungen durch äußere Einflüsse (Beleuchtungen, Antriebe, Funkstörungen) aber auch systembedingte Auswirkungen der DDS an sich. Als Beispiel sei hier die Wahl des Tuning Words erwähnt.

Trotz der Störungen mit Pegeln oberhalb der -130 dBc / Hz ist das System auch für Dopplerradarsysteme geeignet. Es obliegt dann der Signalauswertung des Empfängers zu entscheiden, ob es sich bei den einzelnen Signalanteilen um Störungen oder wirkliche Zielreflexionen handelt.

4.4. Zusammenfassung

Entstanden ist bei der Arbeit ein auf dem DDS-IP-Core von Xilinx basierendes DDS-System, welches Ausgangsfrequenzen zwischen 0 und 149 MHz erzeugen kann, in Schritten von 0,02 Hz.

Am Ausgang steht ein komplexes digitales bzw. analoges Signal zur Verfügung.

Das System zeigt ein für DDS-Systeme sehr gutes Phasenrauschen, welches die Eignung für Hochfrequenzanwendungen wie Mobilfunk und Dopplerradar bestätigt.

Eine Fotografie des Gesamtsystems ist in Bild 4.4 zu sehen. Links im Bild das Virtex-II Pro Evaluationboard mit den aufgesetzten Adaptern, in der Mitte die Flachbandkabel, rechts das Board mit dem AD9786.

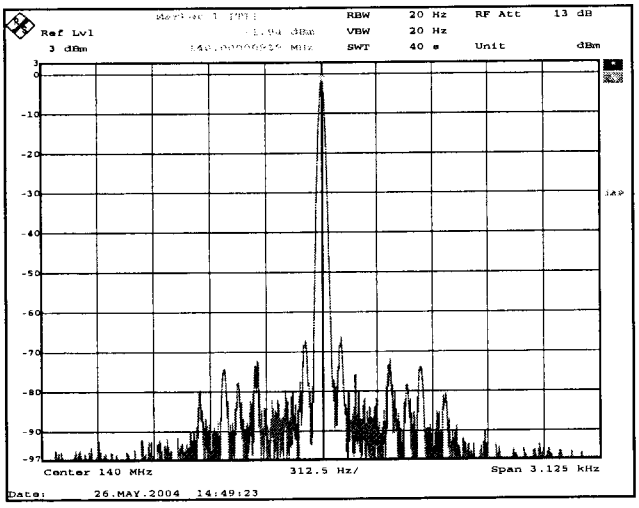


Bild 4.2: Spektrum bei 140 MHz

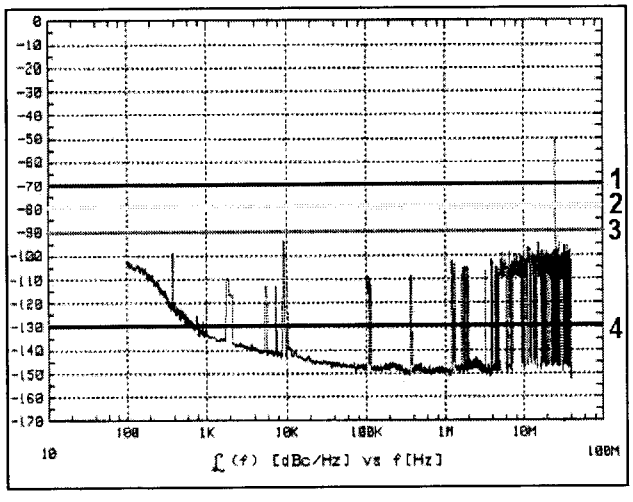


Bild 4.3: Phasenrauschen bei 125 MHz

5. Fortführungsmöglichkeiten

Diese Arbeit liefert eine solide Basis für weitere Entwicklungen, von denen hier einige erwähnt seien:

- Nutzung des auf dem Virtex-II Pro vorhandenen Power-PC-Kerns zur Steuerung des Tuning Words (Frequenzmodulation), des Phasenoffsets (Phasenmodulation) und der Skalierungseinheit (Amplitudenmodulation)
- Konfiguration des Digital Analog Converters über das SPI-Interface
- Einbindung weiterer Digital Analog Converter, interessant wäre auf jeden Fall die Verwendung des störsicheren LVDS-Interfaces
- Design eines Prototypen mit FPGA und DAC auf einer Platine

6. Literatur

Analog Devices, A Technical Tutorial on Digital Signal Synthesis, 1999, Paper (dds_tutor.pdf)

Eltawil / Daneshrad, Interpolation based Direct Digital Frequency Synthesis for Wireless Communications, 2002, Paper (taylor.pdf)

Ludloff, Praxiswissen Radar und Radarsignalverarbeitung, 1998, Vieweg Verlag

Siemens AG, Solutions for highly integrated future generation software radio basestation transceivers, 2001, Paper (CICC_MAY 2001.pdf)

Thumm / Wiesbeck / Kern, Hochfrequenzmesstechnik, 1998, Teubner Verlag

Xilinx, DDS v4.2 Product Specification, 2003, Datenblatt (dds.pdf)

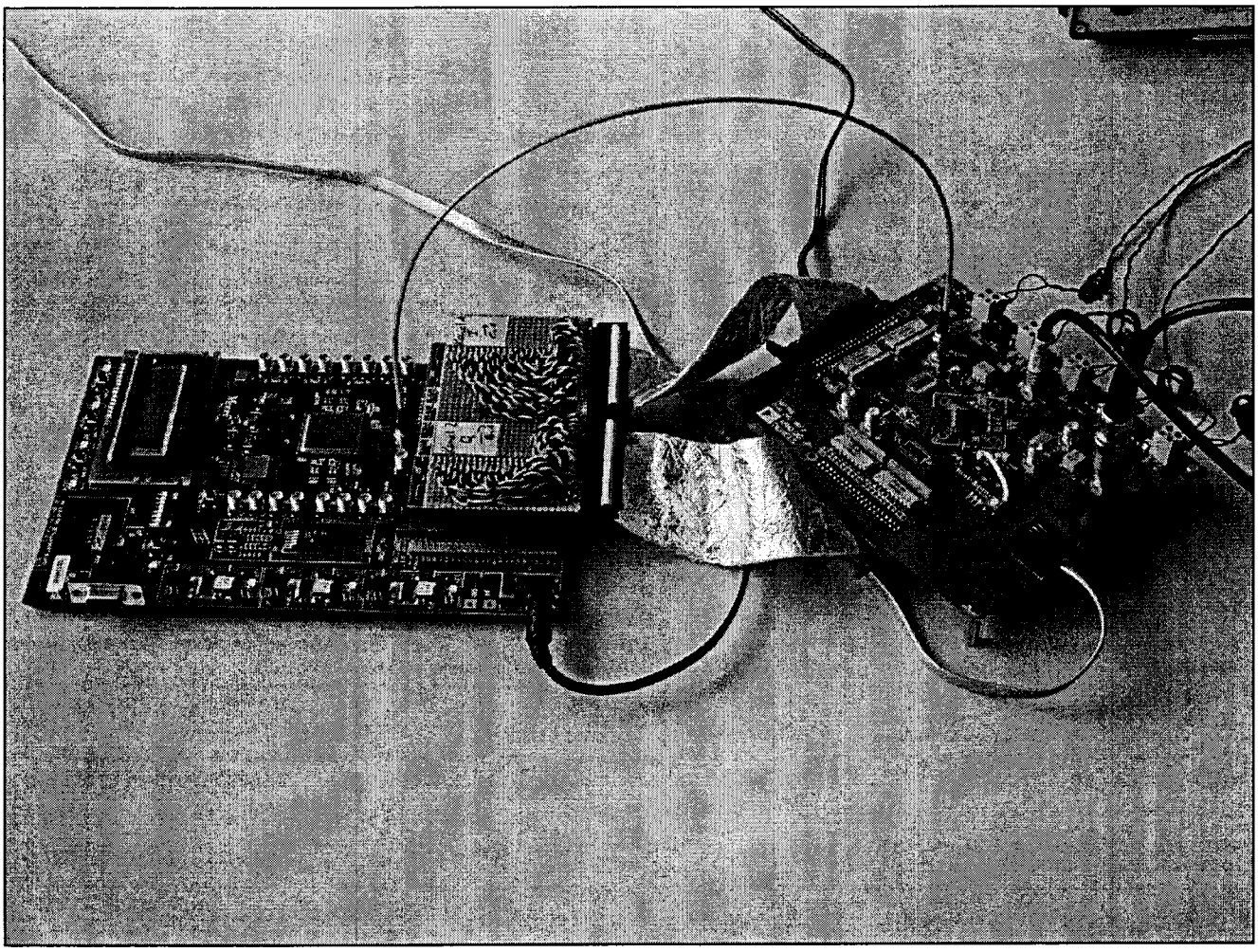


Bild 4.4: Fotografie des Gesamtsystems

Feasibility-Study & Concept Proposal for an optical Gas Sensor for use in automotive application

Alexander Müller

Fachhochschule Ulm, Prittwitzstraße

Abstract

The present work is a feasibility- study and concept proposal for an optical gas sensor for use in automotive applications. The motivation for this project are upcoming new developments of air-condition systems of the automotive industry. They use carbon dioxide as refrigerant medium. In case of leakage, carbon dioxide can bring serious health problems for occupants. For that reason, concentration detection is necessary. The aim of this thesis is to develop a prototype, which measures the concentration of carbon dioxide.

1. Introduction

The automotive industry develops in a new generation of air condition, which is using carbon dioxide (R744A) as refrigerant medium. The Mobile Air Conditioning Climate Protection Partnership (MAC-CPP), in which are all big automotive manufactures and automotive suppliers involved, expect the introduction of this air conditions, in approximately 2009. Presently hydro fluorocarbon 134A (HFC-134A) is used. Although this substance does not harm the ozone layer, it does have a high global warming potential (GWP) of 1300 [1]. Carbon dioxide is used as reference with the GWP of 1. This is the lowest value and all other gases are in relation to it.

The new air condition system will contain about 500 gram of carbon dioxide [1]. In connection with the high pressure (up to 120 bars) leakage cases can occur. Leakage may lead to the rise of the carbon dioxide concentration to a value of 4 to 8% inside a car. For a normal person it is not harmful but for occupants, having problems with their heart respectively lung, it can bring serious health problems. If the concentration of carbon dioxide is high enough the driver will become drowsy respectively, his reaction can be impaired. Then, the driver is a danger to road traffic.

Because of the mentioned risks, there is the need of leakage detection. Imagine the following situation: A car is parking at a supermarket, the engine is off, windows are closed, the heat exchanger burst and the concentration of carbon dioxide is increasing. The baby and the dog inside the car are inhaling the gas and suffocate.

2. Basic components and function of the sensor

There are certain basic components commonly used in all infrared gas sensors:

* an infrared emitter (e.g. incandescent light bulb)
* a detector (e.g. thermopiles, pyroelectric detectors)
* a filter to select appropriate wavelengths (e.g. band pass interference filter)
* a measurement chamber with gas inlet and gas outlet

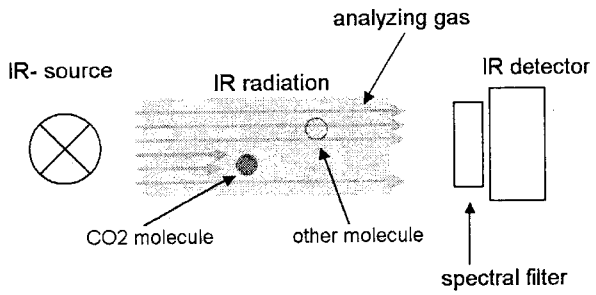


Figure 1: Principle setup of the sensor

The infrared light is emitted with an approximate black body spectral distribution. If the drive current will change, the temperature of the black body will also change. The simplest way to set the current is by using a resistor and a voltage reference. The infrared light emits radiation in the measurement chamber. The presence of gas in the chamber reduces the light

intensity by absorbing energy. This happens at a specific wavelength corresponding to the gas. The result is that the amplitude of the intensity of the radiation at the specific wavelengths drops down. So-called absorption bands come into being. Figure 2 shows clearly the absorption bands from some gases. CO2 begins to swing at a wavelength of ? = 4.3 ? m. Of course, this wavelength is found again in Figure 2.

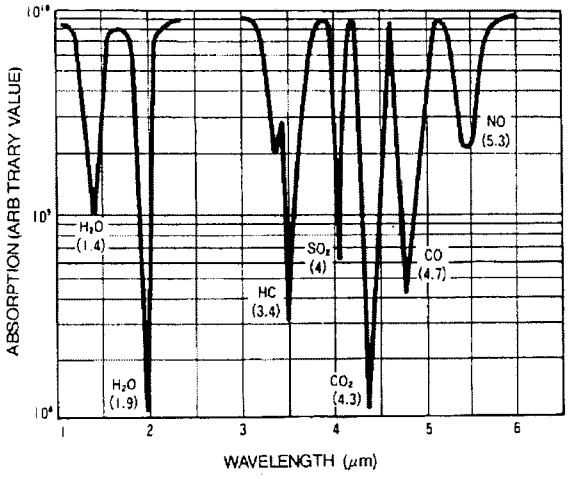


Figure 2: Absorption bands [2]

Most of the light corresponding to the gas specific absorption wavelengths gets absorbed in the chamber. The reduction of the light intensity occurs after the Law of Lambert Beer [3]. It depends on the distance between source and detector and of the concentration level.

The light is then transmitted through a bandpass interference filter. The filter characteristic is matched to the absorption band of the target gas. There are packages with the sensor element and the filter all in one. After the bandpass filter, the electromagnetic radiation heats generates warmth on the detector surface. The detector converts the heat into charge, which can be converted into a voltage.

3. Measuring of different concentrations

The highest concentration is limited by the premixed concentration of a gas bottle. It is 9000 ppm CO2. It was measured eight different concentrations: 0 ppm, 400 ppm, 1000 ppm, 2000 ppm, 4000 ppm, 6000ppm, 8000ppm, 9000 ppm. Every concentration was measured 100 times to get the functional characteristics. The average values of the amplitudes are shown in Figure 3. It is possible to insert a polynomial- or an exponential smoothing function. The theory describes that the amplitudes is an exponential function (Law of Lambert Beer [2]). Figure 3 shows that the polynomial function has less deviations matches better to the measurement points. Possible reasons are many small influences, which are not considered in the theory for example reflection of surface. Despite of this a clear functionality between concentration and amplitude could be found.

In addition, Figure 3 shows that the difference between reference and signal become lower at higher concentrations. At a concentration of 9000 ppm, the difference is nearly zero. On this account, the concentration cannot be measured with this sensor setup. The light path between IR source and detector is too large. The energy of the IR radiation was absorbed too much, thus too less energy hits the detector. A shorter measurement chamber is used higher concentrations can be measured. On the other side, the sensitivity at low concentrations will decrease. The conclusion is that the length of the measurement chamber and the sensitivity of the sensor cannot agree. Therefore, the automotive use is limited. Moreover, increasing the length, to get the right sensitivity, means need more space and that is a central problem at the automobile.

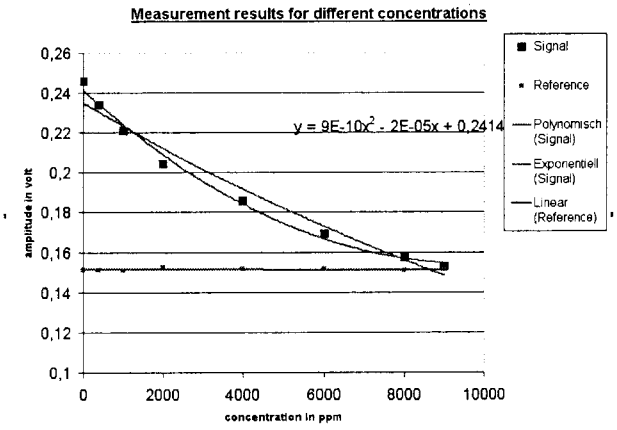


Figure 3: Average values of the amplitude at different concentrations

4. Conclusion

In the context to the work, a functional carbon dioxide sensor was built up and tested. Therefore, an automated test procedure is used. The measurements are saved by LabView.

Comparing the aimed objectives with the realized setup one can be very pleasant. In the next step the environmental conditions should change and the reaction to the sensor system should analyze. The following values should change systematically.

* Repeating changing of the temperature (the gas should has the same temperature as the sensor element!)
* Changing of the supply voltage (about 4.9 volt and 5.1 volt)
* Changing of the measurement chamber length (the manufactured chamber length)

Moreover, through **D**esign **o**f **E**xperiments (DoE) the measuring expenditure could be reduced and the dependency of the different parameters could be made visible.

In the end, it can be said that the sensor works reproduceable under normal condition. The automotive use is critical because during the implementation and measurements some criteria were coming up that made the automotive use difficult. There is the temperature behavior that makes the output signal unstable. Through the high amplification of about 1000, every small changing from the output signal of the sensor is shown after amplification. This effect is the most crucial argument against the feasibility.

[1] **World's first CO2 air conditioning system, Article in AutoTechnology 1/2004**

[2] **Jürgen Schilz PhD; thermophysica minima: applications of thermoelectric infrared sensors: Gas detection by infrared absorption, NDIR; PerkinElmer Optoelectronics GmbH; 2000**

[3] **Tomas Nezel; Mikrosensoren für die Detektoin von Gasen; Skript Gassensorik; Centre of Chemical Sensors Zurich; 2000**

Parameterbestimmung an integrierten Testdioden

Huanping Luo
Dr. U. Ricklefs
Dr. W. Bonath
FH Gießen-Friedberg
Wiesenstraße 14
D-35390 Gießen
0641-3091943
huanpingl@yahoo.de

Zusammenfassung

Das Verhalten von Photodioden wurde unter FEMLAB simuliert. Das grundsätzliche Verhalten der Dioden unter Lichteinfall konnte gezeigt werden. 3D-Strukturen waren nicht simulierbar.

Die maximale spektrale Empfindlichkeit zeigen die Dioden bei etwa 620nm.

Kleine Dioden konnten nicht gemessen werden, da die integrierte Schutzbeschaltung nicht abgeschaltet werden konnte. Ein sinnvoller Fit der Diodenparameter war für kleine Dioden nicht möglich. Große Dioden zeigen sinnvolle Ergebnisse.

Die Kapazität der Dioden wurde mit etwa 0,0005 – 0,0006 pF/µm^2 gemessen.

Das Projekt wurde vom HMWK gefördert. Die Teststrukturen wurden im Rahmen von Europractice erstellt.

1. Einleitung

Optische Sensoren werden in vielen Anwendungsgebieten eingesetzt. Der Bedarf an intelligenten kleinen Sensoren steigt. Photodiodenarrays sind mit Ausleseschalung und Signalaufbereitung auf einem Silizium-Chip als ASIC integrierbar.

In der Entwicklung analoger ASICs sind an der FH Giessen-Friedberg umfangreiche Erfahrungen vorhanden. In mehrere Projekten wurden Sensoranwendungen umgesetzt. Keine Erfahrungen waren bislang im Bereich der Integration optoelektronischer Komponenten vorhanden.

Im Umfeld der FH Giessen-Friedberg gibt es eine große Anzahl so genannter SMEs, die im optischen Bereich tätig sind. Eigen entwickelte optoelektronische Schaltkreise sind für diese Firmen zunehmend von Interesse. Um für diese Firmen als Kooperationspartner interessant zu sein, wurde das Projekt gestartet. Ziel war es dabei, die Machbarkeit der integrierten Photodioden mit Standardprozessen zu zeigen und deren Verhalten kennen zu lernen.

Nach Diskussionen mit mehreren Firmen wurde die notwendige Bandbreite für die Verstärker festgelegt. Um diese Bandbreiten erreichen zu können wurde der Standard 0,5µm analog-CMOS Prozess als Grundlage für die weiteren Untersuchungen festgelegt.

Um zu einer Einschätzung zu kommen, wurden auf einem Testchip unterschiedliche Photodioden realisiert. Im Rahmen dieser Arbeit wurden die zur Verfügung stehenden Photodioden gemessen und die Ergebnisse ausgewertet.

Mit dem Programm FEMLAB wurde versucht die Photodioden zu beschreiben.

Das Projekt wurde vom HMWK als Forschungsförderung gefördert. Die TestICs wurden im Rahmen von Europractice erstellt.

2. pn-Photodioden Aufbau

Im Standard-CMOS-Prozess stehen unterschiedliche Strukturen als Photosensoren zur Verfügung (/1- 6/). Als Photodioden sind der p+/n-Wannen-Übergang, der n-Wannen/p-Substrat-Übergang und der n-/p- Substrat-Übergang möglich. Als Photosensoren mit innerer Verstärkung sind der bipolare pnp-Phototransistor (c) und der Photo-MOSFET mit „floatender" n-Wanne (d) verfügbar (Abb.2.1).

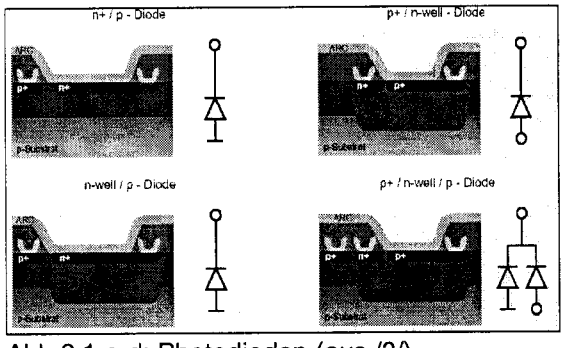


Abb.2.1 a-d: Photodioden (aus /3/)

Die obere linke Diode (a) hat den geringsten Platzbedarf und ist damit für hochauflösende Photodiodenrarrays geeignet, ist aber gegen das Umfeld nicht isoliert.
Eine Isolation vom Substrat wird erreicht, indem der pn-Übergnag in eine n-dotierte Wanne eingebettet wird. Diesen Fall zeigt die Diode oben rechts (b). Der Aufbau ist geeignet für Dioden, die innerhalb eines Arrays vollständig voneinander isoliert sein müssen.

Unterschiedliche Anwendungen stellen unterschiedliche Randbedingungen und Anforderungen an die eingesetzten Photosensoren, wie z.B. hohe Anforderung an die Empfindlichkeit, den Dynamikbereich oder die Ansprechzeit.

Die in dieser Arbeit untersuchten Photodioden entsprechen dem Typ b – d.

2.1 pn-Übergang

Der Grenzbereich zwischen einer p-dotierten Zone (p-Gebiet) und einer n-dotierten Zone (n-Gebiet) wird pn-Übergang genannt. Im p-Gebiet sind die Löcher Majoritätsladungsträger und die Elektronen Minoritätsladungsträger, während im n-Gebiet die Elektronen Majoritätsladungsträger und die Löcher Minoritätsladungsträger sind. Aufgrund des Unterschieds der Dotierungskonzentration diffundieren die freien Elektronen aus dem n-Gebiet ins p-Gebiet und freie Löcher aus dem p-Gebiet in das n-Gebiet. Zurück bleiben die ortsfesten ionisierten Störstellen (Raumladung), die eine Raumladungszone in der Grenzschicht aufbauen. Die beiden sich gegenüberstehenden Raumladungen erzeugen ein elektrisches Feld (die Diffusionsspannung), dessen Feldstärkevektor E vom n- zum p-Gebiet gerichtet ist. Das elektrische Feld wirkt dem Diffusionsvorgang entgegen, dadurch stellt sich schließlich ein stationärer Zustand ein. Der stationäre Zustand ist erreicht, wenn sich Driftströme, die durch das elektrische Feld entstehen, und Diffusionsströme, die durch Ladungsträgerdiffusion entstehen, kompensieren.

2.2 pn-Übergang mit Vorspannung

Beim Anlegen einer äußeren Spannung werden die vorher beschriebenen Gleichgewichtsverhältnisse gestört. Auf Grund der Polarität der anliegenden Spannung sind zwei Fälle möglich.

2.2.1 pn-Übergang in Sperrrichtung

An dem Halbleiter wird eine äußere negative Spannung angelegt (p-Zone am Minuspol und n-Zone am Pluspol). Durch die Dotierung haben sowohl das p- als auch das n-Materials eine hohe Leitfähigkeit. Dadurch fällt diese Spannung voll über der Raumladungszone des pn-Übergangs ab. Zudem liegt über der Sperrschicht selbst die Diffusionsspannung.

Die beiden Spannungen überlagern sich gleichsinnig. Dadurch wird die Potentialdifferenz vergrößert. Die Majoritätsträger müssen gegen die zunehmende Potentialschwelle anlaufen, so dass der Majoritätsträgerstrom schnell abnimmt. Der Minoritätsträgerstrom (Sperrstrom), der in Feldrichtung fließt, ändert sich aber praktisch nicht. Der Übergang ist damit hochohmig. Die Diode sperrt.

2.2.2 pn-Übergang in Durchlassrichtung

Man legt eine positive Spannung an den Halbleiter an. Die Spannung ist der Diffusionsspannung entgegen gerichtet. Nun laufen die Majoritätsträger gegen die kleinere Potentialschwelle, so dass der Majoritätsträgerstrom (Diffusionsstrom) zunimmt. Der Minoritätsträgerstrom wird nicht beeinflusst. Die Diode ist in Durchlassrichtung gepolt. Man muss aber dafür sogen, dass der zulässige Maximalstrom nicht überstiegen wird. Sonst tritt ein innerer Kurzschluss ein, der die Diode in der Regel zerstört.

2.3 pn-Photodiode

Die Photodiode wird in Sperrrichtung betrieben. Ohne Lichtzufuhr fließt ein Sperrstrom. Wenn Licht auf den pn-Übergang trifft, so werden die eindringenden Photonen im Kristall absorbiert und Elektron-Loch-Paare erzeugt. Die erzeugten Ladungsträgerpaare werden durch das elektrische Feld in der RLZ getrennt (beschleunigt) und die Elektronen zur N-Seite und die Löcher zur P-Seite gezogen. Dadurch fließt ein zusätzlicher Photostrom I_{ph}, der proportional zur einfallenden Strahlungsleistung P_{opt} ist.

Betrachtet man die einfallende Lichtleistung P_{opt} als einen Strom von Photonen, wobei jedes Photon die Energie $W_{photo} = h^*f$ trägt. Die Lichtleistung ist gleich dem Produkt aus der Zahl der Photonen N_{ph}, die innerhalb der Zeitspanne Δt mit der Photonenenergie W_{photo} in die Diode eindringen:

$$P_{opt} = h \cdot f \cdot \frac{N_{Ph}}{\Delta t}$$

Nicht alle auf die Oberfläche auftreffenden Photonen erzeugen Elektron-Loch-Paare, die zum Gesamtphotostrom beitragen. Dies wird in dem externen Wirkungsgrad ηext erfasst, der angibt, wie viele der auftreffenden Photonen zum Photostrom beitragen. Mit der Anzahl Npaar (Npaar= Iph. Δt/q) der je Zeiteinheit Δt in den Einflussbereich der RLZ gelangenden Paare gilt:

$$\eta_{ext} = \frac{N_{paar}}{N_{ph}} = \frac{I_{ph} \cdot h \cdot f}{q \cdot P_{opt}}$$

$$I_{ph} = \eta_{ext} \cdot \frac{q \cdot P_{opt} \cdot \lambda}{h \cdot c}$$

I_{ph} ist proportional zu P_{opt} und λ. Danach nimmt I_{ph} mit steigender Wellenlänge zu. Dies gilt allerdings nur in einem begrenzten Wellenlängenbereich. Bei großer Photonenenergie nimmt η_{ext} stark ab und bei zu kleiner Photonenenergie ist W_{photo} kleiner als die Bandlückenenergie Wg. Die Photonenenergie reicht dann nicht mehr aus, Elektronen ins Leistungsband zu heben.

Die folgende Gleichung beschreibt den Zusammenhang zwischen der auffallenden optischen Leistung und den elektrischen Größen. Der erste Term beschreibt die Diodenkennlinie mit dem Dunkelstrom I_{S0}. Angestrebt wird ein geringer Dunkelstrom, da er das Rauschen erhöht.

$$I_D(U, P_{opt}) = I_{S0} \cdot \left(\exp\left(\frac{U}{U_T} - 1 \right) - I_{ph}(P_{opt}) \right)$$

Der zweite Term ist der Photostrom. Gemäß der obigen Gleichung erhält man $I_{ph}(P_{opt})$ unmittelbar durch Messung des Stromes im Kurzschlussfall: $I_{ph}=I_D(U, P_{opt})(U=0)$.

Aus der Gleichung erhält man die Kennlinie der pn-Photodiode, Abb1.2 stellt die Kennlinie für verschiedene Werte von P_{opt} graphisch dar.

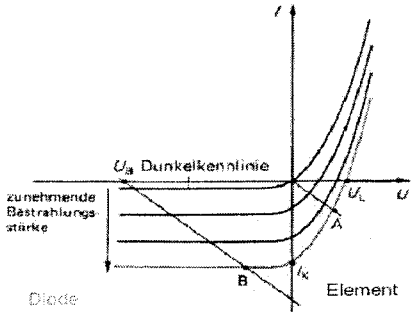


Abb 2.2 Photodiodenkennlinie

3. Parameter

Im Folgenden werden die charakterisierenden Parameter von Photodioden behandelt.

3.1 Empfindlichkeit

Die spektrale Empfindlichkeit (Responsivität) ist definiert als:

$$R(\lambda) = \frac{I_{ph}}{P_{opt}} = \eta_{ext} \cdot \frac{\lambda \cdot q}{h \cdot c}$$

Die Abhängigkeit R(λ) ~ λ kann anschaulich so erklärt werden, dass mit abnehmender Wellenlänge bzw. zunehmender Photoenergie bei gleicher optischer Leistung weniger Photonen je Zeiteinheit auf die Halbleiteroberfläche auftreffen.

Zusätzlich spielt auch die Geometrie eine erhebliche Rolle. Die Absorption der Strahlung im Halbleitermaterial kann durch das Lambert-Beersche Gesetz beschrieben werden. Die Absorptionskonstante ist aber abhängig von der Wellenlänge. Mit steigender Wellenlänge nimmt die Eindringtiefe zu (Abb.3.1). Dementsprechenden nimmt die Ansprechzeit auch zu. Da man auf den Herstellungsprozess keinen Einfluss hat, sollte die Wellenlänge der Strahlung, die der Diode zur Verfügung gestellt wird, der Lage der RLZ angepasst sein (Filter).

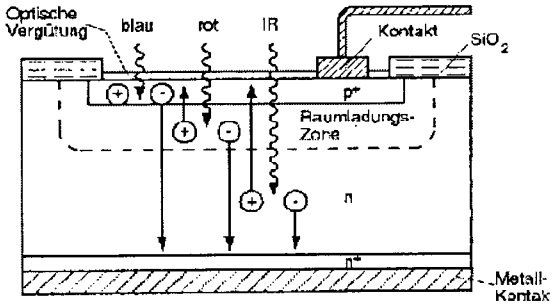


Abb. 3.1 Diodenaufbau

Nach Bludau /5/ gilt:

$$\eta_{ext} = (1 - R)e^{-\alpha \cdot d_p} \cdot \left(1 - \frac{e^{-\alpha \cdot d_{RLZ}}}{1 + \alpha \cdot L_p} \right)$$

(α Absorptionskonstant, d_p Abstand der RLZ zur Oberfläche, d_{RLZ} Breite der RLZ, L_p Diffusionslänge) Die Gleichung zeigt die

Abhängigkeit der Empfindlichkeit von Material und Geometrie.

Bei der Herstellung eines ASICs hat man keinen Einfluss auf die Wahl dieser Parameter. Durch die Fixierung auf eine Technologie – in unserem Fall die 0.5 µm-analog CMOS Technologie – sind diese Parameter vorgegeben. In der Regel sind die Hersteller auch nicht bereit nähere Informationen hierüber heraus zu geben. Die einzigen Geometrieparameter die variiert werden können, sind die äußeren Geometrieparameter.

3.2 Kapazität

Die Kapazität begrenzt die erreichbare Bandbreite.

Die Raumladungszone speichert elektrische Ladungen. Der eine Teil der RLZ enthält fest eingebaute negative Ladungsträger, der andere fest eingebaute positive Ladungsträger. Elektrotechnisch wirkt die PD-Sperrschicht wie eine Kapazität C.

$$C_i = \text{const} \cdot \varepsilon \cdot \frac{A}{d_{RLZ}}$$

Bei einer Änderung des Abstandes der pn-Sperrschicht ändert sich die Kapazität. Da die Breite der RLZ von der Spannung abhängig ist, lässt sich so die Kapazität und Bandbreite verändern.

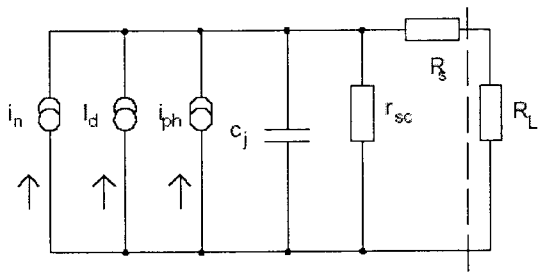


Abb.3.2 Ersatzschaltbild

Der Shuntwiderstand r_{sc} (Abb 3.2) ist normalerweise vernachlässigbar. Deshalb wird die Photodiode vereinfacht als Stromquelle mit der parallel geschalteten Kapazität C_j und einem Serienwiderstand R_s, der häufig auch vernachlässigbar ist, dargestellt. R_L ist der wirksame äußere Lastwiderstand. Die Zeitkonstante τ mit $\tau = C_j^*(Rs+R_L)$ beschreibt mit $f_B = \tau^{-1}$ die elektrische Bandbreite und stellt somit die entscheidende Begrenzung dar. Eine größere Bandbreite ist nur über einen niedrigen Widerstand R_L oder durch eine geringe Kapazität C_j realisierbar.

Die Ansprechzeit der Photodiode ist nicht allein durch die Kapazität der PD bestimmt, sondern auch durch die so genannte Ladungsträger-Sammelzeit. Ladungstäger, die außerhalb der RLZ erzeugt werden, benötigen diese Zeit, um in die Randbereiche der RLZ zu diffundieren. Dieser Vorgang ist relativ langsam. Beeinflusst werden kann diese Zeitkonstante durch die Vorspannung und den Aufbau der Diode mit einer dünnen p-Schicht. Durch die Herstellervorgaben kann diese aber nicht verändert werden.

3.3 Rauschen

3.3.1 Schrotrauschen des Dunkelstroms

Durch die Quantelung der elektrischen Ladung entsteht das so genannte Schrotrauschen.

$$i_{Schrot} = \sqrt{2 \cdot q \cdot I_D \cdot B}$$

(B: Bandbreite) Integrierte Photodioden wird man ohne Vorspannung auf dem ASIC kaum betreiben können. Damit hat man einen Diodenstrom. Das Rauschen dieses Stroms lässt sich nicht kompensieren.

3.3.2 1/f -Rausch

Bei pn-Übergang tritt zusätzlich ein 1/f-Rausch auf, dessen Rauschdichte umgekehrt proportional zur Frequenz ist.

$$i_f = k \cdot \left(\frac{1}{f}\right)^m$$

k und m sind experimentelle Konstanten .

4. Simulation mit FEMLAB

Die Photodioden wurden mit FEMLAB modelliert. Bei der Modellierung wurde die räumliche Ladungsträgerkonzentration, die Bilanzgleichungen für die Feld- und Driftstrome und der Strahlungstransport im Halbleiter berücksichtigt. Man erhält einen Satz von gekoppelten Differentialgleichungen. Die ungefähren Geometriedaten und Dotierungen wurden den Herstellerangaben entnommen.

Zuerst wurde eine 2D-Diode im vertikalen Schnitt modelliert. Als Ausgangspunkt für die Erstellung der Struktur diente das Dotierungsprofil des 0.5 µm Prozess. Die folgende Abbildung (Abb 4.1) zeigt die Ladungsträgerverteilung der Diode ohne Lichteinstrahlung Abb 4.1 und mit Lichteinstrahlung Abb 4.2.

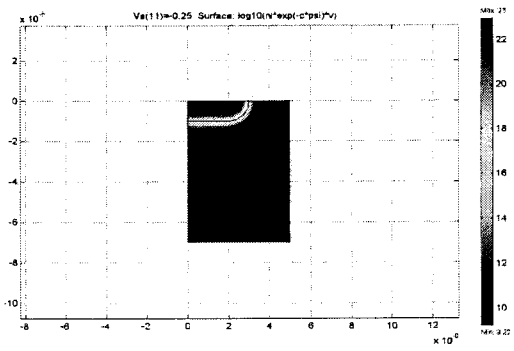


Abb. 4.1 Ladungsträgerverteilung ohne Lichteinstrahlung

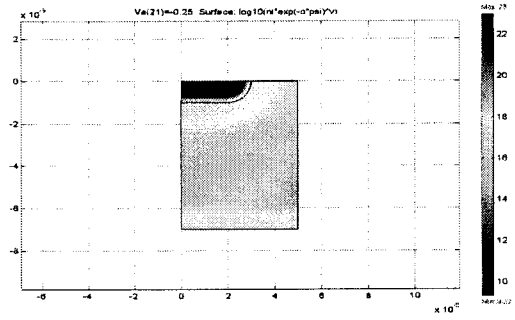


Abb. 4.2 Ladungsträgerverteilung mit Lichteinstrahlung

Nach der Modellierung der Diode startet man mit einer FEM-Struktur, die in der Regel noch zu keinen Ergebnissen führt. Durch Verfeinerung der FEM-Struktur kann man mit großer Rechenzeit diese Ergebnisse erhalten.

Im 2D-Fall konnten mit der Simulation die Diodenkennlinie und der Photostrom simuliert werden.

Die uns eigentlich interessierende 3D-Modellierung scheiterte aus numerischen und Speicherverwaltungs-Gründen.

5. Versuchsdurchführung

5.1 Testdioden

Auf dem Testchip sind 14 Dioden (Typ a – d) untergebracht. Die Dioden haben unterschiedliche Geometrie und Größe. Im Randbereich des Chips befinden sich Schutzdioden. Bei kleinen Dioden und einfachen diodenstrukturen können die Schutzdioden zu deutlichen Fehlmessungen des Dunkelstroms führen.

Bei den Messungen wurden die Schutzdioden gesperrt. Zudem wurde der Randbereich des Chips durch schwarze Plättchen aus Metal abgeschattet, um den Photostrom der Schutzdioden zu unterdrücken.

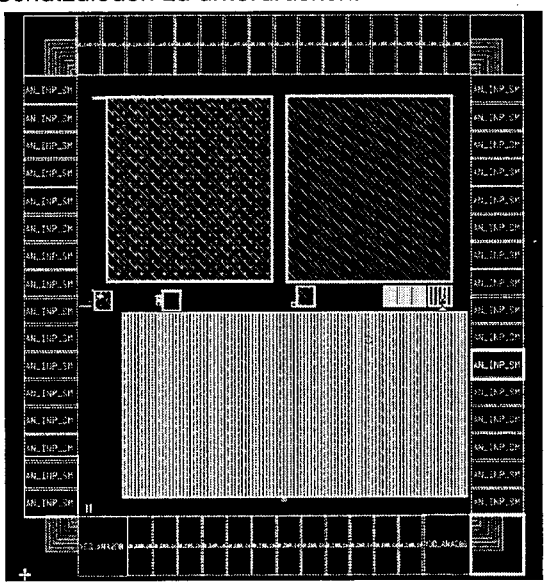


Abb. 5.1 Testchip

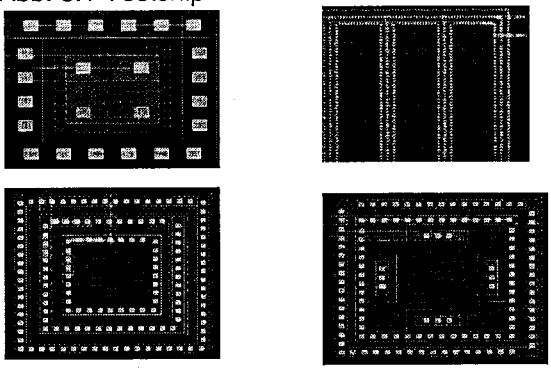


Abb. 5.2 Teststrukturen

5.2 Messung von Photostrom und spektraler Empfindlichkeit

Die Messung des Photostromes wird mit einem kalibrierten Vergleichsempfänger durchgeführt. Dabei werden die Dioden mit Haolgenlampenlicht, dass mit einem Lichtleitbündel übertragen wurde, beleuchtet. Eine Erwärmung der Dioden wird so verhindert. Vorteilhaft ist dabei, dass das Beleuchtungsniveau hoch genug ist, um ausreichend große Signale zu erzeugen.

Zur Messung der spektralen Empfindlichkeit sollten die Dioden über einen Monochromator, der einen engen Spektralbereich zur Verfügung stellt, beleuchtet werden. Dabei sind die Beleuchtungsstärken allerdings so klein, dass an eine sinnvolle Ausleuchtung und die gleichzeitige Messung mit einem Vergleichsempfänger nicht mehr zu denken ist. Dies wäre nur über integrierte I/U-Wandler möglich.

Nach Möglichkeit sollten Dioden und Vergleichsempfänger gleichzeitig über eine Ulbricht-Kugel (integrating sphere) gleichmäßig ausgeleuchtet werden. Leider schlucken diese Kugeln sehr viel Licht. Da die U-Kugel noch nicht geliefert wurde, wurde versucht, über eine entsprechende Ausrichtung der Empfänger für eine annähernd gleichmäßige Ausleuchtung zu sorgen. Empfänger und Testdioden wurden nach einander gemessen. Als Lichtquelle wurden fünf LEDs unterschiedlicher Wellenlänge gewählt.

Diese Anorderung ermöglichte es, verschiedene Photodioden zu vergleichen. Quantitativ kann die spektrale Empfindlichkeit so nicht bestimmt werden.

Die Messung des Photostroms läuft computergesteuert. Die Software Labwindows/CVI steuert über USB das Geräte QADPAD-6052E(National Instruments) und bekommt die Messdaten. Bild 3-2 zeigt die Beschaltung der Diode zur Messung. Variiert wird die Spannung U.

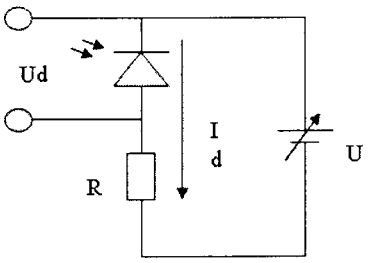


Abb. 5.3 Diodenbeschaltung

5.3 Kapazitätsmessung

Mit einem RCL Gerät wird die Kapazität bei unterschieden Sperrspannungen gemessen. Um sicher zu sein, dass mit dem Gerät tatsächlich die Kapazität bestimmt wurde, wurde das Ergebnis mit einer Empfängerschaltung überprüft. Beide Ergebnisse sind vergleichbar.

6. Ergebnisse

6.1 Strom-Bestrahlungsstärke-Kennlinie der Diode

Der gemessene Photostrom ist proportional zur Bestrahlungsstärke bzw. Leistung. Das in Abb 6.1 dargestellte lineare Verhalten ist typisch.

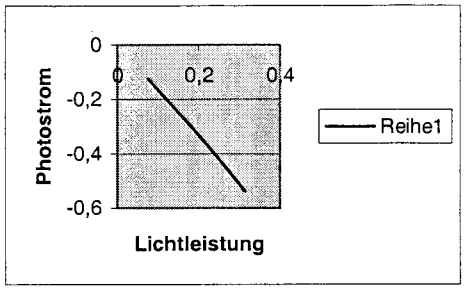


Abb. 6.1 $I_{PD}(P_{opt})$

6.2 Diodenempfindlichkeit

Es wurde der Einfluss des Diodentyps auf die Empfindlichkeit untersucht. Zu diesem Zweck wurden die Dioden vom Typ a-d gemessen. Der Arbeitspunkt der Lichtquelle (Weißlicht) und die Sperrspannung wurden bei diesen Messungen konstant gehalten. Zur Überprüfung der Linearität wurde der Abstand zur Lichtquelle variiert. Die Linearität war bei allen Dioden gegeben.

Die Größe der zur Verfügung stehenden Dioden ist sehr unterschiedlich. Deshalb ist ein direkter Vergleich nicht möglich.

Zum Vergleich werden die gemessene Ströme der Dioden durch die Diodenfläche geteilt. Irreführender Weise wird auch diese Angabe häufig als Empfindlichkeit bezeichnet.

Tab. 6.1

Diode	Typ	I_{PD}/µA	Fläche/µm²	I_{PD} /Fläche
4	c	0,066	81	0,00081005
3	c	0,149	327	0,00045609
10	a	0,088	277	0,00032023
6	d	0,103	676	0,00015271
7	d	0,084	676	0,00012568
9	a	0,029	309	9,3692E-05
5	c	0,676	11664	5,7994E-05
8	d	0,497	13456	3,6936E-05
11	a	0,477	13572	3,5192E-05
13	a	0,469	17035	2,7555E-05
12	a	0,660	30265	2,1822E-05
1	d	6,987	1025156	6,8163E-06
2	c	6,771	1025156	6,6058E-06
14	d	8,247	2141396	3,8514E-06

Aus der Tabelle ersieht man, dass die Empfindlichkeit fast nicht von der geometrischen Struktur abhängig ist, sondern in erster Linie von der Größe. D.h. je kleiner die Diode ist, je höher ist die gemessene Empfindlichkeit.

Zur Überprüfung wurden die Messungen mit einem Multimeter wiederholt, das eine Messunsicherheit von wenigen nA hat. Überprüft werden konnten nur die Messungen für die drei größten Dioden. Die Messungen bestätigen die Ergebnisse der automatischem Messmethode.

Bei den anderen 11 kleineren Dioden konnten die Ströme mit dem Gerät nicht gemessen werden. Bei der automatischen Methode wurden aber einige Hunderte nA gemessen. Abschätzungen zeigen, dass die Auflösung des Gerätes von National Instrument nicht hoch genug ist, um Ströme unter 100nA zu messen.

6.3 Diodenkennlinie

Die Parameter der Diodenkenlinien wurden mit der Methode der Kleinsten-Fehlerquadtrate an die Messwerte angepasst.

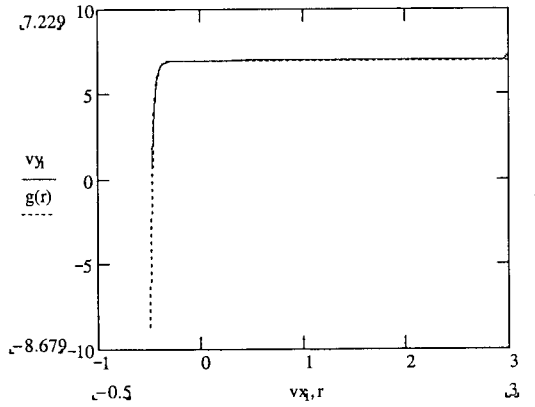


Abb. 6.2 Parameterfit große Diode

I_{S0}/mA	$-8.83*10^{-8}$
U_T/V	0.026
I_{opt}/mA	-6.924

Tab. 6.2 Kennlinienparameter Diode 1

Die rote Kurve ist die graphische Darstellung des Fits. Die blauen Punkte sind die Messwerte von Photostrom und Diodenspannung. Die beiden Kurven stimmen sehr gut über ein. Die bestimmten Parameter sind sinnvoll.

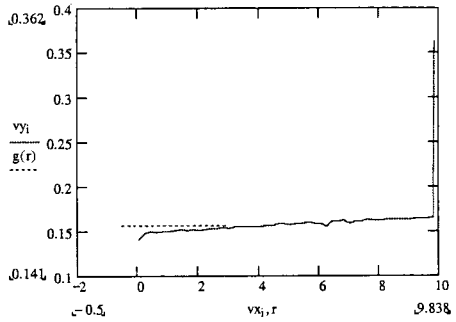


Abb. 6.3 Parameterfit kleine Diode

I_{S0}/mA	$-1.04*10^{13}$
U_T/V	$1.18*10^{17}$
I_{opt}/mA	-0.157

Tab. 6.3 Kennlinienparameter Diode 1

Nach Abb. 6.3 kann die Diodenkennlinie zwar an die Messwerte angepasst werden, die Parameter machen aber keinen physikalischen Sinn.

Im Rahmen der Arbeit war es nicht möglich, die kleinen Signale zu messen und aus den zur Verfügung stehenden Photodioden die beste auszuwählen.

Es lässt sich vermuten, dass die Schutzdioden die Messung der kleinen Dioden unmöglich machten. Gerade der Dunkelstrom wird durch die Schutzbeschaltung stark beeinflusst. Eine Kompensation des Schutzdiodeneinflusses war nicht möglich.

6.4 Kapazität

Die Kapazität wurde mit Messbrücke und zum Vergleich mit einem Empfänger gemessen. Dabei zeigte sich, dass die Kapazität etwa

$C/A = 0.00051 - 0.00056\ pF/\mu m^2$

betrug. Ein wesentlicher Einfluss der Sammelladungszeit konnte nicht festgestellt werden.

6.5 Spektrale Empfindlichkeit

Mit dem angegebenen Verfahren wurde die spektrale Empfindlichkeit gemessen. Da aber die Messfehler noch zu groß sind, sind diese Messungen wenig zuverlässig. Es zeigt sich aber eine maximale Empfindlichkeit im roten Wellenlängenbereich zwischen 600 nm und 650 nm (630 nm).

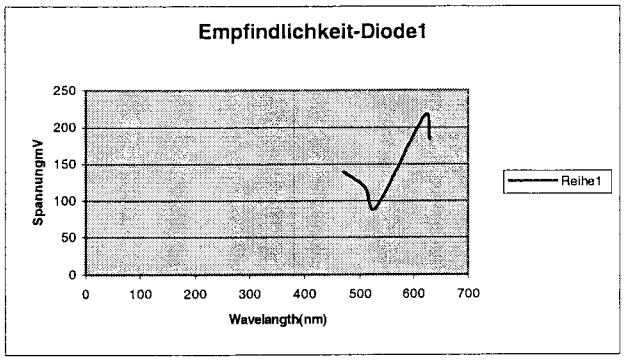


Abb. 6.4

Literatur

/1/ **www.uni-kassel.de/fb16/fsg/dateien/wde/Skript/WDE_TEIL_07.PDF**
/2/ **www.tu-chemnitz.de/etit/opto/ lehre/praktikum/ked.pdf**
/3/ www.pb.izm.fhg.de/mimosys/publish/status99-2/4apew/V3-thesys.pdf
/4/ Bodo Morgenstern: Elektronik 1, Vieweg (1993)
/5/ Klaus Beuth: Bauelemente, Vogel Verlag (1997)
/6/ W. Bludau: Halbleiter-Optoelektronik, Hanser (1995)

FACHHOCHSCHULE FURTWANGEN
HOCHSCHULE FÜR TECHNIK UND WIRTSCHAFT
UNIVERSITY OF APPLIED SCIENCES

Schnelle Kulisch-Arithmetik auf einem FPGA

Wolfgang Rülling
Fachhochschule Furtwangen

Es wird eine Schaltung vorgestellt, mit der beliebig viele reelle Zahlen vom Typ `double` summiert werden können, ohne dass sich dabei Rundungsfehler akkumulieren. Das Endergebnis ist also im Rahmen der Darstellungsgenauigkeit von `double`-Zahlen exakt.
Als Besonderheit werden alle Daten in RAMs gespeichert, um den Hardwareaufwand zu reduzieren. Trotzdem werden pro Addition nur zwei Taktzyklen benötigt.

1 Einführung in die Kulisch-Arithmetik

Das Ziel der **Kulisch-Arithmetik** besteht darin, arithmetische Grundfunktionen so zu implementieren, dass sie (im Rahmen der Darstellungsgenauigkeit der Zahlen) zu exakten Ergebnissen führen. D.h. das Ergebnis muss bis auf eine Rundung stimmen.
Eine Grundoperation, bei der dies nicht selbstverständlich ist, ist das Skalarprodukt von Vektoren

$$(a_1, a_2, \ldots, a_n) * (b_1, b_2, \ldots, b_n) = \sum_{i=1}^{n} a_i \cdot b_i$$

Sind die beteiligten Vektorkomponenten Zahlen vom Typ `float`, so sind bei der Berechnung mehrere Produkte vom Typ `double` zu summieren. Dabei entstehen typischerweise Rundungsfehler. Als Beispiel betrachten wir folgende Summation in der Programmiersprache C++

```
double x1= 1E100;
double x2= 300000;
double x3= -x1;
double y=  x1 + x2 + x3;
cout << y << endl;
```

Hier sollte y= 300000 entstehen, aber tatsächlich erhält man y=0, weil das Zwischenergebnis x1+x2 fälschlicherweise aufgrund der beschränkten Darstellungsgenauigkeit von double-Zahlen den Wert x1 annimmt.
Per Software kann man derartige Probleme durch eine Sortierung der Summanden nach ihrer betragsmäßigen Größe vermeiden, was allerdings zu einer wesentlichen höheren Rechenzeit führt. Deshalb ist eine Spezialhardware für die Kulisch-Arithmetik wünschenswert, die die Summation von double-Zahlen (intern) rundungsfehlerfrei durchführt und erst das Endergebnis in den Datentyp double konvertiert.
Die benötigten Operationen eines solchen Rechnerwerks sind

1. `init():` `accu = 0;`
2. `add(double x):` `accu = accu + x;`
3. `double get():` `y = double(accu);`

Nach der IEEE-Norm 754 werden double-Zahlen mit einer Länge von 64 Bit dargestellt. Sie bestehen aus einem Vorzeichenbit, 11 Bits für den Exponenten und einer Mantisse mit 52 Bits. Der maximal zulässige Exponent ist $2^{11} - 2 = 2046$.

VZ	Exponent	Mantisse

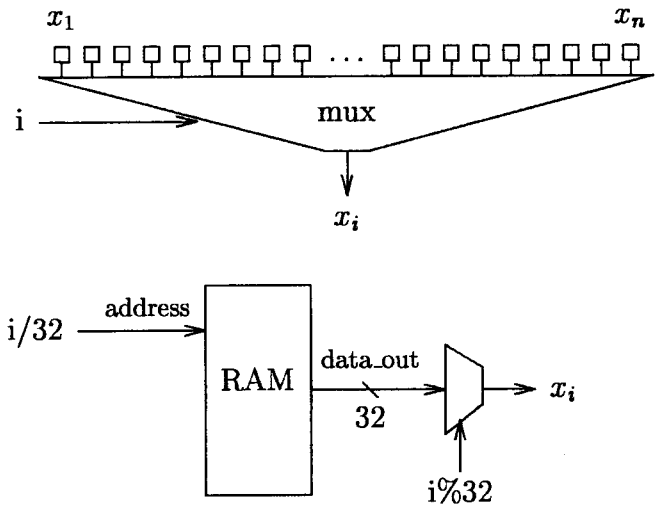
Der Wert der so dargestellten Zahl ergibt sich gemäß folgender Formel:

$$x = (-1)^{\text{Vorzeichen}} \cdot 1.\text{Mantisse} \cdot 2^{\text{Exponent}-1024}$$

Die betragsmäßig kleinste darstellbare Zahl ist etwa 4.9E-324 und die größte Zahl ist etwa 1.8E308. Als einfaches Beispiel für eine double-Zahl betrachten wir $x = +5 \cdot 2^{-8}$. Sie lässt sich durch $(-1)^0 \cdot (1.01)_{\text{bin}} \cdot 2^{-6}$ darstellen. In diesem Fall gilt also Vorzeichen $= 0$, Mantisse $= (01000\ldots000)_{\text{bin}}$ und Exponent $= 1018 = (01111111010)_{\text{bin}}$.
Um solche Zahlen in einem Festkomma-Zahlsystem darzustellen, muss man die Mantisse entsprechend dem Wert des Exponenten verschieben. Deshalb braucht man einen Akkumulator mit insgesamt $53+2^{11}-2 = 2099$ Binärstellen. Eine naheliegende Implementierung besteht deshalb darin, einen Akkumulator und einen Addierer für die Datenwortlänge 2099 zu verwenden. In Software wurde eine solche Implementierung als **golden device** in C++ vorgenommen. Mit dem Programm kann man einerseits untersuchen, wie stark die Ergebnisse der klassischen Arithmetik im Computer von der Kulisch-Arithmetik abweichen. Zum Anderen lassen sich damit auch sehr gut Testdaten zur Beurteilung von Hardware-Implementierungen erzeugen.

2 Ziele beim Schaltungsentwurf

In der Literatur findet man mehrere Ansätze für geschicktere Hardware-Realisierungen der exakten Summation von `double`-Zahlen (siehe [1], [2], [4], [5]). Ihnen ist gemeinsam, dass lediglich ein relativ kleiner Addierer verwendet wird und die Addition einer `double`-Zahl in mehreren Taktzyklen durchgeführt wird.
Die Motivation für die vorliegende Arbeit war, die Schaltung auf einem relativ kleinen FPGA der Firma XILINX zusammen mit einem PCI-Interface zu realisieren. Auf diese Weise sollte das Rechenwerk auf einfache Weise unter Verwendung eines an der FH Furtwangen verfügbaren FPGA-Demoboards als Co-Prozessor in einem PC verwendet werden können.
Da die Schaltung extrem klein werden muss, ist es beispielsweise völlig ausgeschlossen, den Akkumulator mit Flipflops zu realisieren, weil damit bereits 45% der verfügbaren FPGA-Kapazität nur für die Flipflops benötigt würden. Eine interessante Alternative besteht darin, die Daten in RAMs zu halten. Bei den verwendeten FPGAs lassen sich in einem CLB (Configurable Logic Block) wahlweise 32 Bit RAM-Speicher oder 2 Flipflops realisieren. Damit braucht ein Flipflop 16 mal soviel Ressourcen wie ein RAM-Bit.
Der Einsparungseffekt verschärft sich noch, wenn man neben den Speicherelementen auch die erforderlichen kombinatorischen Schaltungsteile zum Zugriff auf die Speicherdaten betrachtet. Will man beispielsweise aus dem Akkumulator der Länge 2099 ein Bit an einer bestimmten Position i herausgreifen, so braucht man einen Multiplexer mit 2099 Dateneingängen.



Bei der RAM-Lösung mit einer Datenwortbreite von beispielsweise 32 Bit würde man allein durch Anlegen der Adresse $i/32$ den Suchbereich auf ein

Datenwort reduzieren, so dass man nur noch einen Multiplexer mit 32 Dateneingängen benötigt. Damit wäre die Auswahlschaltung bei Verwendung von Flipflops in diesem Beispiel etwa um den Faktor 2099/32=66 größer.
Grob gesagt besteht dieser zweite Einsparungseffekt bei den RAMs gegenüber den Flipflops darin, dass man Operationen auf kurzen Adressen anstatt auf langen Daten durchführen kann. Als Nachteil der RAM-Speicher ist allerdings zu beachten, dass man in jedem Takt nur ein Datenwort zugreifen kann. Trotzdem sollte das Rechenwerk pro Addition mit nur zwei Takten auskommen.

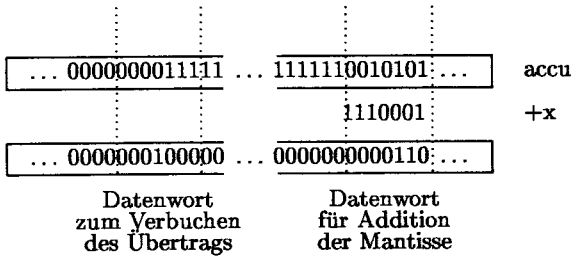
Zusammenfassung der Designziele:

- kleine Hardware durch Verwendung von RAMs statt Flipflops
- pro Addition maximal zwei Taktzyklen

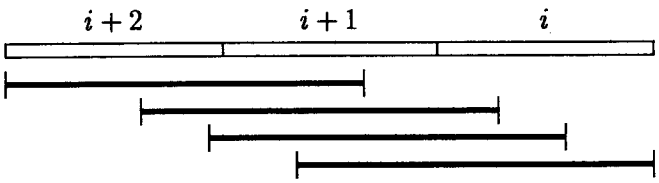
3 Prinzipielle Vorgehensweise

3.1 Speicherung des Akkumulators in RAMs

Zur Vereinfachung der Darstellung nehmen wir an, ein positiver Summand x soll zum Akkumulator addiert werden. Dann muss man entsprechend dem Exponenten von x einen 53-Bit langen Bereich des Akkumlators auswählen und dort lokal die Mantisse addieren. Lediglich wenn sich dabei ein Übertrag ergibt, muss eine eventuell links davon stehende Kette von Einsen in eine 0-Kette überführt werden und davor der Übertrag im Akkumulator verbucht werden. (ähnliches gilt für Subtraktionen, bzw. die Addition negativer Summanden.)



Um einen beliebigen 53-Bit langen Bereich im Speicher selektieren zu können, wird der Akkumulator in Blöcke zu je 32 Bit unterteilt, die mit Hilfe der 6 höherwertigen Bits des Exponenten adressiert werden können. Dann reicht es für jede beliebige Lage der Mantisse aus, drei aufeinanderfolgende Blöcke aus dem Speicher zu holen, um den benötigten Datenbereich zu erhalten.

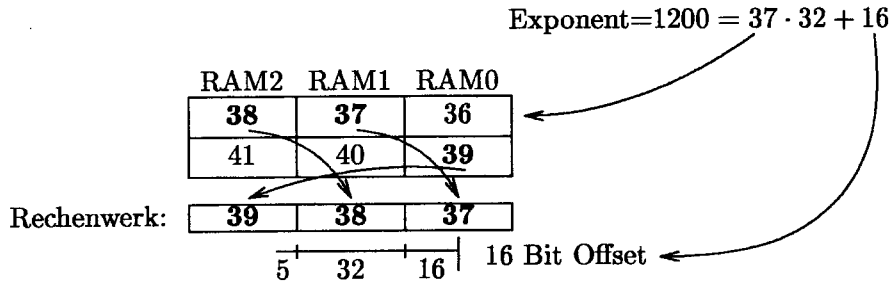


Einige mögliche Lagen der Mantisse relativ zu drei aufeinanderfolgenden Blöcken des Akkumulators

Um jeweils drei Blöcke des Akkumulators im gleichen Takt im Speicher zugreifen zu können, werden die Blöcke folgendermaßen auf drei RAMs verteilt.

RAM2	RAM1	RAM0
2	1	0
5	4	3
8	7	6
...		
65	64	63

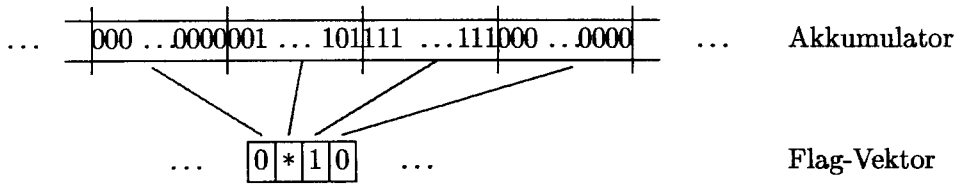
Auf diese Weise werden pro RAM 22 Datenwörter mit je 32 Bit Länge gebraucht. Der Hardwarebedarf auf dem XILINX-FPGA beträgt dafür 96 CLBs. Tatsächlich werden auf diese Weise sogar 32 Datenwörter pro RAM bereitgestellt, so dass ohne Mehraufwand auch größere Akkumulatoren verwendet werden könnten. Die Speicherung des Akkumulators in Flipflops hätte stattdessen 1050 CLBs benötigt.
Die folgende Skizze zeigt am Beispiel des Exponenten $1200 = 37 \cdot 32 + 16$, wie der Zugriff auf den Speicher erfolgt und in welchem Bereich die Addition der Mantisse durchgeführt werden muss.



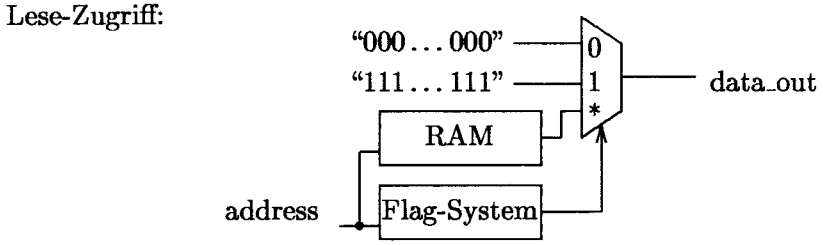
Beispiel des Speicherzugriffs für einen Summanden x mit Exponenten 1200

3.2 Verwendung eines Flag-Systems

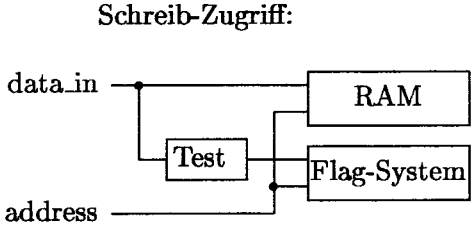
Bei der bisher vorgestellten Speicherung des Akkumulators in RAMs besteht weiterhin das Problem, wie man sehr viele Bits gleichzeitig setzen kann. Beispielsweise sollen bei einem *init*-Befehl gleichzeitig alle Bits auf 0 gesetzt werden. Ähnliches braucht man beim "Umkippen" einer 1-Kette zu einer 0-Kette, da hier ebenfalls viele Bits gleichartig modifiziert werden müssen.
Ein einfacher Lösungsansatz für dieses Problem besteht darin, dass man für jedes der 66 Datenwörter ein zusätzliches Flag einführt, das kennzeichnet, ob das Datenwort "konstant 0", "konstant 1" oder "gemischt" belegt ist. Dazu sollen im folgenden die Symbole "0", "1" und "*" verwendet werden. Auf diese Weise kann der Akkumulatorinhalt grob mit nur 66 Flags dargestellt werden.



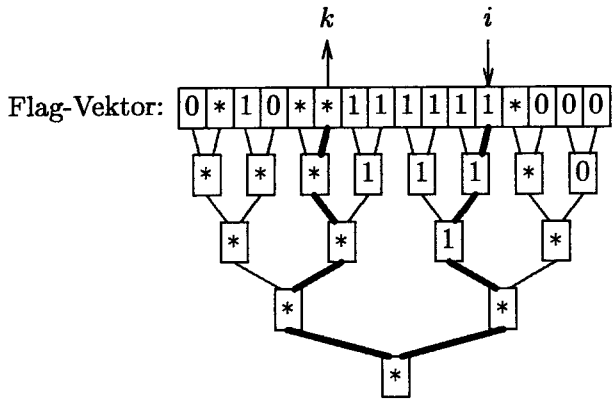
Beim Lesezugriff auf den Akkumulator wird nun zunächst in den Flags nachgesehen, ob das betreffende Datenwort "konstant 0" oder "konstant 1" ist. Nur wenn das Flag auf "*" gesetzt ist, wird tatsächlich auf die RAM-Daten zugegriffen.



Umgekehrt werden bei jedem Schreibzugriff auf das RAM auch die Flags aktualisiert.



Diese Idee wurde bereits in [3] und [4] vorgestellt. Dabei wurde gezeigt, dass man mit Hilfe einfacher arithmetischer Operationen unter Verwendung eines Addierers sämtliche benötigten Operationen auf diesen Flag-Vektoren realisieren kann. Im Rahmen der vorliegenden Arbeit wurde der Flag-Vektor zunächst in $2 * 66 = 132$ Flipflops gespeichert. Tatsächlich gelang es knapp, die komplette Schaltung einschließlich einer PCI-Schnittstelle auf einem XILINX FPGA XC4062 unterzubringen. Ohne PCI-Schnittstelle wurden dazu 2013 CLBs benötigt.
Leider traten aber bei manchen negativen Ergebnissen Rundungsfehler in der letzten Stelle der Mantisse auf. Prinzipiell wäre es nicht schwierig gewesen, diesen Fehler zu beheben. Allerdings hätte dazu der komplette Flag-Vektor nach eventuell vorkommenden Einsen durchsucht werden müssen. Bei n Flags wäre dies mit einem zusätzlichen Hardwareaufwand von $O(n)$ Gattern verbunden gewesen und dazu reichte die Kapazität des FPGAs nicht aus. Außerdem stellte sich diese Implementierung mit einer Taktrate von weniger als 1 MHz als recht langsam heraus.
Deshalb wurde nach einer besseren Vorgehensweise gesucht. Zunächst fiel wieder auf, dass es ähnlich wie beim Akkumulator ungünstig ist, den Flag-Vektor in Flipflops statt in RAMs zu speichern. Weiter ergab sich, dass sich die benötigten Such-Operationen auf dem Flag-Vektor am schnellsten mit Hilfe von **Baumstrukturen** durchführen lassen.
Im folgenden ist als Beispiel ein binärer Baum dargestellt, in dem jeweils zwei benachbarte Flags zu einer neuen Flag zusammengefasst sind. Will man beispielsweise für eine gegebene Position i ermitteln, wo sich die nächste "0" links von i befindet, so kann man einfach von i beginnend einen Pfad in Richtung zur Wurzel folgen, bis man eine linke Abzweigung mit "0" oder "*" findet. Diesem abzweigenden Pfad folgt man dann bis zur Zielposition k.

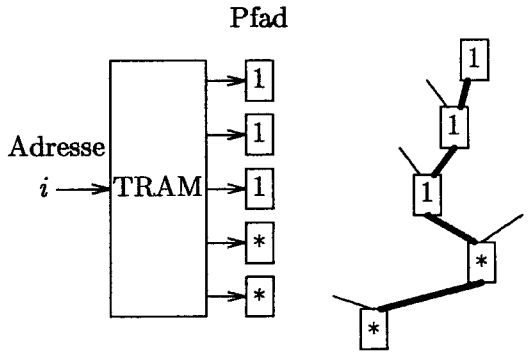


Es stellt sich heraus, dass man auf ähnliche Weise sämtliche für den Chip benötigten Operationen durchführen kann. Beispielsweise kann die Initialisie-

rung des kompletten Akkumulators auf "000...000" im Baum einfach dadurch geschehen, dass man das Wurzelelement auf "0" setzt.
Bei allen Lesezugriffen auf eine Adresse i müssen die Flags dann jeweils top-down propagiert werden, um eine konsistente Baumdarstellung zu erhalten. Umgekehrt müssen nach Schreibzugriffen die Flags bottom-up aktualisiert werden.
Im folgenden soll dargestellt werden, wie sich derartige binäre Bäume effizient in Hardware realisieren lassen.

3.3 Implementierung von Baumstrukturen

Besitzt der Baum n Blätter und eine Höhe von $\log_2 n$, so können die benötigten Suchoperationen per Software in Zeit $T = O(\log n)$ ausgeführt werden. In Hardware kann man das gleiche gute Zeitverhalten erreichen. Interpretiert man eine Softwarelösung als Simulation einer Schaltungsbeschreibung, so ergibt sich, dass tatsächlich nur $O(\log n)$ Gatter an dem Suchvorgang beteiligt sind. Dies sind gerade die Gatter, die auf den Suchpfaden liegen.
Deshalb sollte man den oben dargestellten Baum stufenweise durch RAMs darstellen, so dass man beispielsweise durch Anlegen einer Adresse i sämtliche auf dem Pfad von der Wurzel zum Blatt i liegenden Knoten erhält. Eine solche Speicherung einer Baumstruktur soll im folgenden als **TRAM** (tree-RAM) bezeichnet werden.



Diese Schaltung besteht im wesentlichen nur aus RAMs und erlaubt den direkten Zugriff auf Baumpfade. Eine darauf aufbauende Anwendungsschaltung "sieht" nicht mehr alle n Flags des Flag-Vektors, sondern nur noch $\log_2 n$ relevante Flags des Baumes. Dadurch reduziert sich der Hardwareaufwand erheblich.
An dieser Stelle soll auf Implementierungsdetails verzichtet werden. Der wesentliche Punkt besteht darin, dass die n Flags in RAMs mit einer Kapazität

von $O(n)$ Bits gespeichert werden und die für die Kulisch-Anwendung erforderliche kombinatorische Auswerteschaltung statt bisher $O(n)$ Gatter nur noch $O(\log n)$ Gatter benötigt.
Obwohl zur Vereinfachung der Implementierung statt 66 Flags jetzt $2^7 = 128$ Flags verfügbar sind, reduziert sich die gesamte Schaltungsgröße von über 2013 CLBs auf nur noch 1617 CLBs. Dies zeigt deutlich das Einsparpotential durch die Verwendung von RAM-Speichern und die damit verbundenen Reduzierung kombinatorischer Schaltungsteile.

3.4 Arbeitsweise in zwei Takten

Mit Hilfe des Flag-Systems ist es tatsächlich möglich, sämtliche Arbeitsschritte der Addition in zwei aufeinanderfolgenden Takten auszuführen.
Im ersten Takt wird die Mantisse des neuen Summanden zum aktuellen Akkumulator addiert und gleichzeitig wird geprüft, wie weit man gegebenenfalls im Falle eines Übertrags in Richtung der Baumwurzel laufen muss, um eine Kette zu überspringen. Sofern tatsächlich ein Übertrag nicht lokal verbucht werden kann, werden die Flags dieses aktuellen Pfades zum Umkippen der Kette modifiziert und im zweiten Takt wird dieser Vorgang über einen zweiten Pfad zum Ende der Kette vervollständigt. Gleichzeitig wird dort der Übertrag durch eine lokale Addition verbucht.
Das Auslesen des aktuellen Akkumulators im Format einer `double`-Zahl geschieht dadurch, dass im ersten Takt über das Flag-System nach dem führenden relevanten Bit gesucht wird und im zweiten Takt der entsprechende Akkumulatorbereich in eine korrekte Mantisse überführt wird. Die Position der Mantisse im Akkumulator entspricht dann dem Exponenten der Zahldarstellung.

4 Ergebnisse und Ausblick

Das Ziel, eine kleine Schaltung zu erzielen, die jede Operation in nur zwei Takten ausführt, wurde erreicht. Damit wurde demonstriert, dass man eine komplexe Aufgabe, wie die Kulisch-Arithmetik, auch auf relativ kleinen FPGAs bearbeiten kann, ohne den Algorithmus zu serialisieren.
Als maximal mögliche Taktrate wurde im praktischen Einsatz etwas mehr als 6 MHz ermittelt. Dies ist eine drastische Beschleunigung gegenüber der ursprünglichen Schaltungsversion ohne Baumstrukturen. Diese Beschleunigung wurde allein durch die Struktur der Schaltung erzielt und beruht nicht auf den Fähigkeiten einer optimierten Schaltungssynthese. Tatsächlich wurde die

Schaltung nicht auf Zeit, sondern nur auf Fläche optimiert, weil die Synthesetools bei Verwendung des PCI-Interfaces keine Timing-Constraints zuließen. Bei der Untersuchung des kritischen Pfades stellte sich heraus, dass etwa 50% des Zeitbedarfs auf das Flag-System entfallen. Da hier die Pfadlängen mit der Höhe des Baumes wachsen, lässt sich das Zeitverhalten der Schaltung wahrscheinlich weiter verbessern, wenn man den binären Baum durch eine Baumstruktur mit einem höheren Grad ersetzt und so die Baumhöhe reduziert. Entsprechende Experimente sollen demnächst durchgeführt werden.
Zusätzliche Optimierungsmöglichkeiten liegen beim verwendeten Addierer. Hier wurden der Einfachheit halber recht langsame Carry-Ripple-Adder der Länge 32 Bit verwendet, die natürlich durch schnellere Carry-Look-Ahead Addierer ersetzt werden können.

Die für die Kulisch-Arithmetik gefundene Lösung beruht wesentlich auf der Idee, Baumstrukturen in RAMs abzulegen. Die dazu verwendete TRAM-Implementierung hat sich inzwischen durch weitere Experimente als vielseitig verwendbar erwiesen. Sie eignet sich beispielsweise auch für Binärzähler, Gray-Code-Zähler und Sortierverfahren. Als Nebeneffekt der Hardwareminimierung ist dabei auch eine Reduktion der Leistungsaufnahme zu erwarten. Deshalb scheint eine genauere Untersuchung der Anwendungsmöglichkeiten von TRAMs interessant zu sein.

Literatur

[1] P.R. Cappello, W.L. Miranker, *Systolic super summation*, IEEE Trans. Comput. 37 (1988) 657-676.

[2] R. Kirchner, U. Kulisch, *Accurate arithmetic for vector processors*, J. Parallel Distrib. Comput. 5 (1988) 250-270.

[3] M. Müller, Chr. Rüb, W. Rülling, *Exact Accumulation of Floating-Point Numbers*, Proceedings of 10th IEEE Symposium on Computer Arithmetic, Grenoble, France, IEEE Computer Society Press, Los Alamitos, California (1991) 64-69.

[4] M. Müller, Chr. Rüb, W. Rülling, *A circuit for exact summation of floating-point numbers*, Information Processing Letters 57 (1996) 159-163.

[5] R.J.W.T. Tangelder, *The design of chip architectures for accurate inner product computation*, Ph.D. Thesis, Eindhoven university of Technology (1992).

BOSCH

Entwurfsmethodik mikromechanischer Sensoren

Robert Bosch GmbH Gerlingen-Schillerhöhe
Zentralbereich Forschung und Vorausentwicklung
Abteilung Mikrosystemtechnik und dünne Schichten

Dr.-Ing. Dietmar Krieg

FV/FLD-Krieg, Vortrag XXXII. MPC-Workshop in Albstadt-Sigmaringen am 09.07.2004 folien_040709_kg_mpc_v03.ppt

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Entwurfsmethodik mikromechanischer Sensoren
Dr.-Ing. Dietmar Krieg, Robert Bosch GmbH

Beim Entwurf mikromechanischer Sensoren sind aufgrund der starken Verzahnung von Mechanik, Elektronik und Prozesstechnologie rein intuitive Vorgehensweisen häufig weniger zielführend. Am Beispiel eines mikromechanischen Drehratensensors soll gezeigt werden, wie der Entwicklungsprozess durch Simulationen begleitet werden kann. Dabei kommen verschieden stark detaillierte Modelle zum Einsatz: Finite-Elemente-Berechnungen, Netzwerk- und Systemsimulationen.

➔ **Einführung in mikromechanische Sensoren bei Bosch**

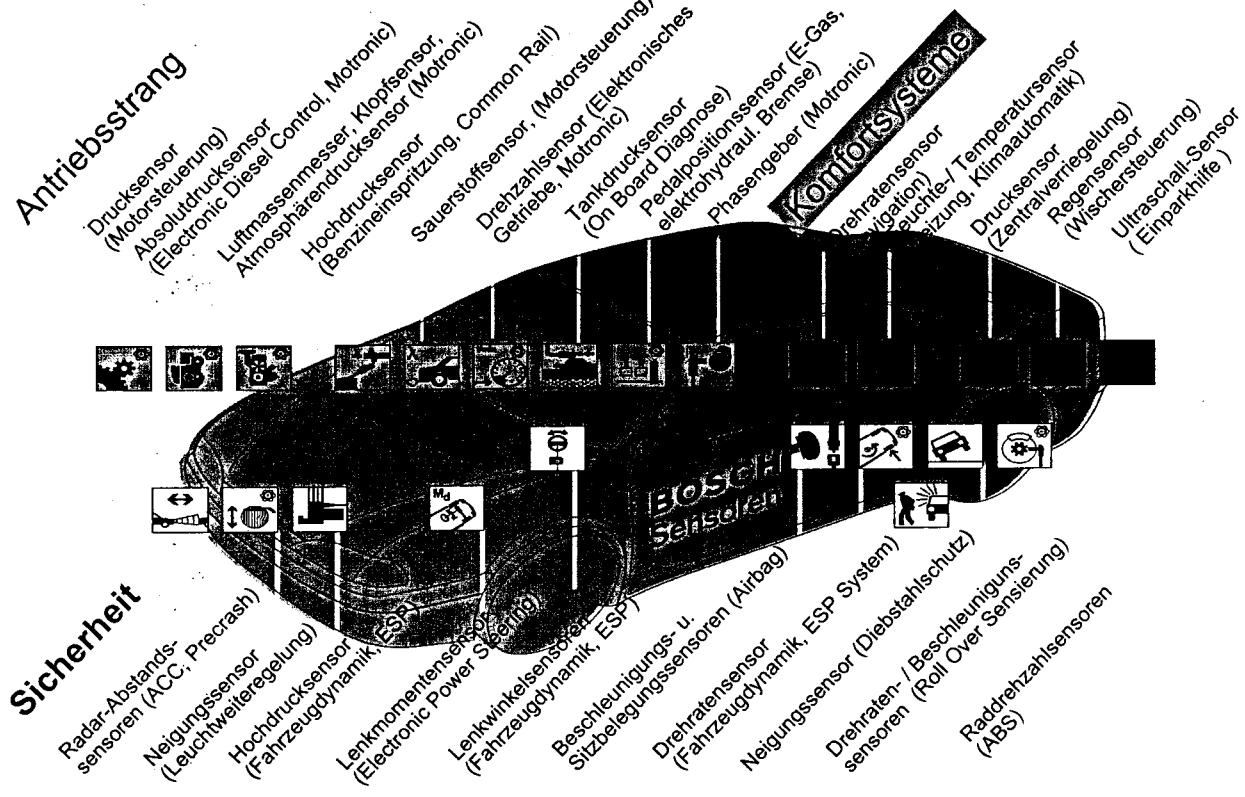
Funktionsprinzip des Drehratensensors

Simulation des Drehratensensors

Modellierungskonzepte und Durchgängigkeit

Zusammenfassung

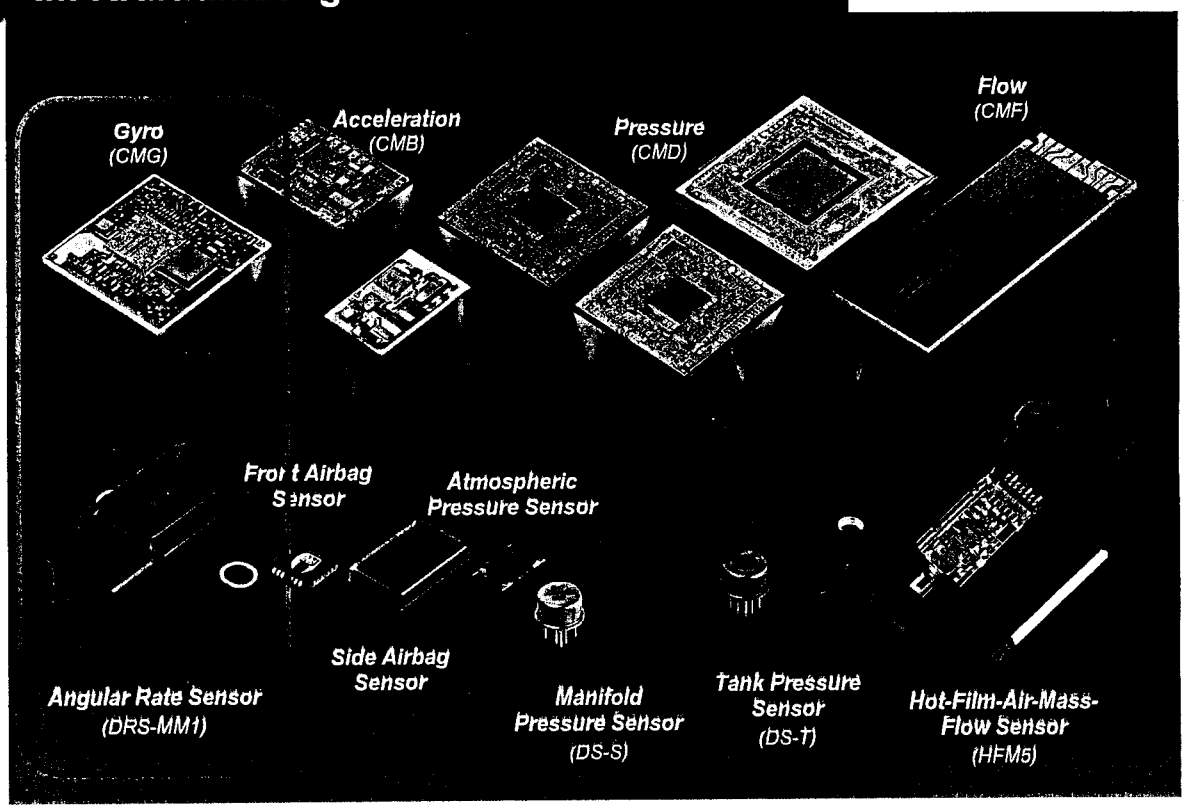
© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.



© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Mikrosystemtechnik-Sensoren im Kraftfahrzeug

BOSCH



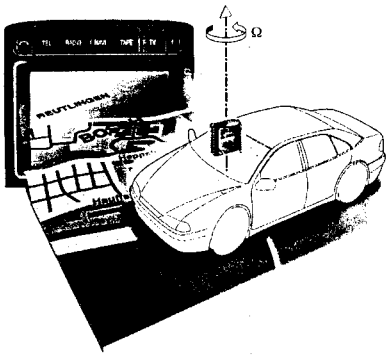
© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Motivation Drehratensensoren

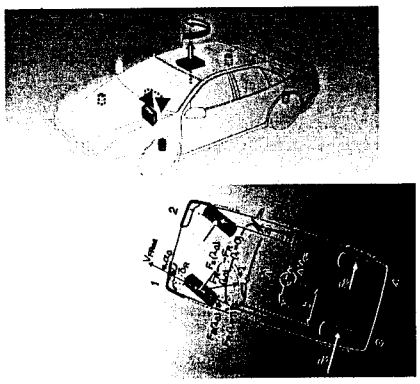
BOSCH

Einsatzgebiete des Drehratensensors

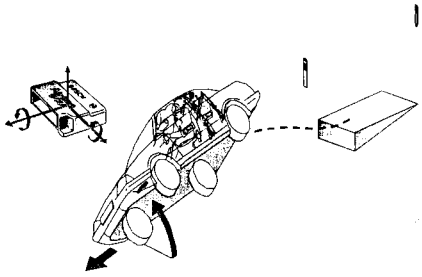
→ **Navigation**
typ. Auflösung: < 0,5 °/s



→ **Fahrzeugdynamik (ESP)**
typ. Auflösung: < 0,2 °/s



→ **Überroll-Detektion**
typ. Auflösung: 1,5°/s



© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Einführung in mikromechanische Sensoren bei Bosch

➔ **Funktionsprinzip des Drehratensensors**

Simulation des Drehratensensors

Modellierungskonzepte und Durchgängigkeit

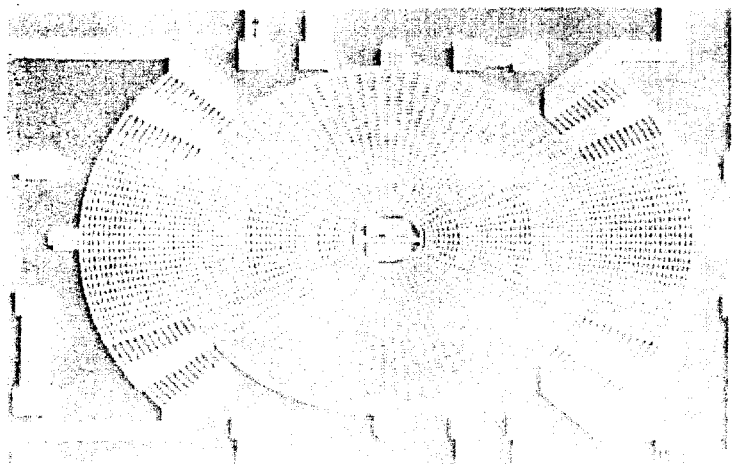
Zusammenfassung

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Entwurfsmethodik
mikromechanischer Sensoren

BOSCH

REM-Bild eines Drehratensensors

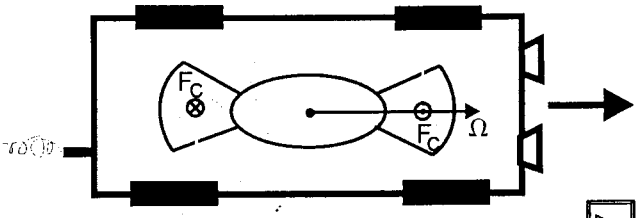


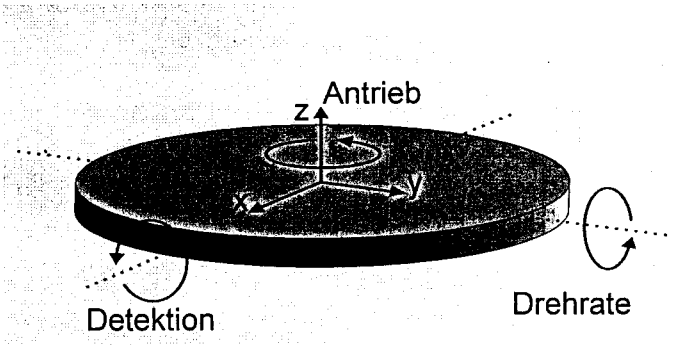
Wie kommt man auf so eine Idee?

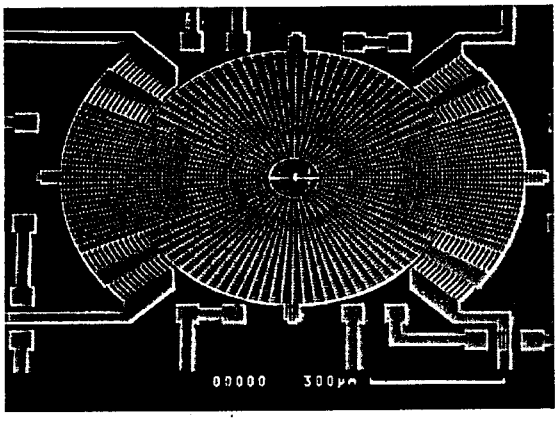
© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

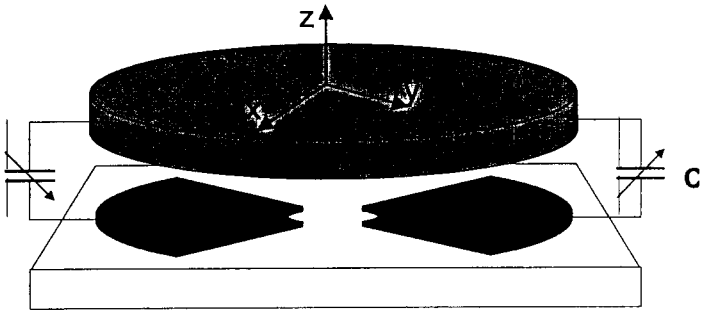
Funktionsprinzip des Drehratensensor

BOSCH







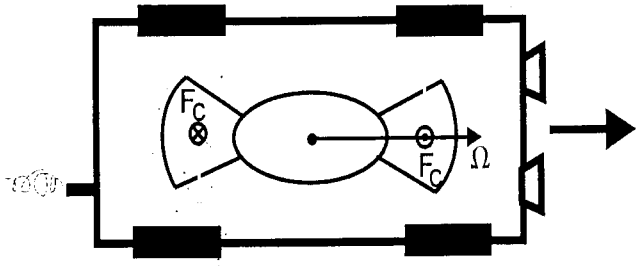


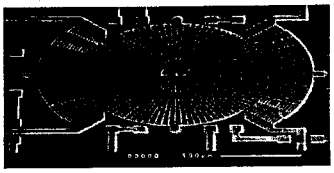
© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

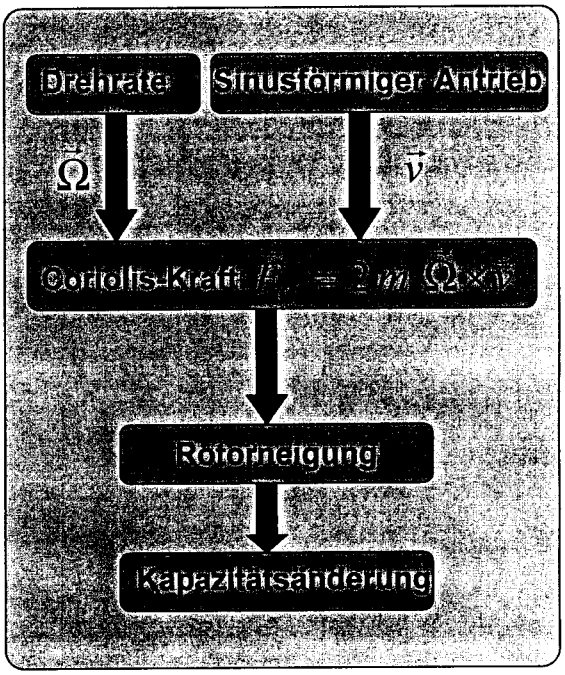
Funktionsprinzip des Drehratensensor

BOSCH

Physikalische Grundlage des Messprinzips: Corioliseffekt



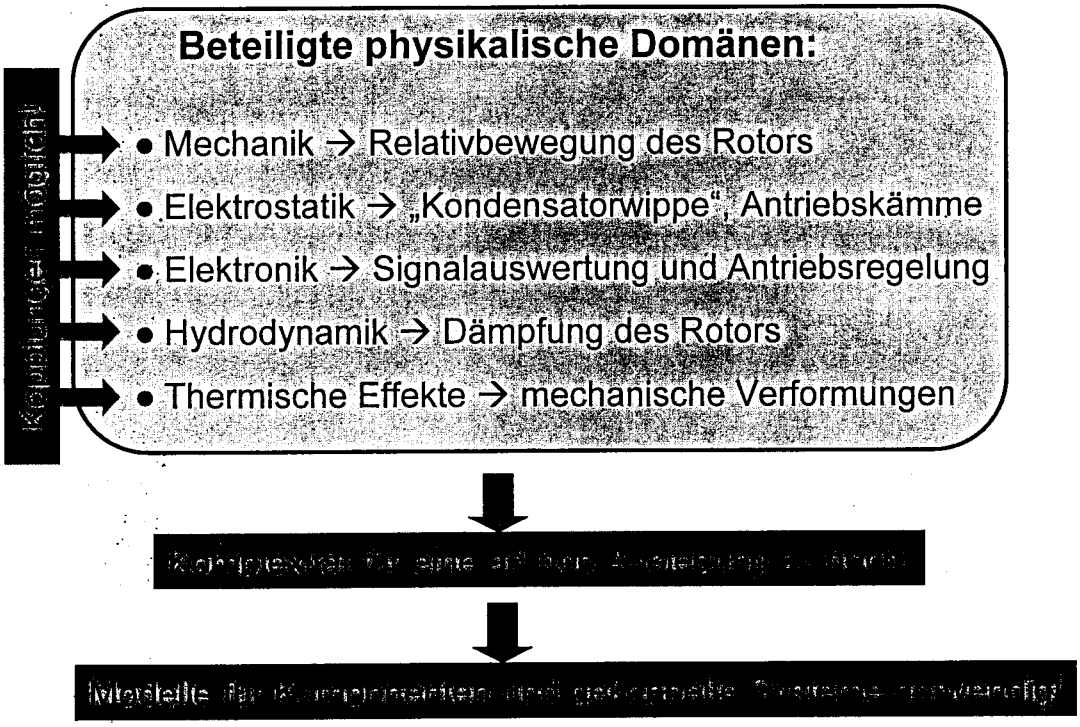




© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Funktionsprinzip des Drehratensensor

BOSCH



© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Entwurfsmethodik mikromechanischer Sensoren

BOSCH

Renaissance der „analogen" Experten

„Why analog is cool again?" (Quelle: Wired)

- **Mobilfunk (UMTS, WLAN)**
- **Settop-Boxen für digitales Antennenfernsehen**
- **Medizintechnik**
- **Hörgerate**
- **Telematik**
- **Kfz-Maut**
- **Auto-Radar**
- **Sensorik**

Quelle: VDI-Nachrichten Nr. 25 vom 18.07.2004: Seite 32

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Entwurfsmethodik mikromechanischer Sensoren

BOSCH

Einführung in mikromechanische Sensoren bei Bosch

Funktionsprinzip des Drehratensensors

→ **Simulation des Drehratensensors**

Modellierungskonzepte und Durchgängigkeit

Zusammenfassung

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Simulation des Drehratensensors: Zerlegung in bekannte Teilprobleme

BOSCH

Evtl. gekoppelte Behandlung erforderlich!

- **Sensor im fahrenden Auto**

➡ **Relativbewegung starrer Körper (Kinematik)**

- **Biegebalken als Rückstellfeder ➡ Balkenstatik**

- **Verformungen des Rotors (Stresseinkopplung)**

➡ **Statik ➡ FEM-Rechnung**

- **Eigenschwingungen des Rotors ➡ Elasto-Kinetik**

- **Taumelbewegung des Rotors**

➡ **„Kreiselgleichungen“ (Kinetik)**

- **Kammantrieb ➡ Elektrostatik**

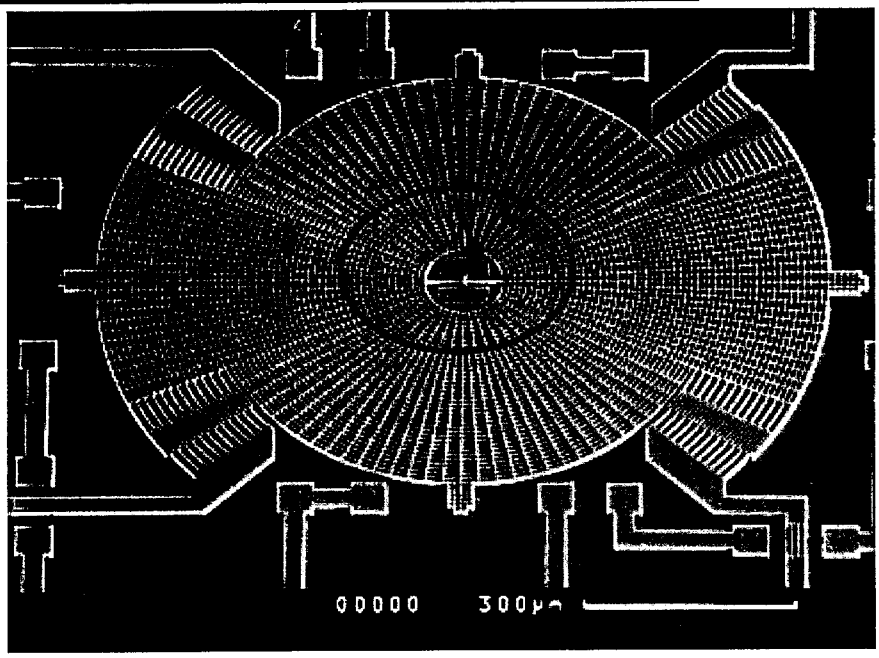
- **Auswertekondensatoren ➡ Elektrostatik**

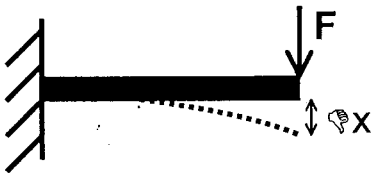
- **Auswerteelektronik ➡ Regelungstechnik**

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Simulation des Drehratensensors: Biegebalken als Rückstellfeder

BOSCH





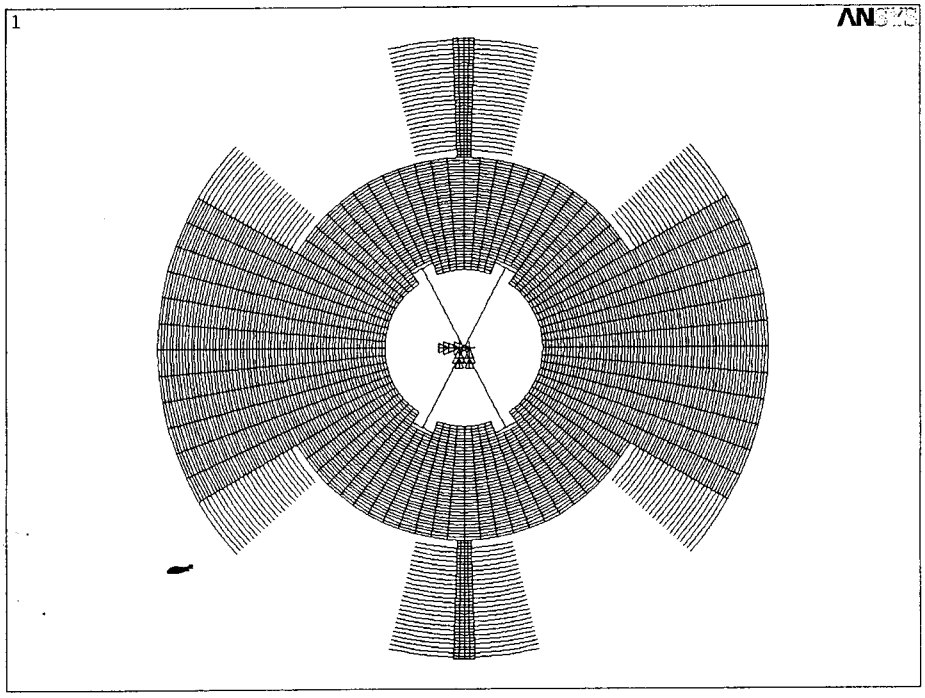
$$\Delta x = \frac{F \cdot l^3}{3E \cdot I} \qquad k_{Feder} \approx \frac{F}{\Delta x}$$

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Simulation des Drehratensensors

BOSCH

Finite-Element-Modell eines Drehratensensors



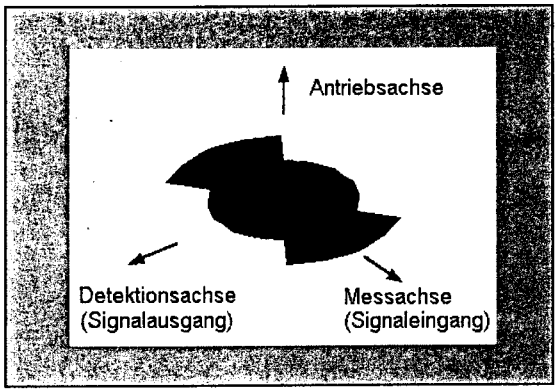
Quelle: FV/FLD-Papathanassiou

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

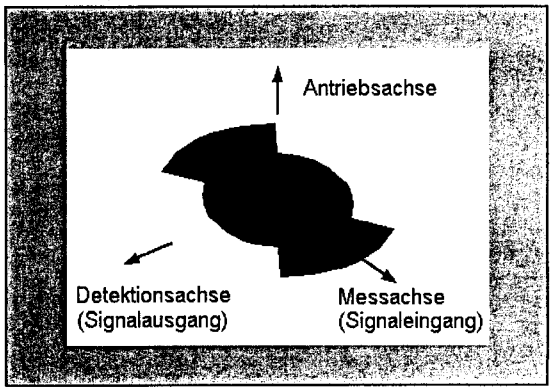
Simulation des Drehratensensors

BOSCH

Taumelbewegung des Rotors
➡ „Kreiselgleichungen“ (Kinetik)



Ohne Drehrate



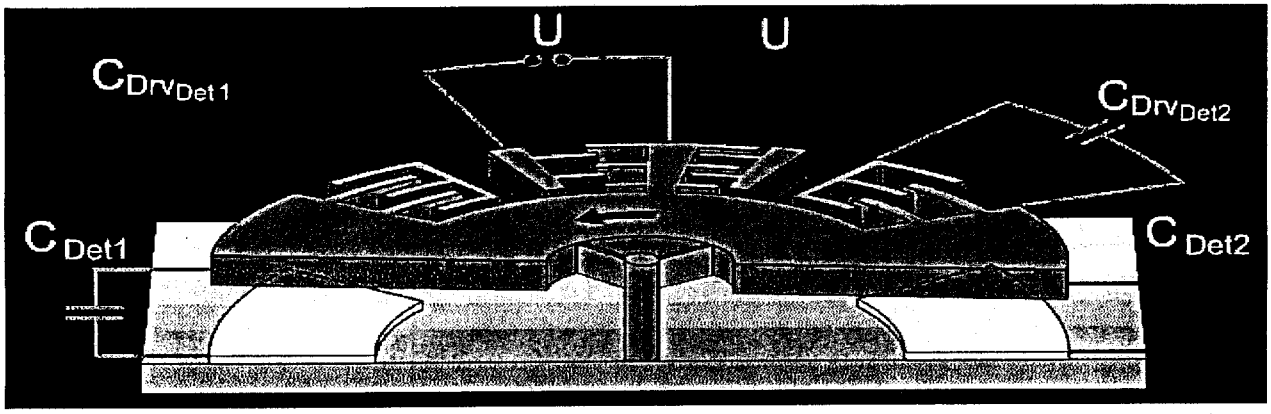
Mit Drehrate

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Simulation des Drehratensensors

BOSCH

Antriebsprinzip: Kondensatorkämme

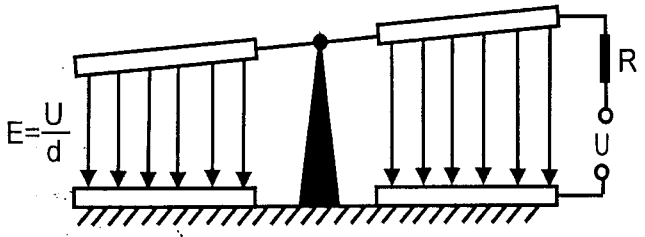


© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Elektrostatikmodellierung: „Plattenkondensatorwippe“

Grundüberlegung: statische oder dynamische elektrische Effekte?

Elektrische Vorgänge laufen wesentlich schneller ab
als mechanische Vorgänge → Quasistatische Betrachtung genügt!



Differenzkapazität:

$$\Delta C(\alpha) = \frac{\varepsilon_r \cdot \varepsilon_0 \cdot A}{d_0 - r_m \cdot \tan\alpha} - \frac{\varepsilon_r \cdot \varepsilon_0 \cdot A}{d_0 + r_m \cdot \tan\alpha}$$

Elektrostatisches Moment:

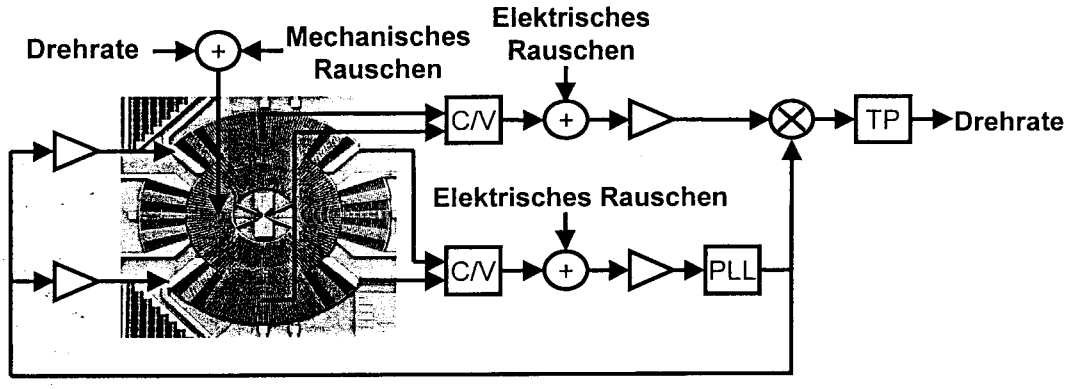
$$F_{el} = \frac{1}{2} C \cdot \frac{U^2}{d}$$

$$M_{el} = (F_{el,1} - F_{el,2}) \cdot r_m$$

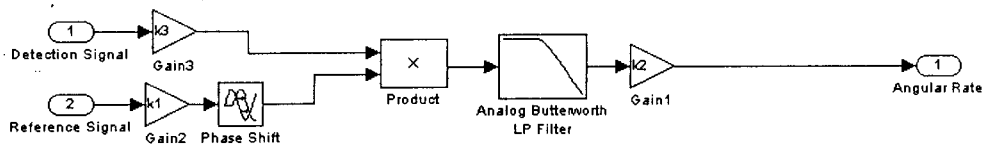
© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Elektronikmodellierung

Beispiel: Signalmodulation und -demodulation



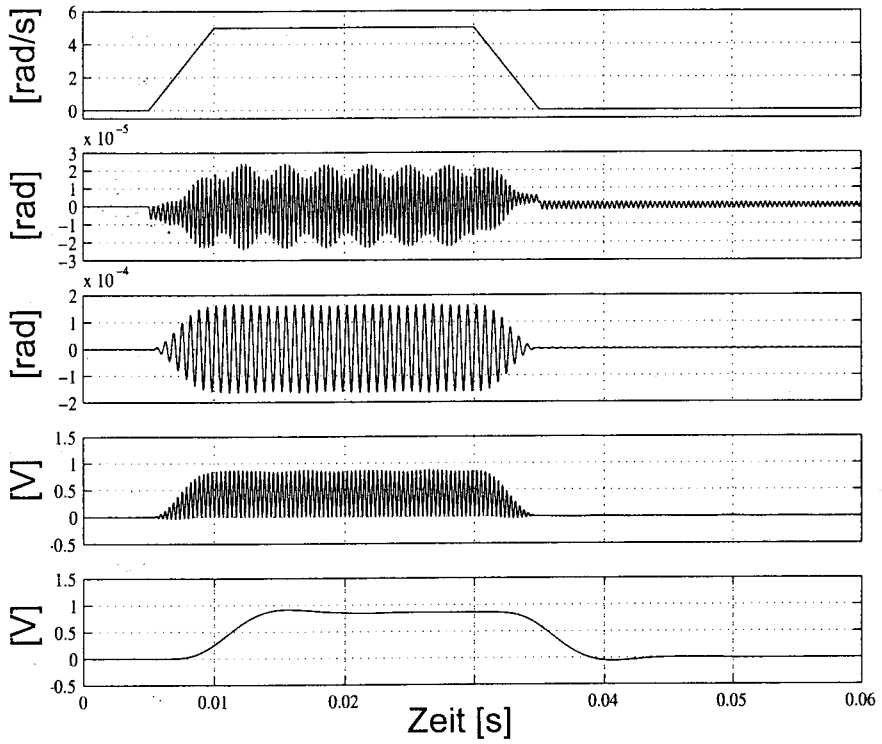
Vereinfachtes Simulink-Modell:



© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Simulation des Drehratensensors: Signalmodulation und -demodulation

BOSCH



Drehrate um die y-Achse

Auslenkung des Rotors um die y-Achse

Auslenkung des Rotors um die x-Achse

Signal nach dem Synchron-demodulator

Drehratensignal nach dem Tiefpassfilter

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Entwurfsmethodik mikromechanischer Sensoren

BOSCH

Einführung in mikromechanische Sensoren bei Bosch

Funktionsprinzip des Drehratensensors

Simulation des Drehratensensors

➔ **Modellierungskonzepte und Durchgängigkeit**

Zusammenfassung

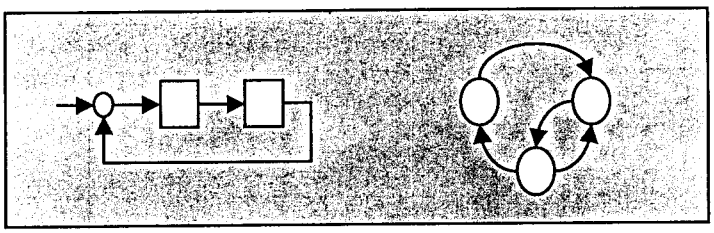
© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Modellierungskonzepte und Durchgängigkeit

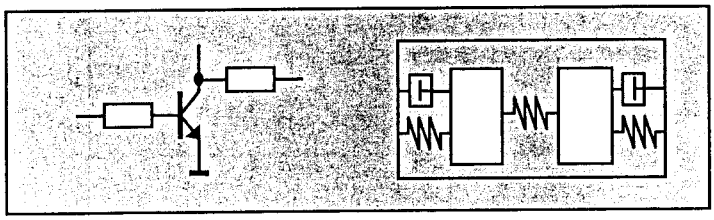
BOSCH

zunehmender Abstraktionsgrad

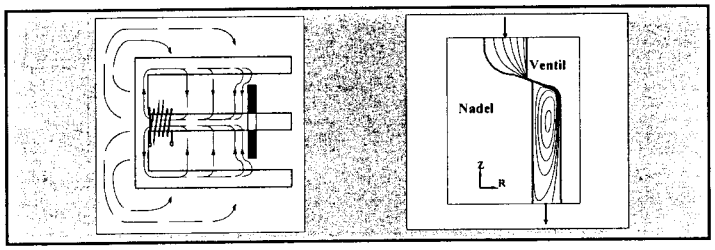
System-ebene



Netzwerk-ebene



Geometrie-ebene



Tools:

Matlab
Simulink
Mathematica
Maple

Saber
Spice
20Sim
VHDL-AMS:
HAMSTER
(SMASCH)

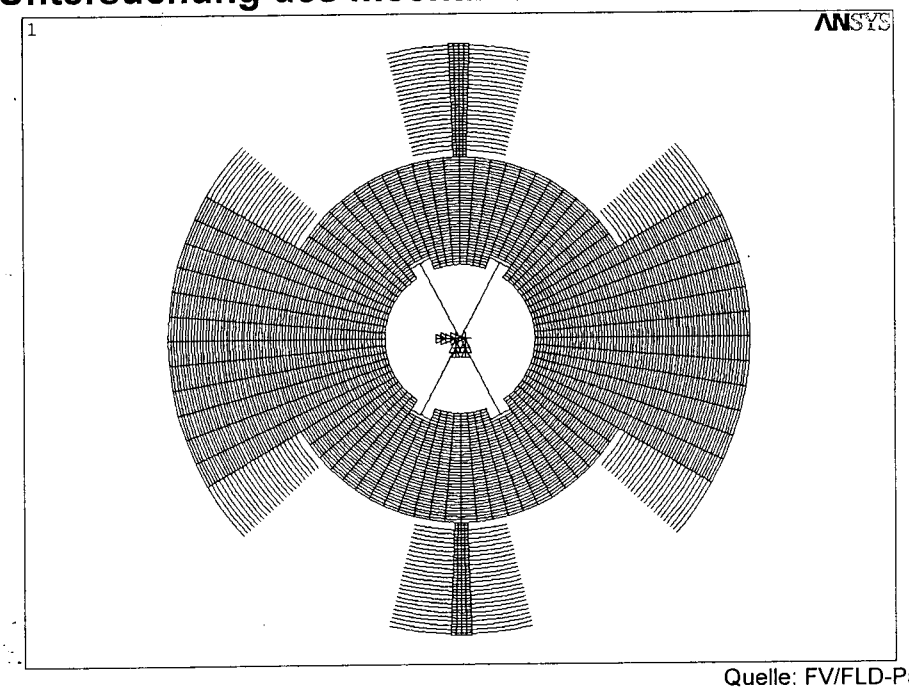
ANSYS
(Multi
Physics
License)

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Modellierungskonzepte: Geometrieebene

BOSCH

Finite-Elemente-Modell des Drehratensensors zur Untersuchung des mechanischen Verhaltens

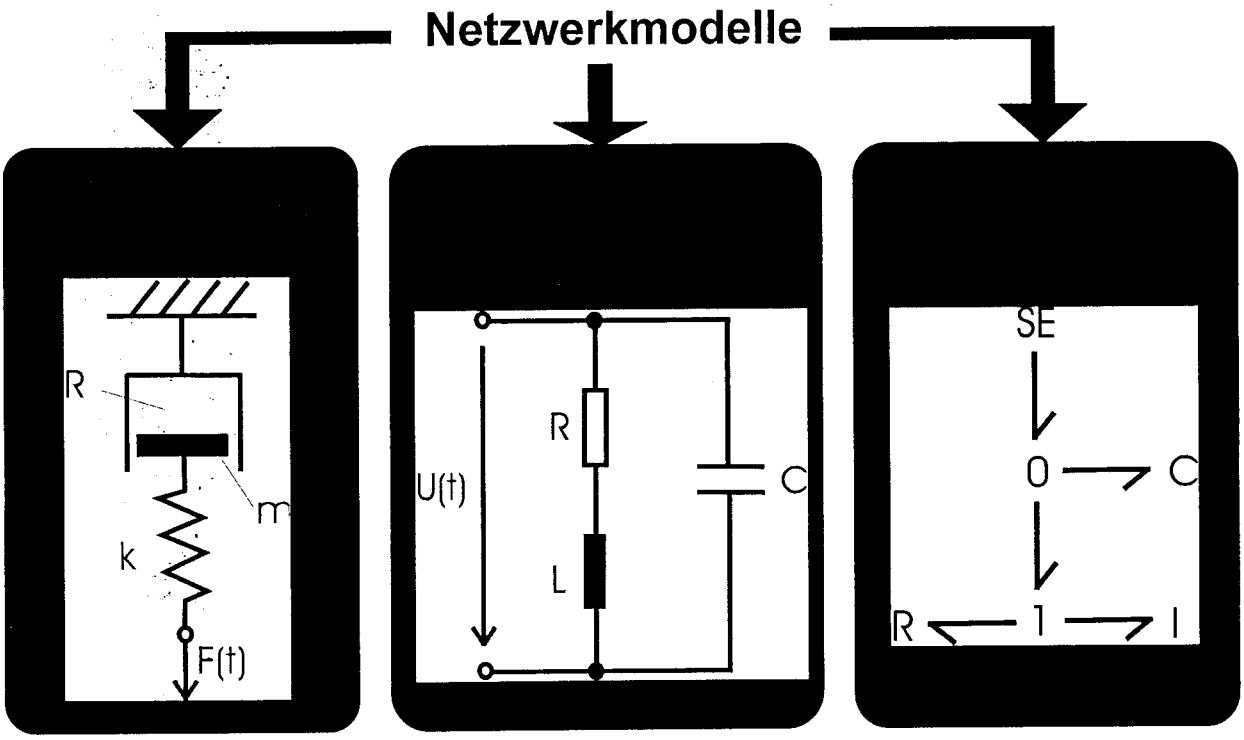


Quelle: FV/FLD-Papathanassiou

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Modellierungskonzepte: Netzwerkebene

BOSCH

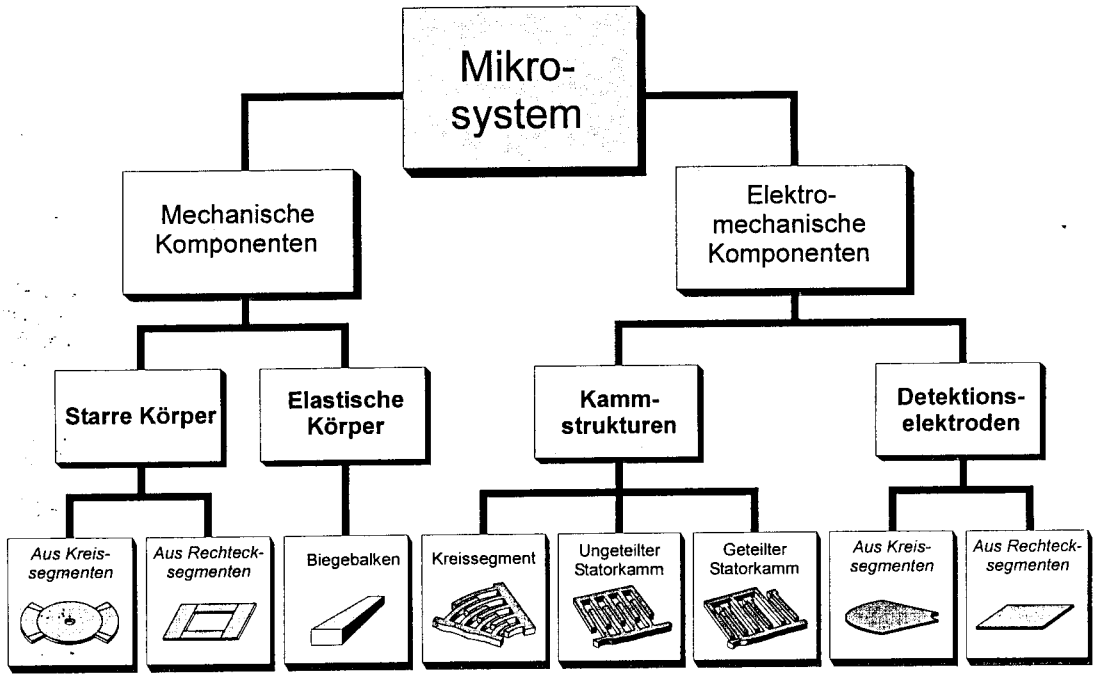


© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Modellierungskonzepte: Netzwerkebene

BOSCH

Mechatronik-Bibliothek auf der Netzwerkebene



© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Einführung in mikromechanische Sensoren bei Bosch

Funktionsprinzip des Drehratensensors

Simulation des Drehratensensors

Modellierungskonzepte und Durchgängigkeit

→ **Zusammenfassung**

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

Entwurfsmethodik
mikromechanischer Sensoren

BOSCH

Zusammenfassung:

- **In der MST sind meist mehrere pysikalische Domänen beteiligt**
- **Automatisierungsgrad des Systementwurfs ist geringer als in der Elektronik**
- **Simulationstools sind im Gegensatz zur Elektronikentwicklung weniger stark auf die Anwendungen in der MST abgestimmt**

→ **Aufbau eines durchgängigen Modellierungskonzepts mit drei Modellierungsebenen:**

- **Systemebene**
- **Netzwerkebene**
- **Geometrieebene**

→ **Erstellung MST-spezifischer Modellbibliotheken**

© Alle Rechte bei Robert Bosch GmbH, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

THOMSON TECHNICOLOR grassvalley RCA THOMSON

Entwicklung und Implementierung moderner Hochfrequenzschaltungen am Beispiel eines 5 GHz-WLAN-Transceivers

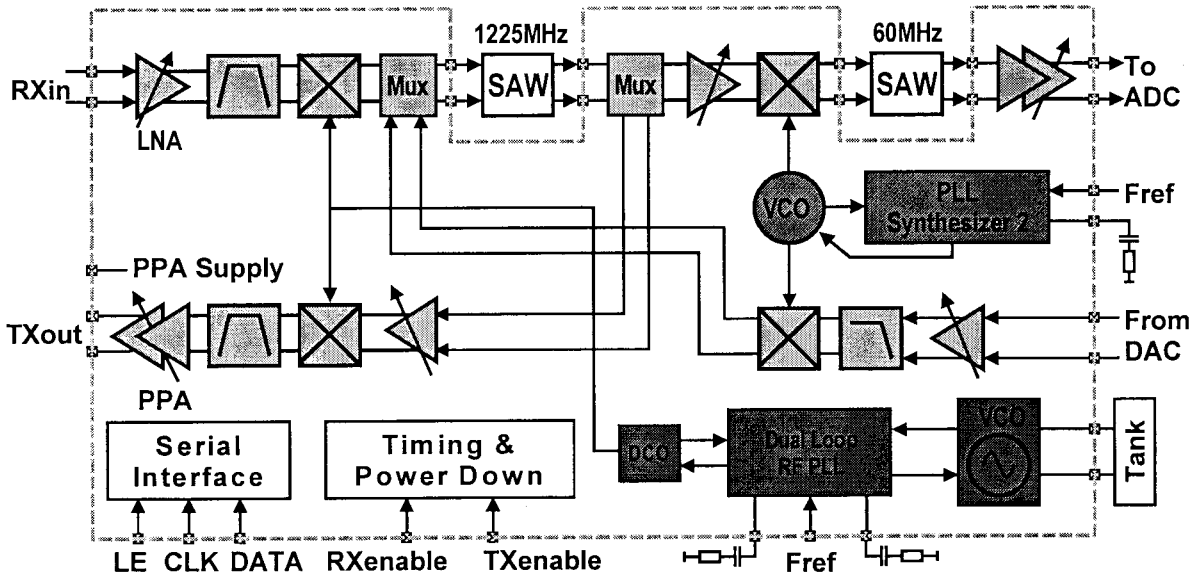
Heinrich SCHEMMANN
Deutsche Thomson Brandt GmbH
Villingen-Schwenningen, DE
email: heinrich.schemmann@thomson.net

Corp. Technology Group
Silicon Components

Outline

- Overview: Single Chip RF Transceiver IC
- *Transmit Path*
  - *Characteristics, Power Amplifier, Design Aspects*
- *Dual Loop RF PLL*
  - *Motivation, Solutions, Results*
- *Fast Digital AGC Concept*
- Silicon Evaluation
- RF ICs: Design Challenges & Specialities
- Chip Micrograph, Conclusion

Block Diagram of the Transceiver

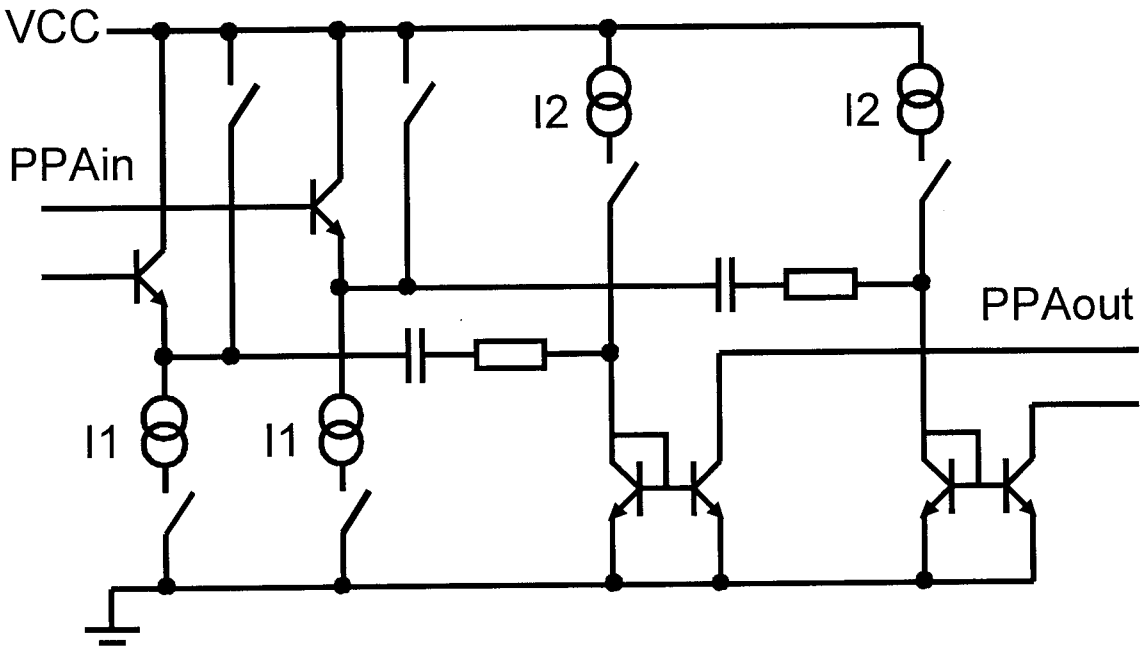


for the standards: HIPERLAN2 and IEEE 802.11a
frequency range covered: 4.90 ... 5.825 GHz

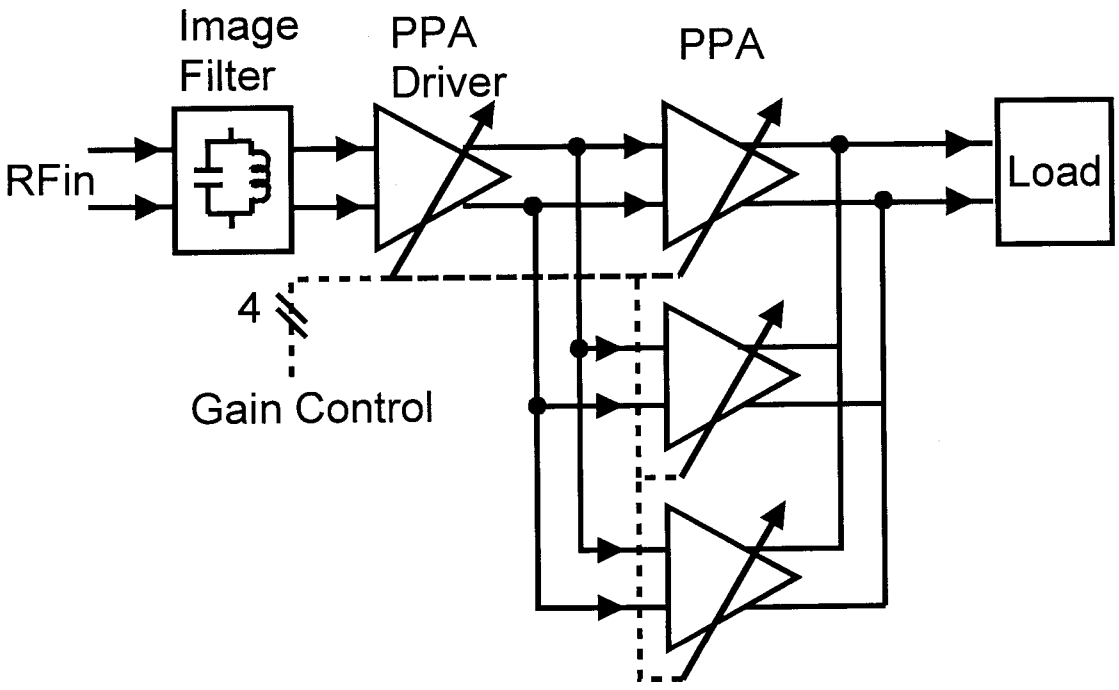
Transmit Part Characteristics

- 50dB of programmable gain range
  - 4dB in IF path
  - 46dB in RF path
- Output 1dB compression: +15dBm
- 12dB sideband rejection on-chip
- LO feed through: < –30dBm
- Spur levels below –32dBm

Power Amplifier Cell Design



Power Amplifier Concept



Power Amplifier Challenges

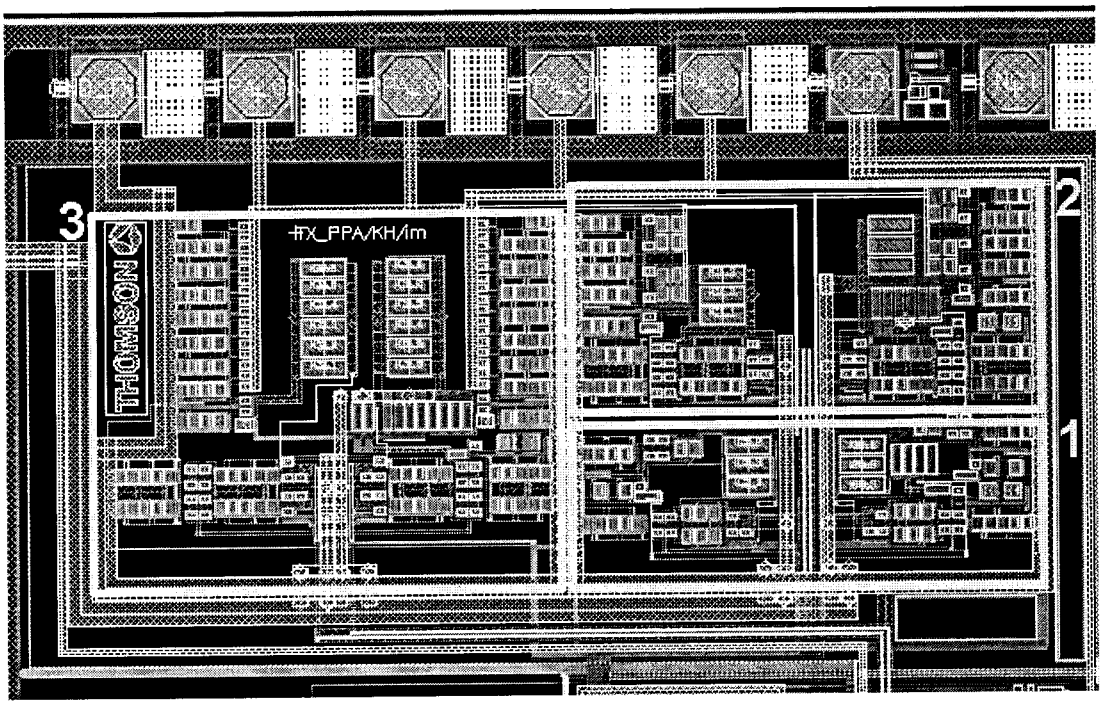
- Compensation of on-chip interconnect parasitics:
  - Insertion of Buffers to drive the interconnections
- Increased intermodulation due to parasitics
  - Design requires coupled R-L-C extraction
- Power gain variations due to parasitics and process tolerance
  - Compensated using the fast digital AGC concept

Power Amplifier Performance

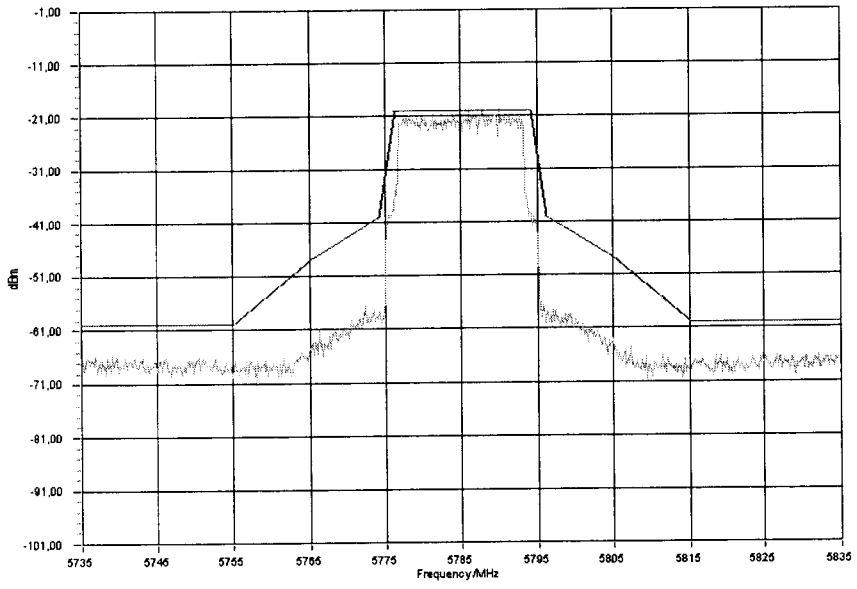
- PPA Driver provides 10dB of gain step
- PPA provides two gain steps with reduced linearity and power consumption accordingly

	High Gain	Mid Gain	Low Gain	Unit
Normalized Gain	0	- 8.5	- 16.5	dB
Output 1dB compression	+ 15	+ 7	- 1	dBm
Total Power Consumption	1130	940	810	mW

Triple PPA – Cell Layout



TX Spectrum with OFDM signal

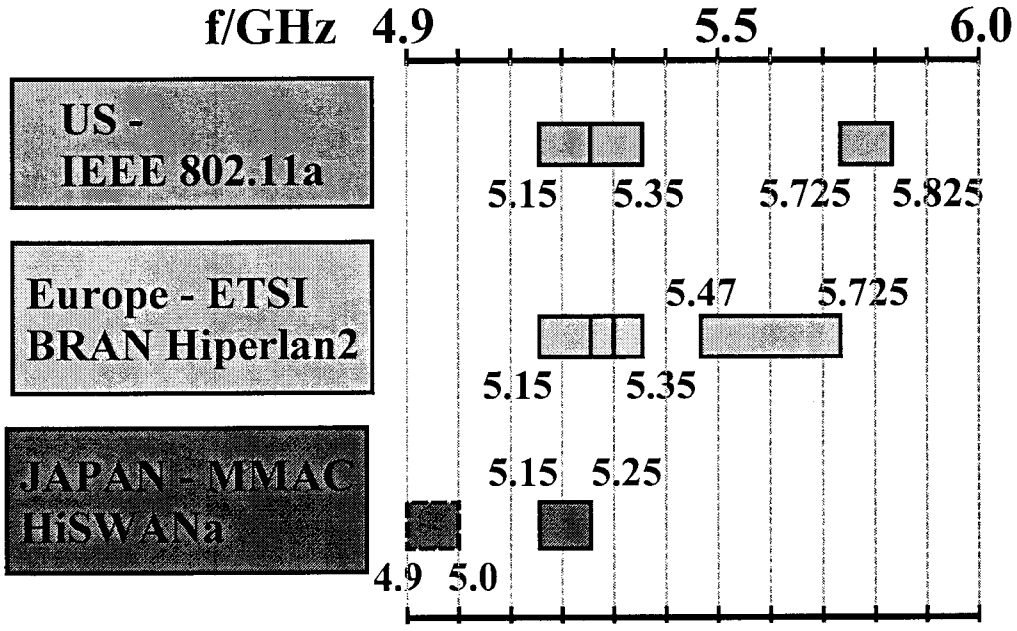


TX Spectrum:
Center
Frequency
5.785 GHz

Spectral Mask
(specified)

Measured

WLAN - RF bands used

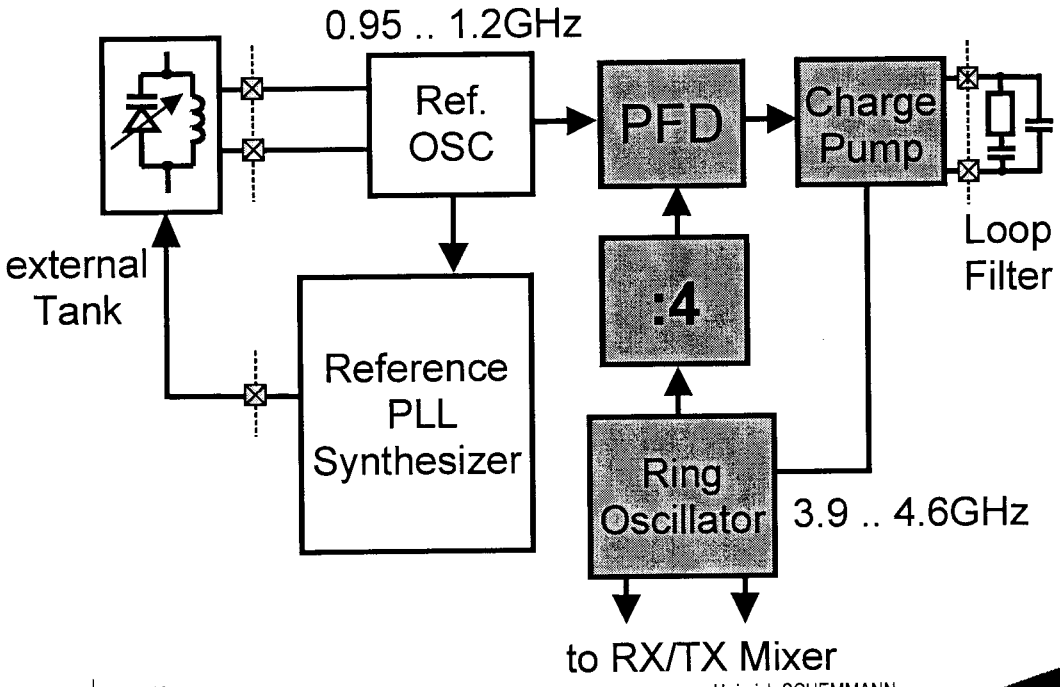


=> A wide range LO & PLL are required

Solutions for wide band LOscillator

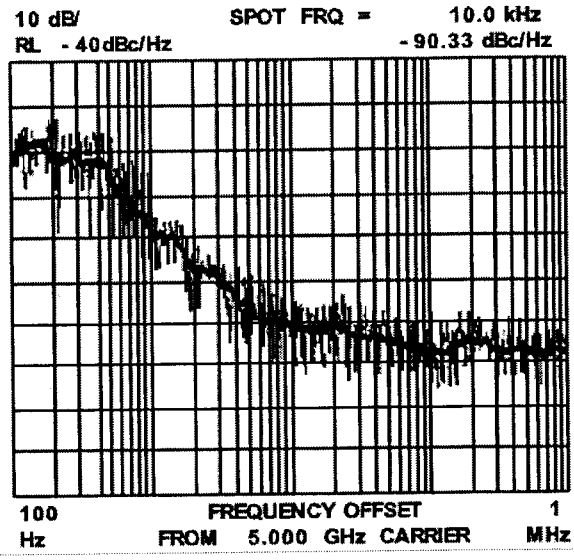
- NEED: good Q for phase noise PLUS wide tuning range
  - Single LC-VCO with L & Varactor on chip
    - Issue: varactor has too small C-ratio, tolerances
  - Shunt capacitors switched to LC with MOS
    - Limits: Switch parasitics, variation of Gain & Q
  - Multiple LC oscillators on chip
    - Autocalibration to secure range overlap
    - Silicon area large
  - External: neither simple nor cheap nor ambitious
  - ...

Dual Loop RF PLL

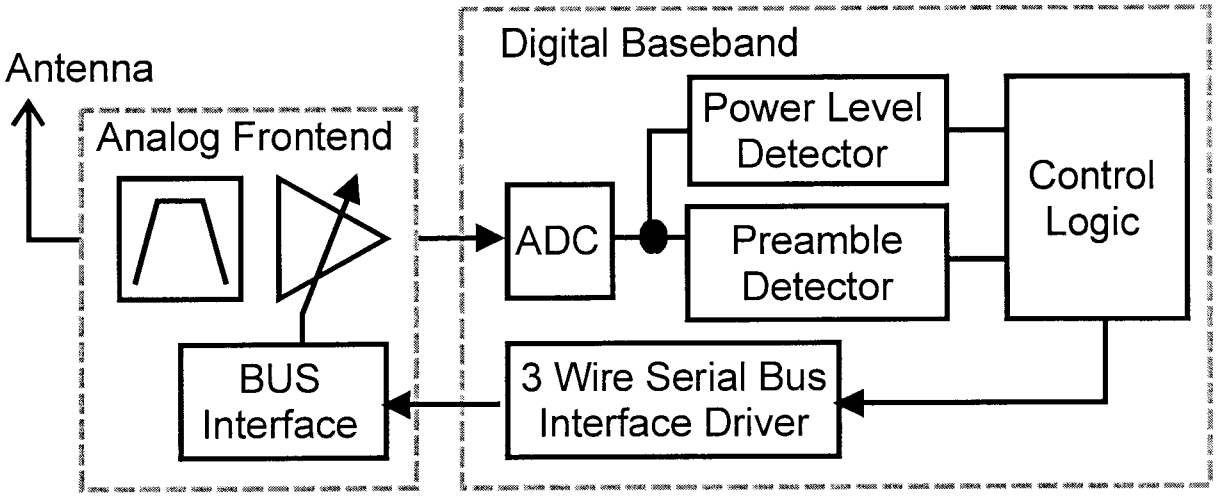


Dual Loop RF PLL Performance

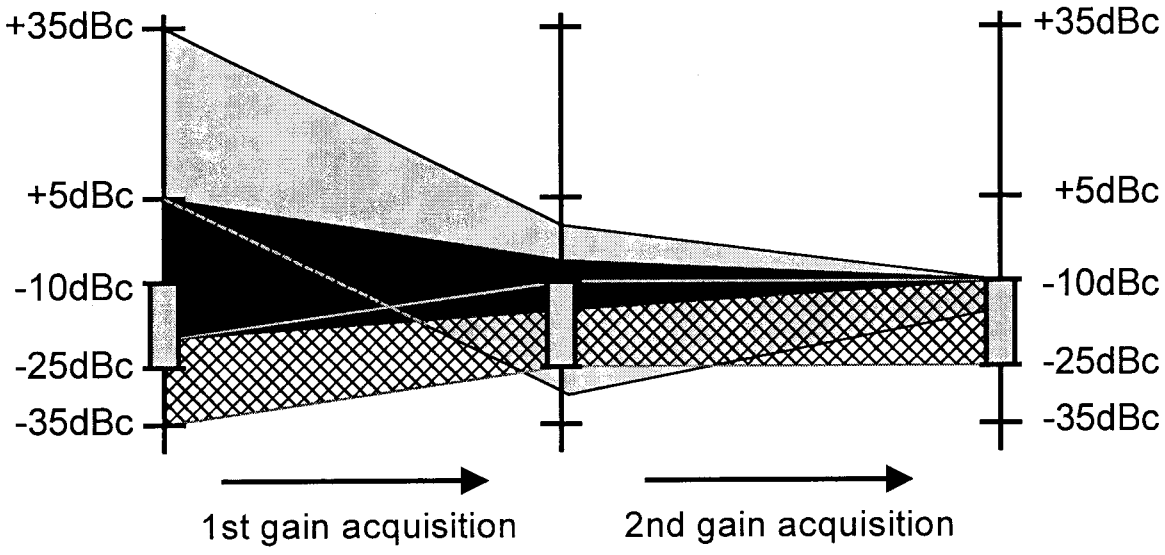
- 1.3GHz of programmable frequency range
  - measured range: 3.630 .. 4.905GHz
  - equivalent RF freq. range: 4.885 .. 6.130GHz
  - With 3V tuning voltage range: 5.10 to 5.89GHz
- PLL step size: 5MHz
- RF Loop Phase noise: - 90dBc/Hz @ 10kHz for LO=4GHz



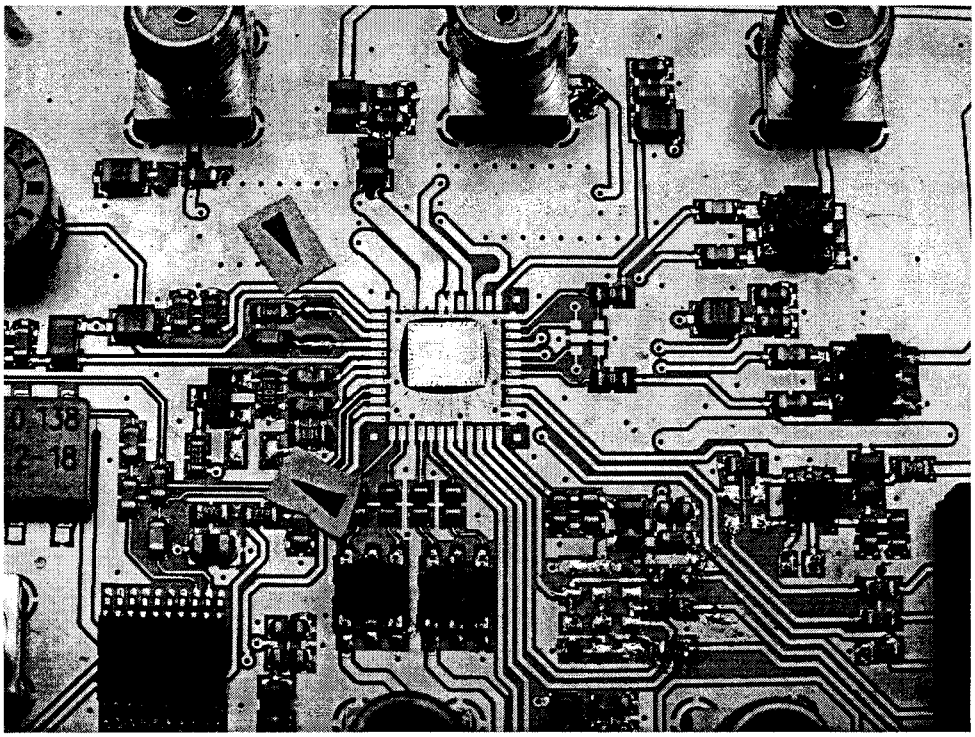
Fast Digital AGC Concept



AGC Related Antenna Signal Level



Silicon Evaluation

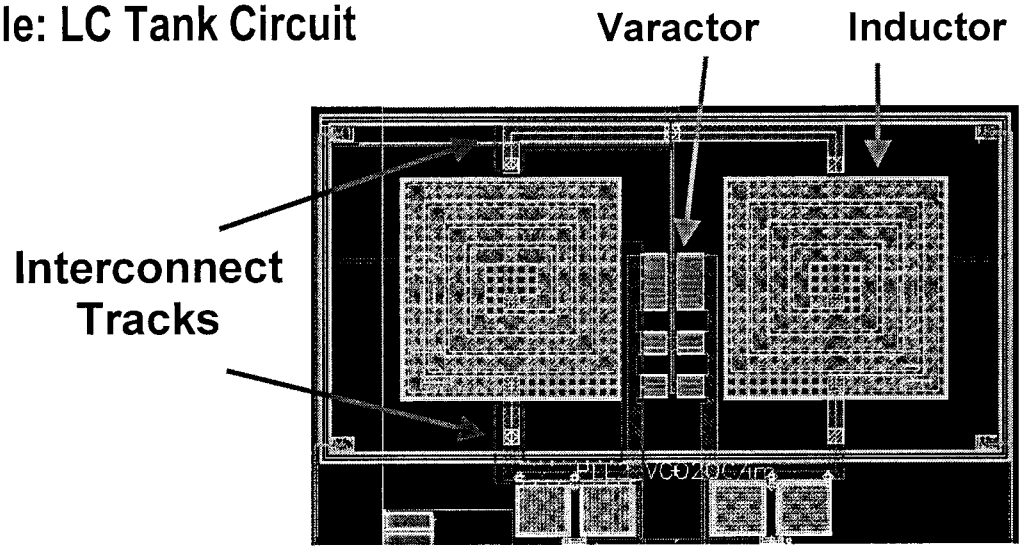


RF-IC Design Challenges – 1(2)

- RF Parasitics – Example of Impact:
  - 1 mm bond wire: 1 nH ~ 38 Ohm @ 6 GHz
  - Short lead capacitance: 0.5 pF ~53 Ohm @ 6 GHz
- Origins
  - Discrete external components, PCB
    - Component models, RF µ-strip design (s-Parameters)
  - Package related (Leadframe, Bond Wire, etc.)
    - Models generated by EM Simulation & Measurement
  - On Chip Parasitics
    - Standard RC extraction (mainly concentrated elements)
    - RF Extraction Tools OR EM Field Solver

RF-IC Design Challenges – RF Extraction

Example: LC Tank Circuit

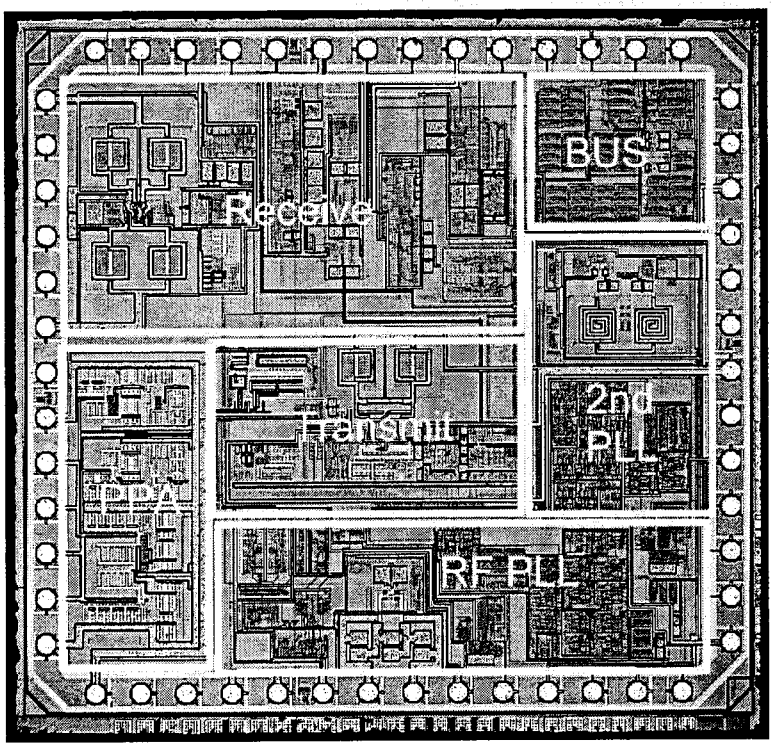


- *Inductor Element: 2.1nH (model from foundry, Si characterized)*
- *Tank Tracks: Ls = 0.52nH; Rs = 9 Ohm; Cpar=300fF (simplified)*

RF-IC Design Challenges – 2(2)

- Modern RF-ICs ⇔ traditional RF technology
  - RF Tradition: Koaxial, 50 Ohm
    measurement equipment, components, discrete PA, antenna
- RF-ICs: Differential Circuits & Signals, 25…200 Ohm
  - Isolation: key for integration of system on Chip (Direct Conversion !)
    - Issues (ex.): RX Input => LO ("Pulling"), PA=>Oscillator
  - With fully differential circuit topologies and signals:
    only non-linearities of order (2N+1) become relevant ;
    even order harmonics still exist on supply and ground rails.
  - Adaptation (balun): cost & performance penalty

Chip Micrograph



- Die area: 17mm^2
- 48pin
  leadless chip carrier
  or flip-chip application
  or COB
- Process: BICMOS
  SiGe 0.5µm
- Application:
  Video Streaming
  key is QoS
- Acknowledgements