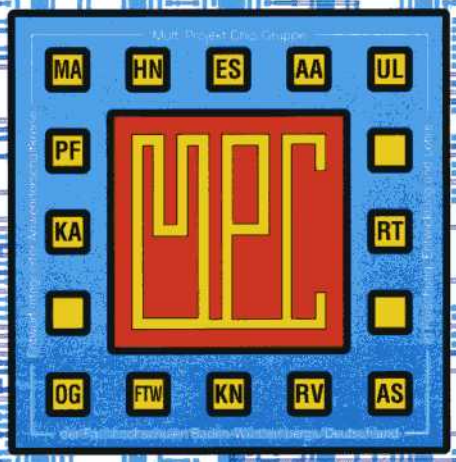
MULTIPROJEKTCHIP GRUPPE

BADEN-WÜRTTEMBERG

MPC-Workshop Juli 2005

Heilbronn



MULTIPROJEKTCHIP GRUPPE

BADEN-WÜRTTEMBERG

MPC-Workshop Juli 2005

Heilbronn

Cooperating Organization
Solid-State Circuits Society Chapter
IEEE Germany Section

Herausgeber: Fachhochschule 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

Entwicklung von Ansteuer-und Auswerteschaltungen für integrierte Hall-Sensoren

J. Thielmann, H. Richter
thielm@ims-chips.de, richter@ims-chips.de
Institut für Mikroelektronik Stuttgart
Allmandring 30a, 70569 Stuttgart
http://www.ims-chips.de

Auf Basis eines 0.5µm CMOS Mixed-Signal-Gate-Arrays (GATE FOREST®) wurde ein integriertes Hall-Sensorsystem entwickelt. Die von integrierten Hall-Platten erzeugte Hall-Spannung wird verstärkt, mittels der Spinning Current Methode offsetkorrigiert und gefiltert. Alle genannten Komponenten inklusive der Konstant-Stromquelle, Kontroll- und Ansteuerlogik sind mit den Hall-Platten gemeinsam auf einem Chip integriert. Das System ist für den Betrieb von 0 bis 100kHz konzipiert und liefert ein dem Magnetfeld proportionales Ausgangssignal von ca. 30 V/T. Das System kann einzeln oder als Sensor-Array und bei Bedarf zusätzlich mit einem A/D-Wandler sowie digitaler Signalverarbeitung kombiniert werden.

1. Einleitung

Die Entwicklung des Hall-Sensorsystems erfolgte im Rahmen einer Diplomarbeit am Institut für Mikroelektronik Stuttgart (IMS). Die Hall-Platten basieren auf dem GATE FOREST® Master des IMS und werden in einem Standard-CMOS-Prozess gefertigt. Die Hall-Sensorsysteme sollen später in kleinen bis mittleren Stückzahlen gefertigt werden und für eine Vielzahl unterschiedlicher Anwendungen verwendbar sein.

Die Ansteuer- und Auswerteschaltungen werden mit den Hall-Platten auf einem ASIC integriert, da die durch die Hall-Platten erzeugte Spannung sehr schwach und somit auch störanfällig ist. Daneben besteht die Forderung, die Ansteuer- und Auswerteschaltungen komplett auf einem Baustein zu integrieren.

In den folgenden Kapiteln wird dargestellt:

2. Hall-Effekt
3. Aufbau des Hall-Sensorsystems
4. Spinning Current Messmethode
5. Umschalter und Signalgenerator
6. Referenzstromquelle
7. Verstärker und Pegelwandler der Hall-Spannung
8. Offsetunterdrückung
9. Filter und Ausgangsverstärker

Die Hall-Platten wurden in einem 0.5µm CMOS-Gate-Array integriert. In diesem Gate-Array können digitale und analoge Schaltungen realisiert werden [1], wie in Abbildung 1-1 dargestellt ist.

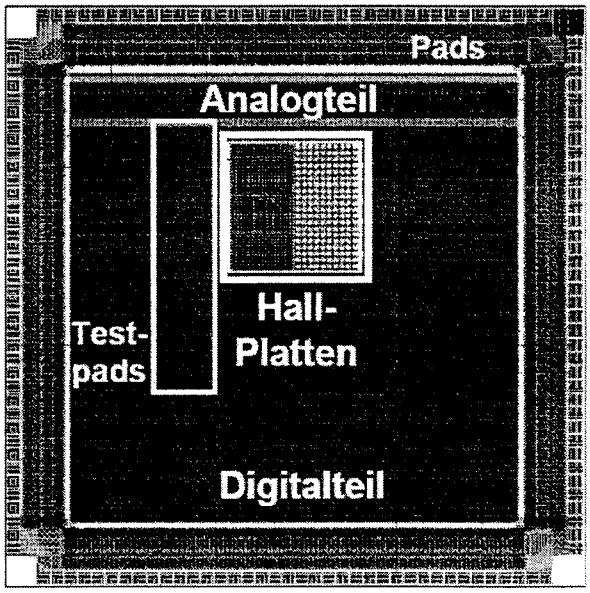


Abbildung 1-1: Aufbau des IMS-Master

Eine zentrale Anforderung der Arbeit lag in der Entwicklung von präzisen, schaltbaren Stromquellen, sowie einer Auswerteschaltung für die vorhandenen Hall-Platten. Die Auswerteschaltung soll die generierte Hall-Spannung verstärken und konditionieren. Schließlich sollen die erreichbaren Zielparameter des Hall-Sensorsystems definiert werden.

Die Anforderungen an das System sind:

-Magnetfeld	0 ... 50 mT
-Frequenz des Signals	0 ... 100 kHz
-Ausgangssignal	2,5 V ± 1 V

2. Hall-Effekt

Der Hall-Effekt, wurde im Jahre 1879 von Edward Hall entdeckt. Er beobachtete, dass wenn ein Strom durch einen Leiter geschickt wird und gleichzeitig dazu senkrecht auf den Leiter ein magnetischer Fluss wirkt, eine Ladungsverschiebung innerhalb des Leiters erfolgt, siehe Abbildung 2-1.

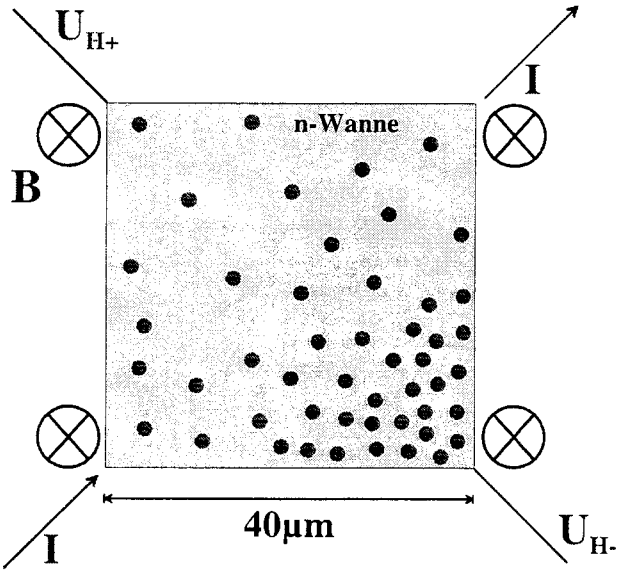


Abbildung 2-1: Hall-Effekt

Bei den hier verwendeten Hall-Platten handelt es sich um isolierte n-Wannen in p-Substrat. Die n-Wanne ist in Abbildung 2-1, grau, dargestellt. Die Hall-Platten haben eine quadratische Form mit der Kantenlänge von ca. 40µm.

Ein Strom wird diagonal durch die Silizium-Platte geschickt. Nur der senkrecht auf die Platte wirkende magnetische Fluss wirkt sich auf die Ladungsverschiebung aus und kann somit ausgewertet werden, siehe [2] und [3]. Durch die Ladungsverschiebung in der Platte wird eine Spannung an den beiden anderen Ecken der Platte erzeugt. Diese Spannung wird Hall-Spannung genannt und kann nach folgender Formel berechnet werden:

$$U_{Hall} = R_{H+} \cdot I \cdot B + U_{Off} \quad (1)$$

Die benutzten Kürzel bedeuten:

R_{H+}	Hall-Konstante, enthält alle geometrischen und technologischen Einflussgrößen
I	Strom durch die Hall-Platte
B	magnetische Flussdichte
U_{Off}	Offset-Spannung

Der Hall-Spannung ist jedoch noch eine signifikante Offset-Spannung überlagert. Die temperaturabhängige Offset-Spannung wird im Wesentlichen durch mechanische Spannungen im Silizium und durch Unsymmetrien in der geometrischen Anordnung hervorgerufen. Somit hängt der Wert der Offset-Spannung insbesondere von Fertigungsschwankungen ab, die nicht vorhersagbar sind und muss somit während der Messung eliminiert werden.

Bei siliziumbasierten Hall-Platten ist R_{H+} und damit die Hall-Spannung sehr gering, ungefähr 50mV bei einem Strom von 1mA und einem R_{H+} von 1000m²/As. Aus diesem Grund ist eine direkte Verbindung mit den Verstärkerschaltungen angebracht und führt zu dem in Abbildung 3-1 dargestellten Hall-Sensorsystem.

Hall-Sensoren werden heutzutage schon vielseitig eingesetzt. Typische Anwendungen sind: Umdrehungsmessung, Geschwindigkeitsmesser, Positionsgeber, Winkelgeber, Phasenmesser, Schalter, usw., siehe [4] und [5].

3. Aufbau des Hall-Sensorsystems

In Kapitel 2 wurde beschrieben, dass die Hall-Spannung von einem variablen Offset überlagert wird. Der Betrag der Offset-Spannung kann teilweise sogar größer sein als der Betrag der Hall-Spannung.

Aus diesem Grunde darf nicht nur die Hall-Spannung an sich aufbereitet werden, sondern es muss eine Schaltung entwickelt werden, bei der die Offset-Spannung eliminiert und nur noch ein Vielfaches der Hall-Spannung weiterverarbeitet wird. Daher wurde die in Abbildung 3-1 gezeigte Anordnung in mehreren Stufen entwickelt. Die Komponenten werden in den folgenden Kapiteln beschrieben.

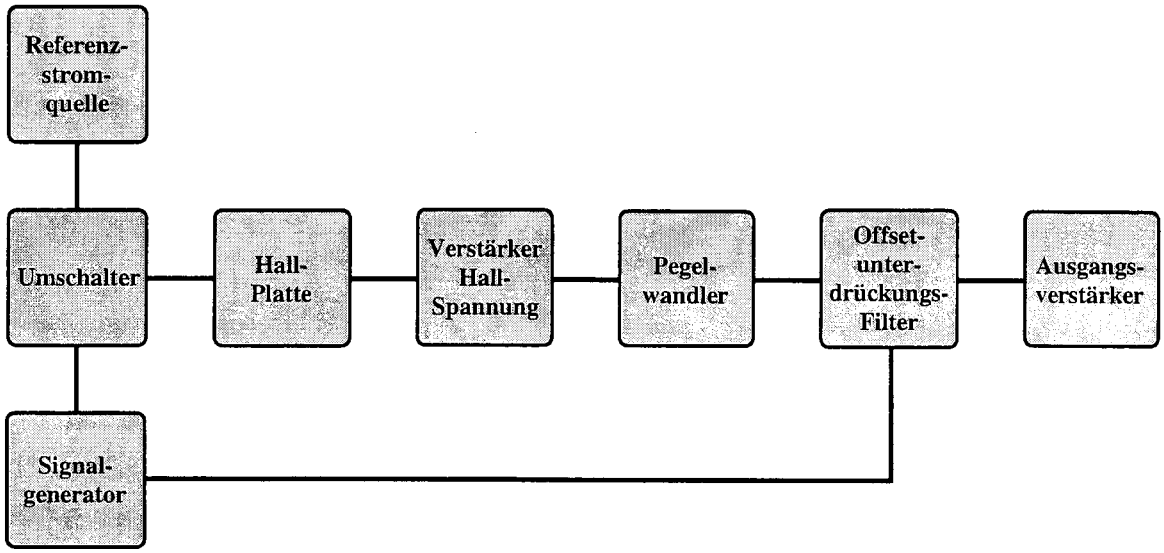


Abbildung 3-1: Aufbau des Hall-Sensorsystems

4. Spinning Current Messmethode

Um die Offset-Spannung aus der Hall-Spannung herausrechnen zu können, wird der Effekt ausgenutzt, dass die Offset-Spannung näherungsweise unabhängig vom Magnetfeld ist. Gleichzeitig dreht sich bei Umkehr der Stromrichtung und Vertauschung des Spannungsabgriffs das Vorzeichen um, siehe [2] und [6].

Die Richtungsabhängigkeit der Offset-Spannung kann dadurch erfasst werden, dass nicht nur die Stromrichtung umgekehrt wird, sondern der Strom entlang der Symmetrie-Achse der Hall-Platte jeweils in beide Richtungen geschickt wird.

Im vorliegenden Master ist eine quadratische Platte implementiert, so dass die in Abbildung 4-1 dargestellten Stromrichtungen möglich sind, die dann in zyklischer Reihenfolge ausgewählt werden. Andere Möglichkeiten basieren auf einer 8-fachen Symmetrie, die eine noch bessere Offset- Reduktion aufweist, [2] und [6].

Bei einer 4-fachen Anordnung steht nach einem vollständigen Durchlauf, entsprechend Formel 2, folgende Spannung zur Verfügung.

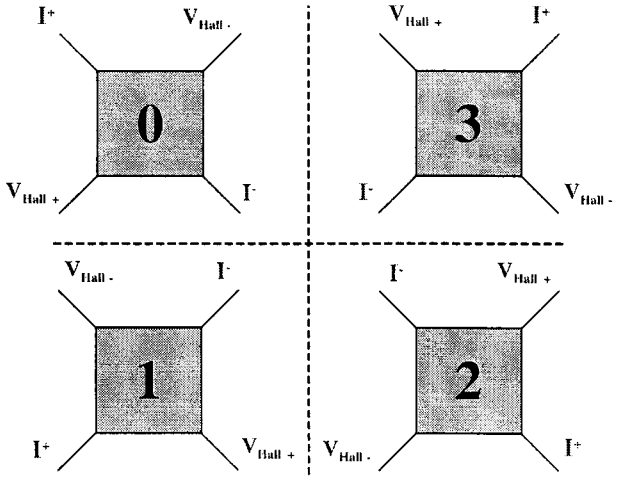
$$\sum_i U^i_{Platte} \approx 4 \cdot U_{Hall} + \Delta U_{off\_rest} \qquad (2)$$

Leider verbleibt auch bei der Spinning Current Messmethode eine Restoffset-Spannung ΔU_{off_rest}, diese kann mit Hilfe der genannten Methode nicht erfasst werden. Sie ist bei entsprechend beherrschtem Herstellungsprozess jedoch so klein, dass sie die Messung nicht stört.

Da ΔU_{off_rest} nicht quantifizierbar ist, wird sie nicht weiter betrachtet. In den folgenden Simulationen wird angenommen, dass $U_{off\_A} = U_{off\_B}$ ist, was aber keine Einschränkung der Allgemeingültigkeit der Ergebnisse bedeutet.

Da eine feste Verdrahtung der Stromanschlüsse nicht möglich ist, wird ein weiteres Element zwischen Stromquelle und Hall-Platte gesetzt, das eine Umschaltung der Stromeinspeise- und Hall-Spannungsmesspunkte erlaubt.

$$U^0_{Platte} = U_{Hall} + U_{off\_A} \qquad U^3_{Platte} = U_{Hall} - U_{off\_B}$$



$$U^1_{Platte} = U_{Hall} + U_{off\_B} \qquad U^2_{Platte} = U_{Hall} - U_{off\_A}$$

Abbildung 4-1: Messprinzip Spinning Current

5. Umschalter und Signalgenerator

Die Umschalteinheit hat die Aufgabe, die verschiedenen Punkte der Hall-Platte an den jeweilig richtigen Anschluss für die Stromeinspeisung oder die Spannungsmessung zu schalten. Da bei der Messung jeder Punkt der Hall-Platte an jeden Punkt der Spannungsmessung sowie an die Referenzstromquelle gelegt werden muss, ist die aufwendige Beschaltung entsprechend der Abbildung 5-1 notwendig.

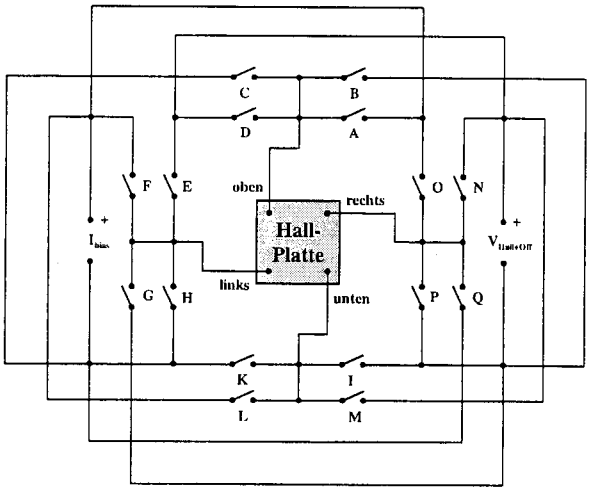


Abbildung 5-1: Aufbau des Umschalters

Als Schalter kommen einzelne, geeignet dimensionierte NMOS- und PMOS-Transistoren zum Einsatz.

Der Signalgenerator hat verschiedene Aufgaben zu erfüllen. Zum einen muss er für den Umschalter die Signale für die Umschaltung der Spannungs- und Stromanschlüsse liefern. Zum anderen werden auch Ansteuersignale für die Offsetunterdrückung benötigt, dort wird mit einem Chopper-Verfahren gearbeitet.

Die Signale für die Umschaltung der Spannungs- und Stromanschlüsse sind so konzipiert, dass es keine Überlappung der Schaltsignale bei der Umschaltung gibt, Abbildung 5-2 (Einrahmung). Hierbei wird deutlich, dass die Spannungsanschlüsse immer genau einen Takt versetzt geschaltet werden. Es wird immer zuerst abgeschaltet, bevor der Strom wieder an einen neuen Punkt angelegt wird. Zwischen jeder Umschaltung des Stromes liegt ein zusätzlicher Takt, damit wird ein mögliches Kurzschließen der Referenzstromquelle verhindert.

Alle Signale werden auch in negierter Form bereitgestellt, da sowohl NMOS- als auch PMOS-Transistoren zum Einsatz kommen. Die Signale "Stufe 1" und "Stufe 2" werden für die Schaltungen im Bereich der Offsetunterdrückung benötigt.

Alle benötigten Schaltsignale werden in einer Einheit generiert, da nur ein Clock-Signal zur Verfügung steht und alle Signale taktsynchron sein müssen.

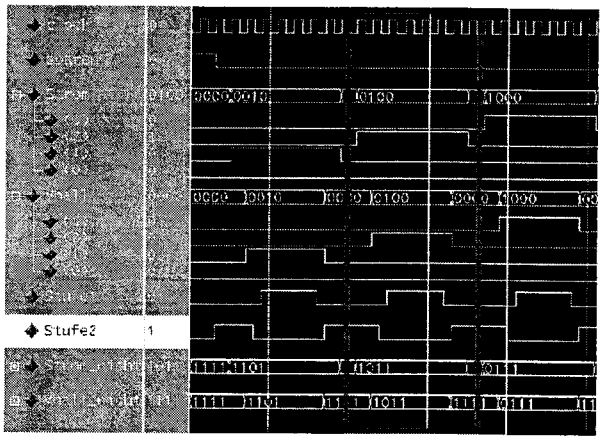


Abbildung 5-2: Taktbild des Signalgenerators

Der Signalgenerator wurde in VHDL codiert und simuliert, [7] und [8]. Danach wurde der VHDL-Code synthetisiert und zusammen mit der Hall-Platte auf Transistorebene simuliert, um die richtige Funktion der Schaltung zu überprüfen.

6. Referenzstromquelle

Der Strom durch die Hall-Platte soll konstant auf ca. 1mA gehalten werden. Dafür wurden verschiedene Referenzstromquellen untersucht, hier wird nur die optimierte Quelle beschrieben.

Die Schaltung mit Vittoz-Referenzquelle wurde in Anlehnung an [9] implementiert. Es handelt sich im Prinzip um eine erweiterte Vittoz-Stromquelle und drei nachgeschaltete Stromspiegel, [10] und [11]. Das Schaltbild der verwendeten Referenzstromquelle ist in Abbildung 6-1 zu sehen.

Die Vittoz-Quelle an sich liefert nur sehr kleine Ströme. Aus diesem Grund wird der Strom mehrfach gespiegelt. Der Stromspiegel 3 muss eine Stromsenke bilden, damit die Hall-Platte an VDD betrieben werden kann.

Bei einer Vittoz-Quelle werden prinzipiell zwei Widerstände benötigt, die als Referenzspannungsquelle dienen. Da jedoch keine Widerstände in der notwendigen Größe zur Verfügung stehen, werden die Widerstände durch Spannungsteilerschaltungen mit Transistoren ersetzt. Diese haben des Weiteren die Aufgabe, die Spannungsvariation zu minimieren und den Einfluss der Temperatur auf die Stromgenauigkeit zu reduzieren.

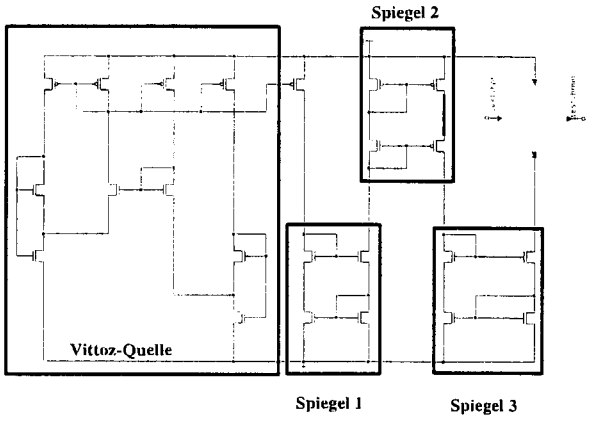


Abbildung 6-1: Schaltung mit Vittoz-Quelle

In der Abbildung 6-2 ist der Verlauf des Stromes durch die Hall-Platte als Funktion der Versorgungsspannung dargestellt.

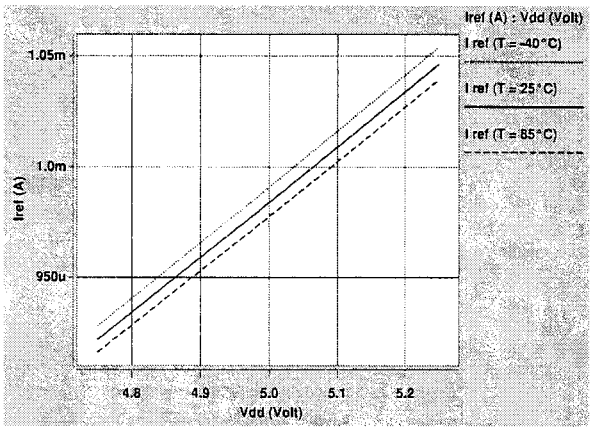


Abbildung 6-2: Simulation der Schaltung mit Vittoz-Quelle als Referenzquelle

Die Variation des Stromes durch Temperaturschwankungen sind wesentlich geringer als bei den anderen Schaltungen.

Ein Vergleich der Referenzstromquellen, siehe Tabelle 6-1, zeigt, dass die Quelle mit Diode im Referenzzweig immer schlechtere Ergebnisse liefert als die Schaltung mit Vittoz-Quelle im Referenzzweig. Bei beiden Stromquellen ist leider eine recht hohe Abhängigkeit vom Verlauf der Versorgungsspannung zu erkennen.

Diese Abhängigkeit der Referenzstromquelle mit Vittoz-Quelle kommt von der erweiterten Vittoz-Stromquelle und wird durch die Stromspiegel nur noch weiter verstärkt. Eine Verbesserung wäre, die Versorgungsspannung für die Vittoz-Quelle durch eine Bandgap-Referenzspannungsquelle wesentlich stabiler zu halten.

Parameter	Referenz-strom-quelle mit Diode	Referenz-strom-quelle nach Vittoz	Einheit
Empfindlichkeit $I_{Hall}\|_{\Delta VDD}$	25	25	$\frac{\%}{V}$
Empfindlichkeit $I_{Hall}\|_{\Delta T}$	0.052	0.012	$\frac{\%}{K}$
Gesamtfehler I_{Hall} über Betriebsbereich	± 9,5	± 7	%

Tabelle 6-1: Vergleich der Referenzstromquellen

Der Gesamtfehler über den Betriebsbereich umfasst die Variation der Versorgungsspannung um ± 5% sowie den Einfluss der Temperatur über einen Bereich von -40 °C bis + 125 °C.

7. Verstärker und Pegelwandler der Hall-Spannung

Die Spannung aus der Hall-Platte soll wenig belastet werden. Deshalb muss die Hall-Spannung unmittelbar verstärkt werden und dafür kann nur ein Verstärker mit einer hohen Eingangsimpedanz eingesetzt werden. Die Verstärkung wurde zu Eins gewählt, um eine Impedanzwandlung durchzuführen und einen höheren Stromfluss zu zulassen. Der Pegelwandler hat an seinen Eingängen Widerstände geschaltet, die sonst die Hall-Platte belasten und den Signalhub verändern würden. Um zu verhindern, dass der Impedanzwandler während des Umschaltens in seinen Grenzbereich geht, wurde eine Abkopplung durch ein Transmissions-Gatter [11] realisiert.

Der Pegelwandler wandelt das differenzielle Signal der beiden Punkte der Hall-Platte in ein Signal um, bei dem sich die Differenz der beiden Spannungen der Hall-Platte um die halbe Versorgungsspannung herum bewegt, auch single-ended genannt. Gleichzeitig wird diese Spannung um einen Faktor von ca. 17 verstärkt.

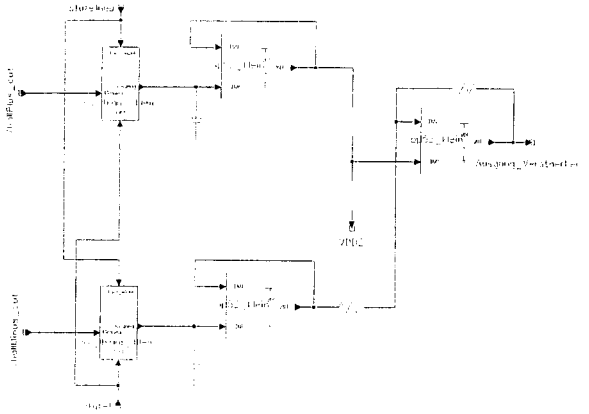


Abbildung 7-1: Schaltplan Verstärker Hall-Spannung und Pegelwandlung

Für alle Simulationen wurde das Magnetfeld durch eine Sinus-Quelle dargestellt. Das Magnetfeld hat eine Amplitude (Spitze-Spitze) von 100mT und die Hall-Spannung beträgt mit den gegebenen Werten etwa 100mV.

In der Abbildung 8-2 ist im Graphen a) das Magnetfeld dargestellt. Graph b) ist die Hall-Spannung. Diese Hall-Spannung, Vhall, ist die Differenz der Spannungen an der Hall-Platte, in Abbildung 7-1 VhallMinus_out und VhallPlus_out genannt. Bei Graph b) ist zu erkennen, dass der Verlauf durch zwei Einhüllende begrenzt wird. Die obere Einhüllende entspricht dem Offsetanteil der Hall-Spannung und die untere Einhüllende dem negierten Offsetanteil. Des Weiteren ist zu erkennen, dass sich der Offset mit doppelter Frequenz zum Abtastzyklus ändert. Dies hängt mit der Form der Modellierung der Hall-Platte für die elektrische Simulation unter HSPICE zusammen. In dem verwendeten Modell wurde nur in einem einzelnen Zweig ein Offsetwiderstand integriert.

Graph c) in Abbildung 8-2 zeigt den Verlauf des Signals nach Verstärkung und Pegelwandlung. Hier ist zu erkennen, dass nur die Spannungspegel verändert wurden, aber die Offset-Spannung weiterhin proportional vorhanden ist.

8. Offsetunterdrückung

Das zum Magnetfeld proportionale Ausgangssignal entspricht der Mittellinie in Abbildung 8-2c. Um dieses aus dem Vorhandenen herauszufiltern, wurden zwei Möglichkeiten untersucht.

Die erste Möglichkeit ist, dieses Signal durch einen Tiefpass zu senden, um damit die hochfrequente Offset-Spannungsvariation herauszufiltern. Da die Frequenz der Offset-Spannungs-Variation von 800kHz sich nicht sehr von der angestrebten maximalen Signalfrequenz unterscheidet, ist es praktisch nicht möglich, die Änderung der Offsetspannung zu unterdrücken, ohne gleichzeitig das Nutzsignal zu beeinflussen. Deshalb wird eine direkte Auswertung nach Formel (2) implementiert. Diese Version benötigt wenige Schalter und Kapazitäten.

Der Aufbau der Offsetunterdrückung ist in Abbildung 8-1 zu sehen. Für jede Phase der Abtastung während eines Umlaufs steht ein Zweig zur Verfügung, der die jeweilige Spannung speichert. Die rechten Schalter werden alle gleichzeitig geöffnet und es findet ein Spannungsausgleich statt, bei dem die jeweiligen Hall-Spannungen und ihre Offsets sich ausgleichen. Nach erfolgtem Ausgleich liegt an der folgenden Schaltung die mittlere Hall-Spannung an.

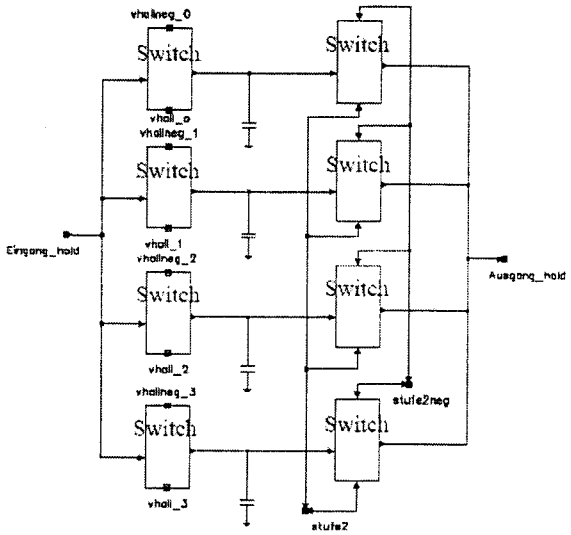


Abbildung 8-1: Schaltplan Offsetunterdrückung

In Abbildung 8-2, Kurve d), ist das Simulationsergebnis nach der Offsetunterdrückung zu sehen. Es wird deutlich, dass der Verlauf im Wesentlichen der Mittellinie von c) entspricht und keine positiven oder negativen Offsets mehr sichtbar sind. Durch die Mittelungsmethode wurde jedoch das Signal stufenförmig. Bei einer anschließenden Analog-Digital-Wandlung würde das Signal an dieser Stelle direkt weiterverarbeitet, indem die Wandlung des Signals genau auf die Zeitspanne gelegt würde, in dem das Signal einen konstanten Zustand eingenommen hat. Somit würde eine Speicherung des Signals für die AD-Wandlung wegfallen.

Soll das Signal jedoch analog weiterverarbeitet werden, muss das Signal noch einmal gefiltert werden, was im folgenden Kapitel 10 beispielhaft durchgeführt wird.

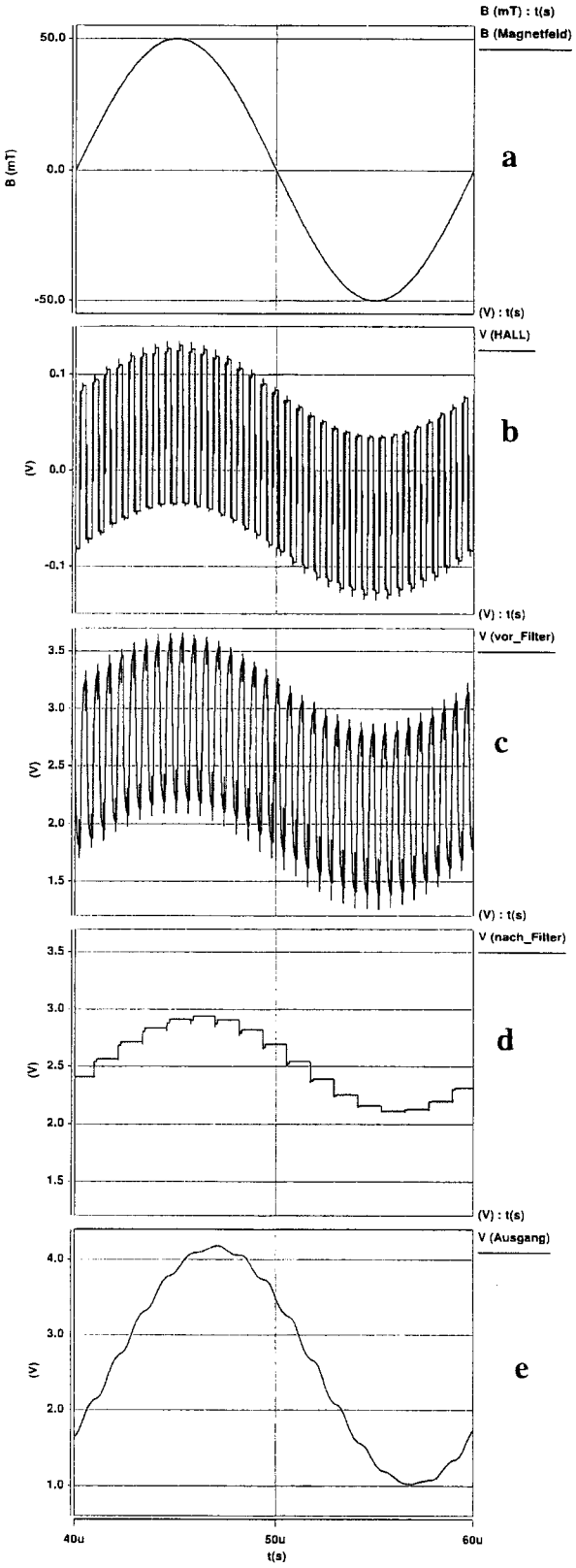


Abbildung 8-2: Simulationsergebnisse

9. Filter und Ausgangsverstärker

Um das Signal dem Verlauf des Magnetfeldes ähnlicher werden zu lassen, wird am Ende der Aufbereitung der Hall-Spannung ein aktives Filter eingesetzt, das die Stufen glättet.

Es wurden verschiedene aktive und passive Filter getestet. Mit dem in Abbildung 9-1 dargestellten aktiven Filter 4. Ordnung, siehe [12] und [13], wurden die besten Ergebnisse erreicht. Die Berechnung der Widerstands- und Kapazitätswerte erfolgte anhand der Werte von Bessel.

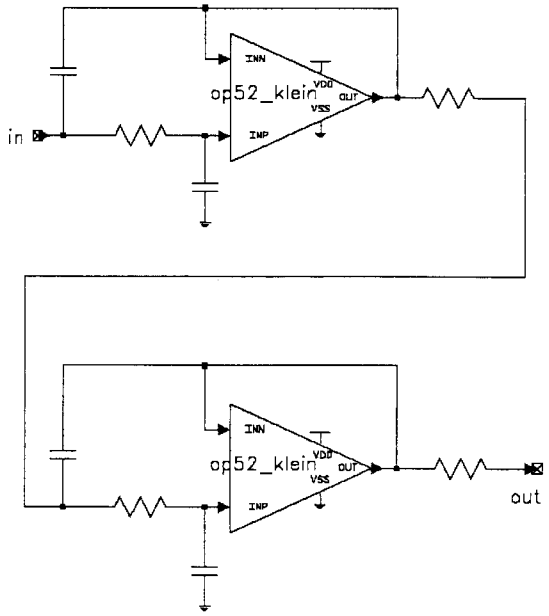


Abbildung 9-1: Schaltplan aktives Filter

Es musste ein Kompromiss zwischen Filterung und Übertragungsfrequenz eingegangen werden. Die Simulationsergebnisse nach der Filterung sind in der Abbildung 8-2e zu sehen. Eine leichte Abweichung ist weiterhin im Signal zu erkennen, aber eine stärkere Filterung dämpft die oberen Signalfrequenzanteile zu stark.

Die Grenzfrequenz des Hall-Sensorsystem wird durch die Messmethode / Messfrequenz und die Filterung auf eine Frequenz von 100kHz beschränkt. Bei höheren Frequenzen wird das Signal durch die aktive Filterung stark gedämpft.

10. Zusammenfassung

Das in diesem Bericht vorgestellte Hall-Sensor-System ist in der Lage, sowohl statische als auch wechselnde Magnetfelder bis zu einer Frequenz von 100kHz zu verarbeiten. Das Mixed-Signal-Gate-Array hat sich für diese Aufgabe als sehr brauchbar gezeigt, da alle benötigten Komponenten, digital wie analog, zusammen auf einem Chip integriert werden konnten.

Die Randbedingungen konnten zu wesentlichen Teilen erfüllt werden. Die zu messenden Magnetfelder können in dem angegebenen Rahmen von 0 ... 100kHz gemessen werden. Ein Signalhub von 2,5V ±1V konnte sogar ein wenig erweitert werden. Der Strom durch die Hall-Platte wurde mit 1mA angesetzt, ein Strom von 975µA wurde schließlich simuliert. Der System-Takt, der an den Chip angelegt wird, kann zwischen 10MHz und 40MHz variieren, er hat keinen Einfluss auf die Signalform, sondern nur auf die Messgeschwindigkeit und somit die Messfrequenz für das Magnetfeld.

Im weiteren Verlauf der Diplomarbeit wird an einer Optimierung der Messmethodik gearbeitet, um auch Magnetfelder mit einer Frequenz oberhalb von 100kHz sicher zu messen.

11. Danksagungen

Danken möchte ich, J.Thielmann, Herrn Prof. Dr. Bonath der FH Gießen, dass er mir die Diplomarbeit ermöglichte. Den IMS Mitarbeitern möchte ich danken, dass sie bei den vielen Fragen, die ich während der Diplomarbeit hatte, immer Zeit für mich hatten und sie mir diese beantworteten. Besonders möchte ich mich bei Herrn C. Burwick und Herr K. Warkentin bedanken, dass sie mir bei Fragen immer weitergeholfen und zur Entstehung der Diplomarbeit wesentlich mit beigetragen haben.

12. Literaturangaben

[1] Karsten Warkentin, "Entwicklung der analog Masterzelle für die Gate Forrest Familie GFQ", Entwicklungsbericht, 2002, IMS interne Literatur

[2] Ralf Wunderlich, "Entwurf eines integrierten, analogen Hallsensorsystems in CMOS-Technik", Dissertation in der Fakultät Elektrotechnik und Informationstechnik an der Universität Dortmund, Juli 2002

[3] Ed Ramsden, "Hall Effect Sensors, Theory and Application", aus dem Advanstar Communications Inc. Verlag, 2001, ISBN 0-9298870-1

[4] http://rb-k.bosch.de/de/start/

[5] http://www.micronas.com/products/overview/sensors/index.php

[6] Carsten Müller-Schwanneke, "Offsetreduktion und Sensitivität bei Silizium-Hall-Sensoren", Dissertation im Institut für Halbleitertechnik der Universität Stuttgart und Max-Planck-Institut für Festkörperforschung Stuttgart, Juni 2000

[7] J. Reichardt und B. Schwarz, "VHDL-Synthese, Entwurf digitaler Schaltungen und Systeme", 2. überarbeitete Auflage von 2001 vom Oldenbourg Verlag

[8] C. W. Scherjon, "VHDL Style Guide", Version 1.2, 2003, IMS interne Literatur

[9] Edgar Mauricio Camacho-Galeano und Carlos Galup-Montoro und Márcio Cherem Schneider, "Design of an ultra-low-power current source" oder "An ultra-low-power self-biased current reference", IEEE-Verlag, 2004, ISCAS 2004, Seite I 333

[10] R. Jacob Baker, Harry W. Li, David E. Boyce, "CMOS-Circuit Design, Layout and Simulation", IEEE-Verlag, 1998, ISBN 0-7803-3416-7

[11] Behzad Razav, "Design of analog CMOS integrated Circuits", McGraw-Hill Verlag 2001, ISBN 0-07-238032-2

[12] Seifart, Manfred, "Analoge Schaltungen", Auflage 1, 1987, aus dem Hüthig Verlag Heidelberg, ISBN 3-7785-1456-3

[13] Tietze, Schenk, "Halbleiter – Schaltungstechnik", 5. Springer Verlag Berlin, 5.Auflage, 1980, ISBN 3-540-09848-8

Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

FPGA basierte Gaborfilterung zur Beschleunigung eines Objekterkennungssystems

Eduardo Monari, Rüdiger Heintz, Prof. Gerhard Schäfer

Hochschule Karlsruhe, Moltkestr. 30, 76133 Karlsruhe

Tel.: 0721/925-2185, Fax.: 0721/925-1513, Ruediger.Heintz@hs-karlsruhe.de

ABSTRACT

In vorangegangenen Arbeiten wurde ein Bildverarbeitungssystem zur rotations- und skalierungsinvarianten Objekterkennung entwickelt. Die Objekterkennung basiert auf einer Merkmalsextraktion mittels Gaborfilterungen. Das Ziel dieser Arbeit war die Beschleunigung der Vorfilterung durch den Einsatz eines FPGA's. Dazu wurde eine IIR Gaborfilterung auf der Grundlage einer Floating Point Arithmetik entwickelt. Zum Vergleich wurde eine rein PC basierte Auswertung herangezogen.

1. Einleitung

An der FH Karlsruhe wurde ein rotations- und skalierungsinvariantes Objekterkennungssystem entwickelt [1]. Dieses Objekterkennungssystem basiert auf der Extraktion von lokalen Merkmalen mittels Gaborfiltern. Zur robusten Objekterkennung sind Merkmale aus ca. 36 Gaborfilterungen nötig. Die Berechnung dieser Merkmale benötigt trotz Verwendung eines geschwindigkeitsoptimierten Filteralgorithmus ca. 80% der gesamten Rechenzeit. Eine Beschleunigung der Filterung und damit der Merkmalsextraktion mit einem FPGA Baustein wurde als sinnvoll angesehen. Es wurden verschiedene Algorithmen zur Gaborfilterung verglichen. Als Filteralgorithmus kam ein speziell separierbares IIR Filter mit einer Floating Point Arithmetik zum Einsatz.

2. Die Merkmalsextraktion

Ein Gaborfilter entsteht durch Multiplikation einer komplexen Schwingung mit einer Gausglocke. Die Gaußglocke kann als Tiefpass interpretiert werden. Durch die Multiplikation mit einer komplexen Schwingung wird die Gaußglocke moduliert und es entsteht ein Bandpassfilter. Wie in [2] gezeigt, wird für die rotations- und skalierungsinvariante Objekterkennung ein spezielles Gaborfilter durch nachfolgende Beziehungen beschrieben

$$g_W(x,y,\omega_k,\theta_n) = \frac{\omega_k^2}{\pi^2}\cdot e^{\left(-\frac{(x^2+y^2)\cdot\omega_k^2}{2\cdot\pi^2}\right)}\cdot e^{(j\cdot\omega_k\cdot(\tilde{x}+\tilde{y}))}$$

$$\text{mit}\quad \begin{bmatrix}\tilde{x}\\ \tilde{y}\end{bmatrix} = \begin{bmatrix}\cos(\theta_n) & \sin(\theta_n)\\ -\sin(\theta_n) & \cos(\theta_n)\end{bmatrix}\cdot\begin{bmatrix}x\\ y\end{bmatrix}$$

Das Gaborfilter kann mittels zweier Parameter konfiguriert werden. Über den Parameter θh wird die Orientierung und über die Parameter ωk die Mittelfrequenz des Filterkernes festgelegt. Abbildung 1 zeigt den Realteil und Imaginärteil eines Gaborfilters.

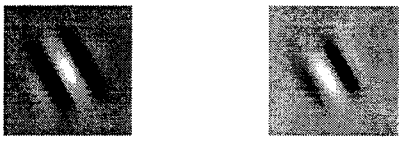


Abbildung 1: Realteil (links) und Imaginärteil (rechts) eines Gaborfilters.

Die Merkmalsextraktion ist in Abbildung 2 dargestellt. Das Eingangsbild wird mit den Gaborfiltern gefaltet. Für jede Filterung erhält man ein Ergebnisbild. Zur Vereinfachung der Darstellung, sind nur die Realteile der Gaborfilter und der Betrag der Filterantworten dargestellt.

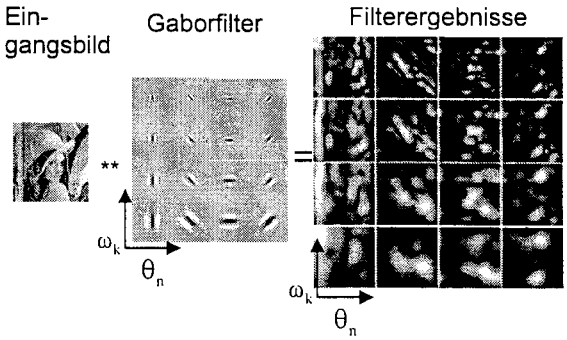


Abbildung 2: Merkmalsextraktion mittels einer Gaborfilterbank

Betrachtet man nun die Filterergebnisse eines Bildpunktes, lassen sich die Filterantworten an dieser Stelle zu einer Matrize zusammenfassen, welche als Jetmatrize bezeichnet wird.

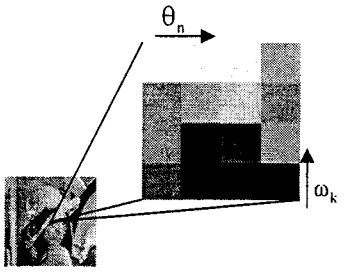


Abbildung 3: Die Jetmatrize

Die Parameter der Gaborfilter werden mittels der Formel (4) und (5) festgelegt.

$$\omega_k = 2^{\frac{m}{2}} \quad \text{mit } m = \{1,\ldots,M\} \; M>1 \qquad (4)$$

$$\theta_n = n \cdot \frac{\pi}{N-1} \quad \text{mit } n = \{1\ldots N\} \; N>1 \qquad (5)$$

Durch die entsprechende Wahl der Parameter [1] kann eine Rotation des Eingangsbildes in eine zirkuläre Verschiebung an der X-Achse der Jetmatrize überführt werden. Eine Bildskalierung führt zu einer Verschiebung der Jetmatrize entlang der Y-Achse.

3. Die Objekterkennung

Der in Kapitel 2 beschriebene Effekt, dass Skalierung und Rotation des Eingangsbildes zu Verschiebungen in der Jetmatrize führen, wurde verwendet um einen rotations- und skalierungsinvarianten Umgebungssucher zu entwickeln. Dieser Umgebungssucher basiert auf der Bestimmung der Korrelation zwischen Referenz - Jetmatrize und allen Jetmatrizen im Suchbild. Die Korrelation liefert einen Korrelationswert als Maß für die Ähnlichkeit und einen Verschiebungswert, mittels welchem die Rotation und Skalierung bestimmbar ist. Diese Funktionen werden in einen übergeordneten Objektsucher verwendet, um ein Objekt zu lokalisieren. Dazu werden markante Umgebungen im Referenzobjekt markiert und diese im Suchbild zugeordnet. Das Objekterkennungssystem besitzt eine sehr gute Erkennungsgüte, wie in Abbildung 4 erkennbar ist.

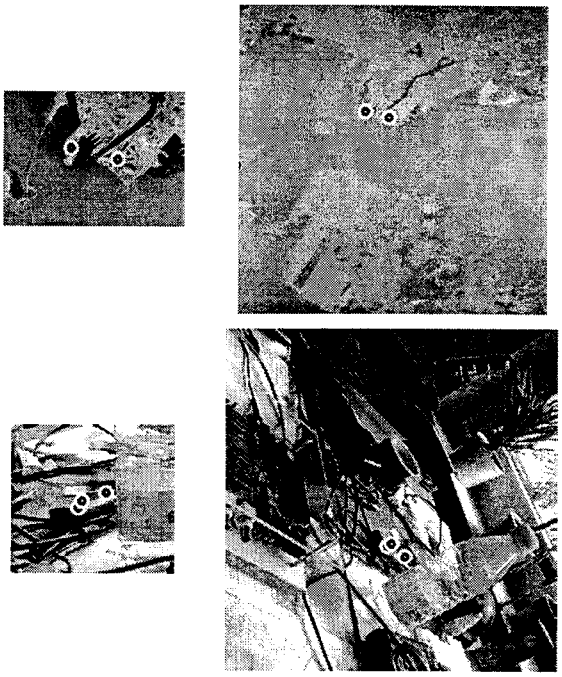


Abbildung 4: Beispiele für die Objektlokalisierung

4. Der Filteralgorithmus

Zur Beschleunigung der Merkmalsextraktion für PC basierte System wurden verschiedene Ansätze verfolgt. Zunächst wurde die Separierbarkeit des Gaborfilters verwendet, um die zweidimensionale Filterung in zwei eindimensionale Filterungen zu zerlegen und dadurch zu beschleunigen. Eine weitere Verbesserung der Performance wurde durch die Realisierung der Faltungen mittels Fouriertransformation erreicht. Die Fouriertransformation wurde dabei mittels der FFTW[3] durchgeführt. Zuletzt wurde eine Realisierung der Gaborfilterung als IIR Filter gefunden[4]. Es stellte sich heraus, dass dieser Algorithmus nochmals ungefähr um den Faktor 2 schneller war, als die FFTW basierte Realisierung.
Dieser IIR basierte Filteralgorithmus teilt die zweidimensionale Faltungen in zwei eindimensionale Faltungen auf. Die eindimensionale Faltung teilt sich wiederum in einen Vorwärtslauf und einen Rückwärtslauf auf. Die Filterstruktur ist für Vorwärtslauf und Rückwärtslauf nahezu gleich, besteht nur aus 12 Multiplikationen und 12 Additionen und ist unabhängig von den Filterparametern. Die Filterstruktur besitzt auch einen Nachteil: die einzelnen Durchläufe folgen sequentiell aufeinander, somit ist keine Parallelisierung der Durchläufe möglich und die Auswertung muss mit Floating Point Operationen erfolgen, um den Fehler gering zu halten.

5. Die Entwicklungsplattform

Ziel dieser Untersuchung war es, zu bestimmen welcher Geschwindigkeitsgewinn mittels eines FPGA basierten Systems erreichbar ist. Eine aufwendige direkte Ankopplung an eine Kamera war daher nicht nötig. Es genügte ein System, welche die Bilddaten vom PC empfangen und zu diesem zurücksenden kann. Dazu bot sich eine Entwicklungsplattform mit PCI Schnittstelle an. Um die volle Flexibilität bei der Erzeugung der Filterbank zu erhalten, wurde entschieden, dass die Erzeugung der Filterbank mittels FPGA von der PC Seite gesteuert werden kann.

5.1. Die Hardware

Die Implementierung erfolgte auf einem "ALTERA PCI High-Speed Development Kit, Professional Edition" bestehend aus

- Stratix EP1S60F1020C6 device
- 32-bit or 64-bit PCI/PCIx Local Bus
- 33/66-MHz PCI/PCIx-Interface
- Memory 256-MByte PC133 DDR SDRAM (SODIMM)

Das PCI-Development-Kit ermöglicht eine optimale Implementierung des FPGA-Filters in das bereits bestehende PC-Bildverarbeitungssystem. Die Kommunikation zwischen Windows-Applikation und FPGA kann somit komplett über PCI-Bus stattfinden.

Der EP1S60 bietet mit 57.000 Logic Elemente (LE) und 144 Multiplizierer in DSP-Blocks sowie ausreichend internen Speicher genug Platz und Möglichkeiten zur Bearbeitung dieser Untersuchung.

Das Design wurde in VHDL mit ALTERA Quartus 3.0 realisiert. Die Board-Level-Simulation wurde mit ModelSim 5.7c durchgeführt.

5.2. FrontEnd-Applikation ImageTool

Das Objekterkennungssystem ist als Multi Dokumenten Interface (MDI) Anwendung mittels C++ implementiert. Diese Bildverarbeitungsplattform im Weiteren als ImageTool bezeichnet, wurde um eine Schnittstelle zur FPGA Karte erweitert und ermöglichte den direkten Vergleich der Filterung mittels FPGA und der rein PC basierten Filterung.

6. FPGA-Design

In diesem Kapitel werden die durchgeführten Vorarbeiten beschrieben, die nötig waren, um ein Filteralgorithmus integrieren zu können.

6.1. Reference Design

Altera liefert in den jeweiligen PCI Development Kits u.a. einen „PCI-to-DDRRam Reference Design" mit [5]. Somit wird dem Entwickler eine beispielhafte Kommunikationsschnittstelle zwischen PCI-Bus und RAM angeboten, welche man für eigene Entwicklungen übernehmen kann.

Das Reference Design besitzt bereits Mechanismen für die Datenübertragungen im Master und im Target-Betrieb (Read/Write), wobei im Target-Betrieb das FPGA-Board als Slave arbeitet. Es werden außerdem Einzel- und Burstzugriffe unterstützt.

Die bereits implementierten PCI-Compiler und DDR-Controller ermöglichen im Zusammenhang mit einer Windows-Testapplikation eine sehr schnelle Inbetriebnahme des FPGA-Boards.

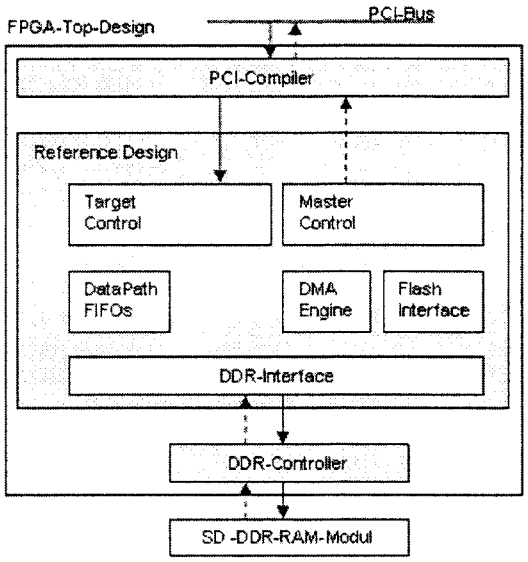


Abbildung 5: ALTERA PCI-2-DDR Reference Design

Das Reference-Design beinhaltet außerdem Beispielcode für Schreib- und Lesezugriffe von internen Registern (z.B. zur DMA-Konfiguration) und für eine Datenübertragung vom/zum SD DDR-RAM.

6.2. PCI-Interface / DDR-RAM-Interface

Leider hat sich gezeigt, dass die erwünschte Performance mit der Kommunikationsarchitektur des Reference-Designs nicht erreicht werden kann. Deshalb wurde für die Untersuchung der komplette Datenpfad, für Burstzugriffe auf dem DDR-RAM, umstrukturiert.

Das Altera-Design kann bis zu 128 Datenpakete im Burstmode übertragen. Danach muss ein neuer Zugriff gestartet werden. Ein PCI-Eingangsfifo nimmt zuerst alle übertragenen Daten auf, bevor das DDR-Interface diese Daten vom FIFO wieder abruft. Somit ergibt sich zum einen eine beschränkte Aufnahmekapazität durch die Größe des FIFOs und zum anderen eine unnötige Zeitverzögerung durch die Speicherung der Daten, da das Schreiben in den DDR-RAM erst nach der kompletten PCI-Übertragung erfolgt.

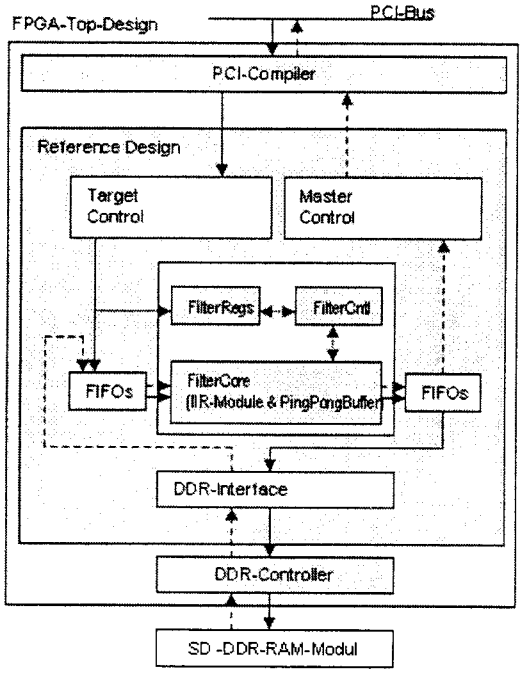


Abbildung 5: FPGA-Top-Design

Durch die Umstrukturierung des Design beginnt das Ablegen der Daten in den DDR-RAM sobald sich das FIFO auf über 8 Eintragungen gefüllt hat. Trotz weiterer Schreibzugriffe seitens des PCI-Busses wird der FIFO durch den höheren Takt des DDR-RAM-Moduls (166-200 MHz) schneller geleert. Sinkt nach einem Read-Zugriff der FIFO-Inhalt unter 8 Werte wartet das DDR-Interface bis sich dieses wieder gefüllt hat. Erst am Ende der Übertragung werden alle restlichen Werte abgerufen.

Somit finden zwischen Filterblock und DDR-RAM-Interface mehrere Burst-Zugriffe statt, die nur abhängig vom momentanen FIFO-Inhalt sind. Ein Counter, sorgt für die korrekte Adressierung der Datenpakete – unabhängig wann diese vom PCI-Bus übertragen wurden. Von der Windows-Applikation können die Daten auf einmal oder in mehreren Bursts übertragen werden. Durch die höhere Übertragungsgeschwindigkeit des RAM-Moduls gegenüber dem PCI-Bus kann man davon ausgehen, dass die Daten unmittelbar nach Beendigung der PCI-Bus-Übertragung und Filter-Verzögerung (Pipelines) im RAM abgelegt wurden.

7. Implementierung des Gabor-FilterCores

Nachdem das Reference Desing an die Aufgabenstellung angepasst wurde, beschreibt dieses Kapitel die eigentliche Realisierung der Filterung.

Es wird zunächst der Filterungsablauf beschrieben. Danach wird die Hierarchie des Designs beschreiben und die Struktur des FilterCores erläutert.

7.1. Filterungsablauf

Die unterschiedlichen Filterungen (variiert in Orientierung und Frequenz) werden nacheinander vom ImageTool aus abgearbeitet, da hierfür neue Filterkoeffizienten benötigt werden und die zugehörigen Koeffizientenregister initialisiert werden müssen.

Die Bilddaten durchlaufen bei der zeilenweise Filterung (siehe Abb. 6 durchgezogenen Pfeile) zunächst einen Eingangs-FIFO der die Daten vom PCI-Bus abfängt. Der Block „FilterCore" (beinhaltet die IIR-Filter für Vorwärts- und Rückwärtsfilterung) arbeitet den FIFO-Inhalt ab und leitet die Filterantworten weiter zum Ausgangs-FIFO, der in diesem Fall die Daten dem DDR-Interface weiterleitet.

Sind alle Bildzeilen abgearbeitet erfolgt die Spaltenfilterung, die im „FilterCore" wie die Zeilenfilterung (gestrichelte Pfeile in Abb. 6) behandelt wird. Die Bildpunkte aus dem Zwischenergebnisbild werden lediglich in Spaltenrichtung vom externen RAM abgerufen und wieder dem ersten IIR-Modul zur Verfügung gestellt.

Am Ausgang des zweiten Filtermoduls erhält man nun die Filterantwort zum aktuell verwendeten Filterkern. Diese kann direkt über den PCI-Bus an die Windows-Applikation übertragen werden.

Nach einer Neukonfiguration der Koeffizientenregister der IIR-Filtermodule, beginnt eine neue Filterung mit einer neuen Übertragung des Eingangsbildes.

7.2. Der „FilterCore"

Im Block „FilterCore" (Abb. 7) sind zwei IIR-Filtermodule implementiert, wobei sich der Erste vom Zweiten nur geringfügig unterscheidet. Durch das erste IIR-Filtermodul wird die „Vorwärtsfilterung" und durch die zweite die „Rückwärtsfilterung" vorgenommen.

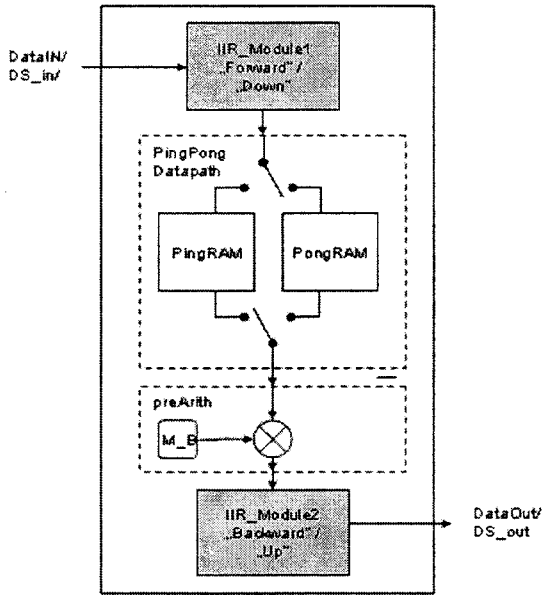


Abbildung 7: Block "FilterCore"

Dazwischen erfolgt eine interne Zwischenspeicherung im Doppelpuffer-Betrieb (auch Ping-Pong-Betrieb genannt), um eine Zeitverzögerung zwischen "Vorwärts"- und "Rückwärtsfilterung" zu vermeiden, da das zweite Filtermodul eine abgeschlossene Vorwärtsfilterung voraussetzt.

7.3. Implementierung eines IIR-Moduls

Die Struktur des Gabor-IIR-Filters stand durch den bestehenden C++-Code bereits fest. Dieser ist in Floating Point Format als rekursives komplexes Filter mit insgesamt 6 Koeffizienten realisiert.

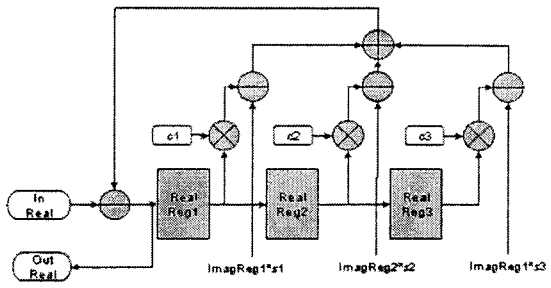


Abbildung 8: IIR-Filter (Grundstruktur)

Wie die Abbildung 8 zeigt, werden mehrere Stufen an algorithmischen Blöcken benötigt. Die Speicher RealReg1, RealReg2, RealReg3 (analog dazu ImagReg1, ImagReg2, ImagReg3) die die Ordnung des Filters darstellen, kann man funktionell direkt in einem VHDL-Modell übernehmen, nicht jedoch die Multiplizierer, Subtrahierer und Addierer.

Theoretisch benötigen arithmetische Blöcke keine Latenzzeit. Zeitverzögerungen werden in der theoretischen Beschreibung lediglich durch die Speicher dargestellt. Somit steht das Ergebnis eines Zustands unmittelbar an der Rückführung an, um beim nächsten Sample mit dem nächsten Eingangswert verrechnet zu werden.

Praktisch sind arithmetische Blöcke jedoch getaktete Module, die eine definierte Verzögerungszeit zwischen Ein- und Ausgang benötigen. In den implementierten Operatoren betragen diese 5 Takte bei den Multiplizierer und 8 Takte bei den Addierer/ Subtrahierer (Es ist geplant in einer weiteren Entwicklungsstufe des Filters die Addierer/Subtrahierer auf 5 Stufen zu senken).

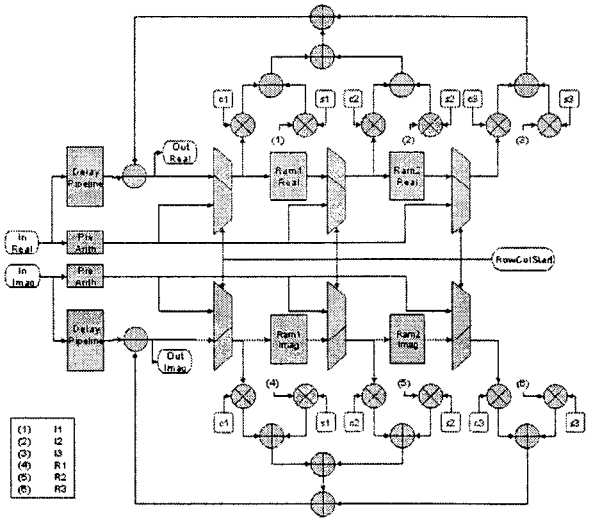


Abbildung 9: Implementierte IIR-Filterstruktur

Somit liegt das Ergebnis eines Zustands nach insgesamt 29 Takte, am Eingang des Rückkopplung-Subtrahierers an. Nach der Subtraktion des Ergebnisses mit dem neuen Wert und weiteren Zwischenstufen entsteht eine Gesamtverzögerung in der Filterschleife von 40 Takten. Während dieser Zeit darf die Faltung einer Zeile/Spalte nicht fortschreiten, da der nächste Wert am Eingang des Filters mit diesem Ergebnis verrechnet werden muss.

Um trotzdem mit jedem Takt einen Wert verarbeiten zu können, wurde die interne Pipeline des Filtermoduls im Zeilen(Spalten)-Multiplex betrieben, so dass 40 Zeilen(Spalten) praktisch gleichzeitig gefiltert werden (die Separierbarkeit des Filters ermöglicht es bei der Filterung in Zeilenrichtung jede Zeile getrennt zu behandeln - analog dazu werden die Spalten während der Filterung in Spaltenrichtung getrennt bearbeitet). Somit ist es möglich in einem Zeilen- bzw. Spaltenmultiplex-Betrieb die Latenzzeiten der Arithmetik zu kompensieren.

Die Filtermodule erwarten somit, bei einer Filterung in Zeilenrichtung, die Bildwerte in einer abgeänderten Reihenfolge, die einem Zeilenmultiplex für jeweils 40 Zeilen entspricht. Analog erfolgt bei der Filterung in Spaltenrichtung ("Down" und "Up") der Datenzufluss in Spaltenmultiplex.

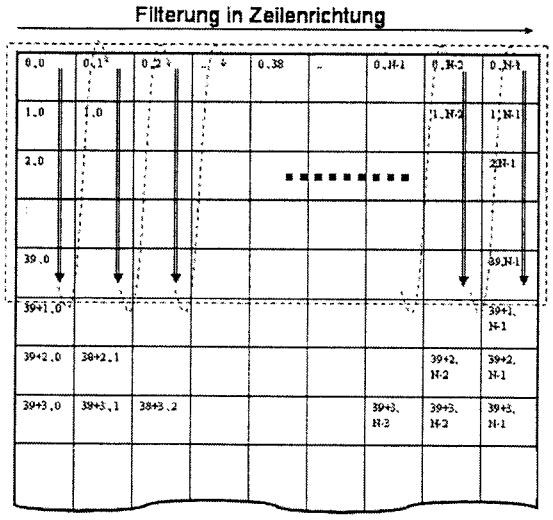


Abbildung 10: Bild-Pixelübertragung

Die Zustandsregister müssen ebenfalls für die parallel abgearbeiteten Zeilen getrennt zur Verfügung stehen. Dies wurde mittels RAM-Speicherblöcken mit 40 32-Bit Speicherplätzen und getrennten Read/Write-Ports realisiert. Die Startix-FPGAs bieten spezielle interne RAM-Blöcke an (TriMatrix-RAM), die keine zusätzlichen Logikelemente beanspruchen.

Während des Multiplex-Betriebs werden mit jedem Takt die RAM-Adressen am Read- und am Write-Port von Modulo40-Zähler angesteuert. Die RAM-Blöcke entsprechen somit 40 parallelen Registern im Multiplexbetrieb.

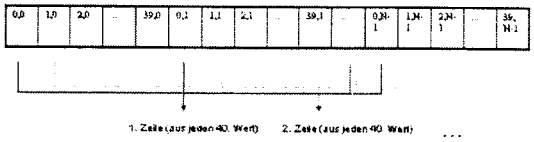


Abbildung 11: Bild-Pixelübertragung (seriell)

Eine weitere Besonderheit stellen die ersten Werte eines Filterungszyklus dar. Der Anfangszustand der Speicherblöcke muss initialisiert werden (Verbesserung des Einschwingverhaltens des Filters). Das IIR-Modul benötigt deshalb eine Vorlaufzeit von 14 Takten zur Berechnung des Initialisierungswerts aus den Eingangswerten (Abb. 12, InitArithmetic) und weitere 29 Takte Verzögerung für den Durchlauf der initialisierten Werte durch die Filterschleife. Dies wurde durch eine Pipeline am Filtereingang erreicht. Für die ersten 40 Werte (Anfangswerte von 40 Zeilen) werden die Eingangswerte auf 2 Pfaden geschaltet.

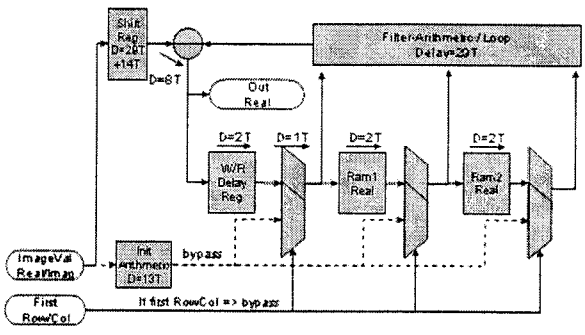


Abbildung 12: Bypass für Zustandsinitialisierung

Zum einen wird der Eingangswert in einer Verzögerungspipeline geschaltet, zum anderen wird aus Real- und Imaginärteil durch die „InitArithmetic" ein geeigneter Initialisierungswert berechnet. Ein Bypass-Switch

leitet das Ergebnis aus der „InitArithmetic" direkt zum arithmetischen Block.

Nach 29 Takten liegt das erste Ergebnis der Filterschleife am Eingang des Rückkopplung-Subtrahierers an - zusammen mit dem verzögerten Eingangswert aus der Eingangs-Pipeline. Weitere 10 Takte später liegt diese Differenz am Ausgang der ersten Speicherstufe an. Der nachgeschaltete Bypass-Switch hat gerade den letzten Initialisierungswert zum arithmetischen Block weitergeleitet, schaltet nun die Inhalte der Speicherblöcke zur Arithmetik und schließt somit die Filterschleife.

Die Rückkopplung verlangt dabei, dass innerhalb der Filterschleife alle Pipeline-Stages besetzt sind. Ungültige Werte sind nicht zulässig, da sich diese auf nachfolgende Ergebnisse auswirken würden. Diese Voraussetzung musste bei evtl. Datenpausen am Filtereingang berücksichtigt werden.

Das Ende der Filterung wird durch einen Counter ermittelt, der die Anzahl gültiger Datenwerte erfasst. Der Counter wird zu Beginn mit der erwarteten Anzahl an Bildpunkten initialisiert und bei jedem gültigen Datenwert dekrementiert. Jedes Filtermodul besitzt eine eigene interne Counter-Steuerung für den Filterungsfortschritt. Somit ist jeder Block absolut entkoppelt, und richtet sich ausschließlich nach den ankommenden gültigen Werten.

7.4. Doppelpuffer-Betrieb / Ping-Pong-Zwischenspeicher

Wie bereits in 7.2 erwähnt, wird das Filterergebnis aus dem ersten Gabor-Filtermodul im zweiten IIR-Modul rückwärts ein zweites Mal gefiltert. Die Ergebnisse aus dem ersten IIR-Filter müssen demnach komplett vorliegen, bevor das zweite Filtermodul starten kann.

Ein Ping-Pong-Betrieb ermöglicht, die Verzögerung des zweiten Filtermoduls auf die ersten 40 Zeilen (Spalten) zu beschränken. Der Zwischenspeicher besteht aus insgesamt 4 RAM-Blöcken (Ping/Pong für jeweils Real- und Imaginärteil) mit einer maximalen Aufnahmekapazität von 32000 Werten. (Bildkantenlänge x 40). Somit beschränkt dieser Block die maximale Bildbreite oder -höhe auf 800 Pixel. Man erkennt den direkten Zusammenhang zwischen maximal mögliche Bildgröße und Latenzzeit der Filterarithmetik (40 Zeilen/Spalten).

Zum Funktionsschema (Beispiel für Zeilenfilterung): Aus dem "Forward"-Filtermodul erhält man ein "Data Strobe"-Signal und die zugehörigen Filterantwort-Daten. Bei einer Bildbreite N erwartet der Zwischenspeicher nun Nx40 Werte, die im Ping-RAM abgelegt werden.

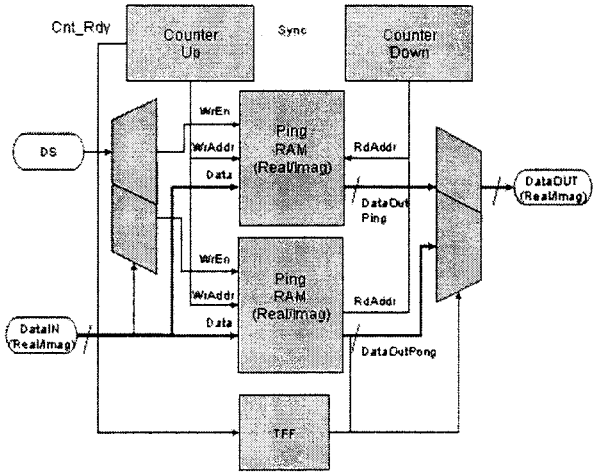


Abbildung 13: Doppelpuffer-Betrieb

Wurden Nx40 Werte erhalten, schaltet das Modul automatisch in Pong-Betrieb um, und schreibt die nächsten Daten Nx40 Werte ins "Pong-RAM". Gleichzeitig werden die Werte im Ping-RAM bereits in umgekehrter Reihenfolge am Ausgang des PingPong-Blocks dem zweiten IIR-Modul (Mit einem "Data Strobe") zur Verfügung gestellt. Dieser kann somit die Rückwärtsfilterung starten.

Bis auch das Pong-RAM erfolgreich beschrieben wurde, sind die Daten im Ping-Ram bereits abgearbeitet. Der Ping-Speicher kann somit mit den nächsten Nx40 Werten gefüllt werden, während wiederum der Pong-Speicher rückwärts ausgelesen wird. Somit entsteht nach einer Anfangsphase für das zweite IIR-Modul keine Zeitverzögerung mehr.

7.5. Auslastung des FPGAs

Die 2 Filtermodule (Forward / Backward) benötigen ca. 40000 LE's. Weitere 15000 LE's benötigen PCI-Controller, DDR Memory-Controller und interne Register sowie Control-Logic. Die Möglichkeit der Implementierung der Multiplizierer in DSP-Blocks wurde zu 100% ausgenützt (Das EP1S60-Startx-Device kann bis zu 18 Multiplizierer in 32bit-Format in DSP-Blocks implementieren, die keine LE's beanspruchen).

8. Ergebnisse

Untersuchungen haben ergeben, dass eine Beschleunigung des auf Gaborfilterbänken basierenden Objekterkennungsverfahrens mittels FPGA möglich ist.

Durch die aus Kostengründen vorgegebene Entwicklungs-Hardware mussten einige für die Performance ungünstige Kompromisse eingegangen werden.

Dadurch konnte das implementierte Gabor-IIR-Filter seine eigentliche Leistungsfähigkeit nicht komplett entfalten bzw. weitere Parallelisierungen aus Platzgründen nicht vorgenommen werden. Dennoch ist es nun anhand der ermittelten Informationen über Platzbedarf und Leistungsfähigkeit einzelner Filterkomponenten möglich, Bedingungen für eine angepasste Hardware zu definieren und Vorhersagen über die mögliche Performance zu treffen.

8.1. PCI-Schnittstelle

Die Filterstruktur wurde mit ALTERA Quartus II 3.0 bis 150MHz verifiziert. Somit wäre ein Datendurchsatz in 64Bit-Architektur von 9,6 Gbit/s denkbar.

Die Inbetriebnahme des Development-Boards auf einen Standard-PC mit 32Bit-PCI-Slot und 33MHz Bustakt konnte jedoch nur ein Bruchteil der möglichen Beschleunigung umsetzen. Die Filterungszeiten werden direkt durch den PCI-Bus begrenzt. Ein 800x600-Bild mit 8-Bit Grauwerten benötigt für die Filterung mit einem Filterkern ca. 4ms für die PCI-FPGA Übertragung und Zeilenfilterung. Jedoch weitere 29ms für die Spaltenfilterung mit anschließender Übertragung vom FPGA zur Windowsapplikation. Diese lange Übertragungszeit ist dadurch begründet, dass das Ergebnis aus zwei Bilder mit 32Bit-Werte besteht (Real- / Imag. Bild).

Simulationen wurden für Standard-PCI (32Bit, 33 MHz) und PCIx (64Bit, 66MHz) durchgeführt. Bei PCIx kann man die Performance durch den doppelten Takt und die doppelte Busbreite erheblich steigern. Hier wurden Filterungszeiten von ca. 3ms+7ms pro Filterung realisierbar.

Für PCIe konnten leider keine Simulationen durchgeführt werden, da kein PCIe-Compiler zur Verfügung stand. Durch PCI-Express dürfte aber eine weitere, erhebliche Leistungssteigerung möglich sein.

8.2. FPGA / DDR-Ram

Eine weitere Beschleunigung des Objekterkennungssystems kann durch die Parallelisierung der Zeilen- und Spaltenfilterung erfolgen. Diese setzt jedoch eine neue Hardwarekonfiguration, bestehend aus einem größeren FPGA und mindestens zwei getrennte DDR-Ram-Modulen voraus.

Bei einer Implementierung von zwei kompletten FilterCores werden ca. 100.000 Logic Elements benötigt, was moderne High-End FPGAs bereits besitzen.

Da die Spaltenfilterung ein komplettes Ergebnisbild aus der Zeilenfilterung erwartet, ist ein großer Doppelpufferbetrieb notwendig. Dies könnte durch 2 separate DDR-Module erreicht werden.

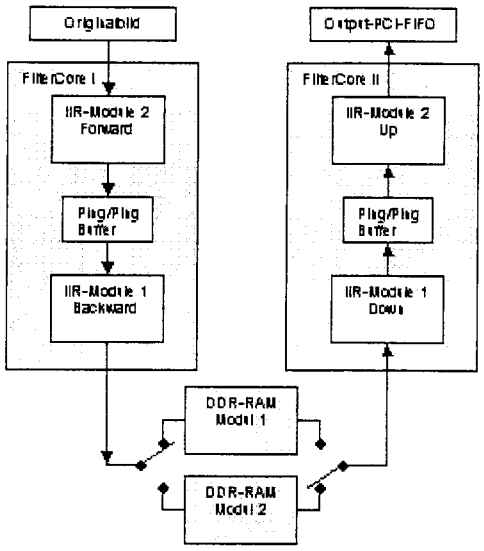


Abbildung 14: Optimale Hardwarekonfiguration

Dabei würde eine Verzögerung durch abwechselnde Zeilen- und Spaltenfilterung bei mehreren Filterkernen vermieden werden.

Eine solche Implementierung ermöglicht eine weitere, erhebliche Beschleunigung des Systems.

Tabelle 1 stellt die FPGA-Beschleunigung der rein PC-basierten Lösung gegenüber, für ein Grauwertbild von 800x600 Pixel, und 36 Gaborfilterkernen.

System	Laufzeit	Beschl.-Faktor
Standard-PC (1GHz)	ca. 2100ms	1
Implementiertes Design mit PCI	ca. 1100ms	1,9
Implementiertes Design mit PCIx	ca. 540ms	3,8
Aktueller PC (3 GHz)	ca. 1600ms	1,3
Aktueller FPGA mit optimierter HW-Konfiguration	ca. 200-250ms	8,5-10

Tabelle 1: Vergleich der Laufzeiten

9. Ausblick

Mittels Parallelisierung der Merkmalsextraktion lässt sich das entwickelte Objekterkennungssystem signifikant beschleunigen. Durch Verwendung aktueller und an den Algorithmus angepasster Hardware wäre eine schnellere Datenübertragung, höhere Parallelisierung und rein geschwindigkeitsoptimierte Realisierung möglich. Diese würde eine weitere starke Beschleunigung ermöglichen und das Verfahren für Aufgaben im Bereich der Echtzeit Anwendung interessant machen. Der neue PCI-Express Bus dürfte in Zukunft die notwendige Grundlage schaffen, die Performance des FPGA-Filters voll auszunutzen und das Objekterkennungssystem echtzeitfähig zu machen.

10. Literatur

[1] Heintz, R; Schäfer, Gerhard; Lokale invariante Objektlokalisierung mittels Gaborfiltern; GMA-Kongress; Vol.X, 2005, pp. XX-XX

[2] Kyrki, V.; Kamarainen, J-K.; Kälviäinen, H., Simple Gabor feature space for invariant object recognition, Pattern Recognition Letters, Vol.25, No.3, 2004, pp.311-318

[3 Frigo, Matteo, Johnson, Steven-G., The Design and Implementation of FFTW3, IEEE Proc. Sig.Proc., Vol.93, No.2, 2005, pp.216–231

[4] Young, Ian T., vanVliet, Lucas J., van Ginkel, Michael, Recursive Gabor Filtering, IEEE Trans. Sig.Proc. ,Vol.50, No.11, 2002, pp.2799–2805

[5] ALTERA Application Note AN223, PCI-to-DDR SDRAM Reference Design,Version 1.0, 2003

Untersuchung von Verfahren zum verlustleistungsoptimierten Entwurf von Schaltwerken

P. Kulle, R. Bartholomä, F. Kesel

Hochschule Pforzheim, Tiefenbronnerstraße 65, 75175 Pforzheim

E-Mail: frank.kesel@hs-pforzheim.de

Im Rahmen des durch das BMBF geförderten Projektes LEMOS ("Low-Power Entwurfs Methoden für mobile Systeme") [5] untersucht die Hochschule Pforzheim, als Unterauftragnehmer der Robert Bosch GmbH, Methoden zum verlustleistungsoptimierten Entwurf integrierter Schaltungen. Diese Arbeit entstand als Diplomarbeit im Rahmen des LEMOS Projektes.

Für die Optimierung von Schaltwerken auf Register-Transfer-Ebene gibt es in der Literatur zwei Ansätze. Zum einen das Partitionieren von Schaltwerken, zum anderen das Optimieren der Zustandskodierung, das detailliert untersucht wurde. Diese Untersuchungen und die gewonnen Ergebnisse sollen im Folgenden vorgestellt werden.

1. Einführung

In den letzten Jahren ist aufgrund anhaltend steigender Integration und mobiler Anwendungen die Leistungsaufnahme integrierter Schaltungen ein immer wichtigerer Entwurfsparameter geworden. Der Leistungsverbrauch einer integrierten CMOS Schaltung setzt sich aus zwei Komponenten zusammen. Zum einen dem dynamischen und zum anderen dem statischen Leistungsverbrauch. Ziel bei den beiden Verfahren zur Verlustleistungsoptimierung von Schaltwerken ist es, die dynamische Leistungsaufnahme zu reduzieren.

Für den dynamischen Leistungsverbrauch gilt:

$$P_{dyn} = \frac{1}{2} \cdot U_{DD}^2 \cdot f \cdot \sum_{i \in Nodes} C_i \cdot A_i$$

Dabei ist f die Schaltfrequenz, U_{DD} die Versorgungsspannung und C_i die am Knoten i umzuladende Kapazität. Die Schaltaktivität A_i beschreibt die Wahrscheinlichkeit, mit der das Eingangssignal seinen logischen Zustand innerhalb der Periode T ändert. Durch unerwünschte Schaltvorgänge (Glitches) kann die Schaltaktivität auch Werte annehmen, die größer als eins sind.

Neben der Frequenz, die häufig durch Spezifikationen oder andere Systeme vorgegeben ist, ist die Schaltaktivität A der einzige Parameter auf den der Entwickler beim Entwurf auf Register-Transfer-Ebene Einfluss hat. Die veröffentlichten Verfahren zum verlustleistungsoptimierten Entwurf integrierter Schaltungen haben die Reduktion dieser Schaltaktivität zum Ziel.

1.1. Ziel der Zustandskodierung

Die Zustandskodierung eines Schaltwerks kann hinsichtlich der Leistungsaufnahme optimiert werden. Ziel der Zustandskodierung ist es dabei, Zustände, die eine hohe Übergangswahrscheinlichkeit besitzen, so zu kodieren, dass deren Codeworte eine möglichst geringe Hamming-Distanz aufweisen.

Die Hamming-Distanz bezeichnet die Anzahl Bits, in denen sich zwei Codeworte derselben Länge unterscheiden. Für die Hamming-Distanz zwischen den Codeworten C und K, die jeweils l Bit lang sind, gilt folgende Gleichung:

$$HD(C,K) = \sum_{i=1}^{l} C_i \oplus K_i$$

Ausgangspunkt für die veröffentlichten Verfahren ist ein gewichteter, ungerichteter Graph $G_w = (V,E)$, dessen Knoten $s_i \in V$ die Zustände des Automaten modellieren. Die Kanten $\langle s_i, s_j \rangle \in E$ des Graphen mit den Gewichtungen w_{ij} beschreiben Zustandsübergänge, wobei w_{ij} aus den Betriebsdaten des Automaten ermittelte Größen sind. Die Gewichtung entspricht der Wahrscheinlichkeit des Zustandsüberganges.

Das Problem der Berechnung einer optimalen Zustandskodierung führt auf die Minimierung der so genannten Kostenfunktion:

$$C = \sum_{\langle s_i, s_j \rangle \in E} w_{ij} \cdot HD(c_i, c_j)$$

Dabei sind c_i und c_j die Codeworte der Zustände s_i und s_j und $HD(c_i, c_j)$ die Hamming-Distanz zwischen den Codeworten c_i und c_j.

Diese Optimierungsaufgabe ist vom Umfang der Problemstellung so komplex, dass für ihre Lösung im Wesentlichen nur heuristische Algorithmen in Frage kommen. Auf die untersuchten Algorithmen wird in den Kapiteln 2 eingegangen.

1.2. Untersuchte Schaltwerke

Für die Untersuchung wurden 22 unterschiedliche Schaltwerke verwendet. Diese stammen zum Großteil aus dem MCNC '89 Benchmark [6]. Außerdem wurde je ein Schaltwerk aus dem PREP Benchmark [8], einer Veröffentlichung [4] und ein selbst erstelltes Schaltwerk untersucht. Die untersuchten Schaltwerke haben zwischen 4 und 48 Zuständen und wurden so gewählt, dass sie möglichst gleichmäßig über den Testraum verteilt liegen. Um die Simulation einfach überwachen zu können wurde jedes Schaltwerk mit einem Ausgabeschaltnetz versehen, das den aktuellen Zustand ausgibt. Diese Vereinfachung beeinflusst die Tendenz der Ergebnisse nicht.

2. Algorithmen

Für die Zustandskodierung sind im Rahmen dieser Arbeit neben drei Standardverfahren (Binär-, Gray- und One-Hot-Kodierung) drei optimierende Algorithmen untersucht worden. Einer der Algorithmen wurde für den Test der gesamten MATLAB Umgebung selbst entwickelt. Bei den beiden anderen Algorithmen handelt es sich zum einen um einen heuristischen Ansatz, der auf dem bekannten Simulated Annealing Algorithmus basiert, zum anderen um den aus der Graphentheorie stammenden Hypercube Embedding Algorithmus. Damit wurde aus den beiden wesentlichen Familien der zur Zustandskodierung dienenden Algorithmen je ein Algorithmus ausgewählt. Die Algorithmen zur Zustandskodierung werden über die Schaltaktivität der Zustandsbits gesteuert.

2.1. Simple Algorithmus

Der Simple Algorithmus entstand zu Beginn der Entwicklung der MATLAB Umgebung, um diese mit einem sehr einfachen Algorithmus testen zu können. Er versucht, Zustände, die eine hohe Übergangswahrscheinlichkeit haben, mit Codeworten geringer Hamming-Distanz zu kodieren. Ausgangspunkt ist wie bei allen untersuchten Algorithmen ein gewichteter, ungerichteter Graph, der das Schaltwerk beschreibt. Die gewichteten Kanten des Graphen werden anhand einer nach der Gewichtung absteigenden Sortierung verarbeitet.

Bei der sequentiellen Abarbeitung der Liste der gewichteten Kanten können drei Fälle eintreten:

(a) beide Zustände der Kante sind bereits kodiert

(b) beiden Zuständen ist noch kein Codewort zugewiesen

(c) nur einer der beiden Zustände ist kodiert

Wurde beiden Zuständen bereits ein Codewort zugewiesen, so wird diese Zuordnung beim weiteren Abarbeiten nicht verändert. Wenn beiden Zustände einer Kante bisher keine Kodierung zugewiesen wurde, wird einem der Zustände ein beliebiger freier Code zugewiesen. Dann kann wie bei Fall (c) dem anderen Zustand ein freies Codewort zugewiesen werden, dessen Hamming-Distanz zum zuerst zugewiesenen Codewort möglichst gering ist. Ist nur einem der beiden Zustände einer Kante ein Codewort zugewiesen, so wird dem anderen ein freies Codewort zugewiesen, welches die geringste Hamming-Distanz zum bereits zugewiesenen hat.

Dieser Algorithmus betrachtet also immer nur die am stärksten gewichtete Kante eines Zustandes. Es besteht kein Einfluss anderer Kanten dieses Zustandes auf dessen Kodierung.

2.2. Simulated Annealing

Simulated Annealing, zu Deutsch simuliertes Abkühlen, ist ein weit verbreitetes, heuristisches Optimierungsverfahren und gehört wie die genetischen Algorithmen zur Gruppe der stochastischen Suchverfahren. Simulated Annealing ist an das natürliche Verhalten von Metallen bei ihrer Abkühlung angelehnt.

Bei der langsamen Abkühlung eines Metalls ordnen sich die einzelnen Moleküle so an, dass sie einen Zustand minimaler Energie einnehmen. Wird das Metall zu schnell abgekühlt, so fehlt den Molekülen die nötige Zeit, das tatsächliche Minimum zu finden und das System erstarrt in einem lokalen Minimum.

Um die optimale Lösung eines Optimierungsproblems zu finden, wird bei Simulated Annealing das Verhalten von Festkörpern bei langsamer Abkühlung simuliert [2], [7]. Der Einsatz von Simulated Annealing für die Kodierung von Zustandsautomaten wurde von Roy und Prasad in [9] beschrieben und wird auch in anderen Veröffentlichungen wie [4] verwendet.

Die Kodierung der Zustände entspricht der Teilchenkonfiguration bei der physikalischen Abkühlung. Der Ausgangspunkt für die Zustandskodierung mit Simulated Annealing ist eine

zufällige Zuweisung der Codeworte zu den Zuständen. Während der Phase der "Abkühlung" wird in jedem Schritt zufällig zwischen zwei Zügen entschieden. Zug (a) ist das Vertauschen der Codeworte zweier Zustände des Schaltwerks. Zug (b) das Ersetzen der Zustandskodierung eines Zustands durch ein nicht verwendetes Codewort. Damit wird die Kodierung, also die Teilchenkonfiguration, modifiziert.

Das Gegenstück zur Energie bei der physikalischen Abkühlung ist die Kostenfunktion beim Simulated Annealing. Die Kostenfunktion wird nach jeder Modifikation der Kodierung neu berechnet. Ob ein Zug akzeptiert wird, wird über die Differenz der aktuellen Kostenfunktion zum Ergebnis der Kostenfunktion für die vorige Kodierung bestimmt. Brachte der Zug eine Verbesserung der Kostenfunktion, wird er übernommen. Verschlechtert ein Zug die Kostenfunktion, wird er mit einer Wahrscheinlichkeit von $p = e^{\frac{-|\Delta C|}{T}}$ übernommen, wobei T die Temperatur und ΔC die Differenz der Kosten ist. Mit sinkender Temperatur nimmt also die Wahrscheinlichkeit ab, einen Zug zu akzeptieren, der das Ergebnis der Kostenfunktion verschlechtert. In einer frühen Phase der Abkühlung werden solche Züge noch mit einer höheren Wahrscheinlichkeit akzeptiert [9]. Schließlich wird ein sehr geringes und somit gutes Ergebnis für die Kostenfunktion erzielt.

Ein großer Vorteil von Simulated Annealing gegenüber anderen heuristischen Optimierungsverfahren ist seine Eigenschaft, nicht in lokalen Minima zu verbleiben. Aus lokalen Minima heraus kann sich der Prozess auf ein globales Minimum hin bewegen. Nimmt man eine unendlich lange Abkühldauer an, so findet man bei Simulated Annealing mit Sicherheit das globale Minimum [2]. Bei endlichen Abkühldauern kann dies nicht immer der Fall sein. Durch Anpassen von Parametern kann die Abkühldauer und somit die Laufzeit des Algorithmus bestimmt bzw. die Qualität der Ergebnisse beeinflusst werden.

2.3. Hypercube Embedding

Der Hypercube Embedding Algorithmus wird in [1] beschrieben. Die Autoren lösen das Problem der Zustandskodierung durch eine Abbildung des gewichteten und ungerichteten Zustandsgraphen auf einen so genannten Hypercube bzw. Hyperwürfel. Eine kurze Darstellung des Hyperwürfels findet man in [3].

Bei einem Hyperwürfel (Bild 1) handelt es sich um die Verallgemeinerung eines 3-dimensionalen Würfels auf d Dimensionen. Ein d-dimensionaler Hyperwürfel hat 2^d Knoten. Benachbarte Knoten eines Hyperwürfels, dass heißt Knoten, die durch eine Kante verbunden sind, unterscheiden sich in genau einem Bit der binären Darstellung ihrer Knotennummer. Die Bitstelle, an der sich die binären Darstellungen unterscheiden, ist die Linknummer. Diese bezeichnet die die Knoten verbindende Kante.

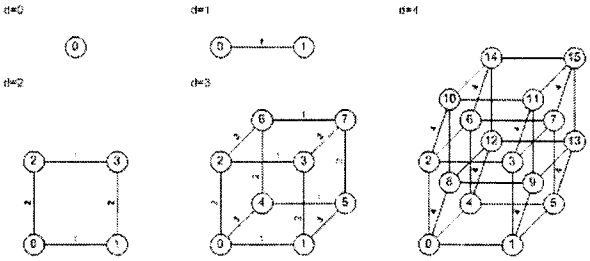


Bild 1: Hyperwürfel der Dimensionen d=1, 2, 3, 4

Ausgangspunkt des Hypercube Embedding Algorithmus ist erneut der gewichtete, ungerichtete Graph G_w. Ziel ist wieder die Minimierung einer Kostenfunktion, die von der Hamming-Distanz der Codewörter und den den Zustandsübergängen zugeordneten Gewichten abhängt. Dazu wird der Graph G_w auf einen d-dimensionalen Hyperwürfel abgebildet. Die Dimension d entspricht dabei der Codewortlänge, also der Anzahl der verwendeten Zustandsbits, die zur Kodierung verwendeten werden.

Bei dem Algorithmus werden ein Anfangszustand und ein Anfangscode frei gewählt. Dass bedeutet, ein beliebiger Zustand wird auf einen beliebigen Knoten abgebildet. Der nächste zu kodierende Zustand wird über die Auswahlfunktion

$$F(s_j) = \sum_{\langle s_i, s_j \rangle \in E_w, s_i \in X} w_{ij}$$

bestimmt. Die Auswahlfunktion beschreibt, wie stark die Nachbarschaft zwischen einem noch nicht kodierten Zustand und allen bereits kodierten Zuständen ist. Dazu ist X die Menge aller bereits kodierten Zustände, Y die Menge aller Zustände, denen noch kein Codewort zugewiesenen wurde, und E_w die Menge aller gewichteten Zustandsübergänge. Der Grad der Nachbarschaft wird über die Gewichtung der Zustandsübergänge, die durch w_{ij} beschrieben werden, bestimmt. Der im nächsten Schritt zu kodierende Zustand $s_j \in Y$ wird so gewählt, dass er die Auswahlfunktion maximiert.

Ist der nächste zu kodierende Zustand ausgewählt, muss ein Codewort für ihn bestimmt werden. Hierbei schlagen die Autoren vor, aus Performancegründen nur solche freien Knoten, also Codeworte, zu betrachten, die eine Hamming-Distanz von 1 zu einem bereits belegten Knoten/Codewort haben.

Diese Knoten werden als Kandidatenknoten bezeichnet. Für alle Kandidatenknoten wird die lokale Kostenfunktion C_{lokal} bestimmt.

$$C_{lokal}(c_j) = \sum_{\langle s_i, s_j \rangle \in E_w, s_i \in X} w_{ij} \cdot HD(c_i, c_j)$$

Der Knoten c_j, welcher die lokalen Kosten minimiert, bestimmt das Codewort für den mit der Auswahlfunktion bestimmten Zustand s_j.

Dies wird solange fortgeführt, bis alle Zustände kodiert sind. Unterschiedliche Ergebnisse ergeben sich je nach Auswahl von Anfangszustand und Anfangscode. Daher sollten unterschiedliche Anfangsbedingungen untersucht werden. Die Anzahl der möglichen Ansätze $m = N_{states} \cdot 2^{\lceil \log_2 N_{states} \rceil}$ wächst sehr schnell. Eine erschöpfende Untersuchung aller möglichen Ansätze ist daher nur bei kleineren Schaltwerken sinnvoll.

Entgegen der vorgeschlagenen Implementierung in [1] werden nicht nur die Kandidatenknoten bzw. -codes, sondern alle freie Codeworte mit der Kostenfunktion untersucht. Es hat sich gezeigt, dass die Auswahl der Kandidatenknoten und deren anschließende Untersuchung zu Laufzeiten führt, die länger sind als die Laufzeiten für die Untersuchung aller freien Codeworte. Mit dieser Anpassung sind die Laufzeiten des Hypercube Embedding Algorithmus besser oder vergleichbar mit denen von Simulated Annealing. Die Größe des Schaltwerks und somit die Anzahl der zu untersuchenden Anfangskonfigurationen m spielt dabei eine entscheidende Rolle.

3. Schaltaktivität

Neben den mittels der vorgestellten Algorithmen bestimmten Zustandskodierungen wurden für jedes Schaltwerk noch drei Zustandskodierungen mit Standardkodierverfahren erzeugt. Bei diesen drei Verfahren handelt es sich um die Binär-, die Gray- und die One-Hot-Kodierung.

Um vergleichbare Ergebnisse für die Schaltaktivität unterschiedlicher Automaten zu erhalten, wird bei der Auswertung für alle sechs Kodierungen die normierte Schaltaktivität bestimmt. Für die Bestimmung der normierten Schaltaktivität A_n nach folgender Gleichung

$$A_n = \frac{\sum_{i=2}^{N} HD(c_i, c_{i-1})}{N_{transitions}}$$

ist neben der Kodierung C zusätzlich der N Elemente umfassende Vektor nötig, der die Zustandsabfolge S des simulierten Automaten enthält. c_i steht für den Code des Zustands s_i. c_{i-1} ist der Code des vorigen Zustands s_{i-1} in der Zustandsabfolge S. $N_{transitions}$ beschreibt die Anzahl der Zustandswechsel in S.

Es wird die Summe der Schaltvorgänge aller Zustandsbits für den gesamten Eingangsvektor bestimmt. Diese wird auf die tatsächliche Anzahl von Zustandswechseln normiert, da bei den Schaltwerken nicht jede Eingangsbelegung in jedem Zustand zwangläufig zu einem Zustandswechsel führt.

Die optimale normierte Schaltaktivität beträgt 1 Bit pro Zustandswechsel. Das bedeutet, dass jeweils nur genau ein Zustandsbit pro Zustandswechsel schaltet. Eine Zustandskodierung mittels eines One-Hot-Codes bedeutet, dass pro Zustandswechsel immer genau zwei Zustandsbits schalten.

Jeder Automat wurde mit den drei zuvor genannten Standardkodierverfahren sowie den drei vorgestellten Algorithmen kodiert. Dafür wurde jeweils ein Pseudo-Random Eingangsvektor mit 1000 Eingangsbelegungen erzeugt. Kann trotz mehrerer Versuche keine Pseudo-Random Folge erzeugt werden, die alle Zustände des Schaltwerks abdeckt, so wird das Schaltwerk detaillierter analysiert und vor der Pseudo-Random Folge eine manuell erzeugte Folge eingefügt. Diese manuell erzeugte Eingangsfolge stellt sicher, dass jeder Zustand des Schaltwerks zumindest einmal angelaufen wird.

In Bild 2 ist die normierte Schaltaktivität für die untersuchten Schaltwerke dargestellt. Die Anzahl der Zustände im Schaltwerk steigt nach rechts an. Die normierte Schaltaktivität beträgt bei der One-Hot-Kodierung konstant 2 Bit pro Zustandswechsel. Für die Binärkodierung liegt sie zwischen 1,4 und 2,4 Bit pro Zustandswechsel. Die mittels Algorithmus optimierten Kodierungen weisen üblicherweise die geringste normierte Schaltaktivität auf. Unter den optimierenden Algorithmen sind die komplexeren diejenigen, die auch zu den besseren Ergebnissen führen. Bei drei Schaltwerken gelingt es den Algorithmen, Kodierungen zu finden, deren normierte Schaltaktivität dem theoretischen Minimum von 1 Bit pro Zustandswechsel entspricht.

Simulated Annealing und Hypercube Embedding ermöglichen eine durchschnittliche Reduktion der normierten Schaltaktivität gegenüber einer Binärkodierung von knapp über 30 %. Mit Hypercube Embedding ist eine durchschnittliche Ersparnis von 33 %, mit Simulated Annealing von 31 % möglich. Selbst der relativ einfache Simple Algorithmus erlaubt eine durchschnittliche Reduktion der normierten Schaltaktivität bezogen auf die Binärkodierung von

23 %. Die größte Ersparnis erzielt der Hypercube Embedding Algorithmus für das dvram Schaltwerk mit 47 %.

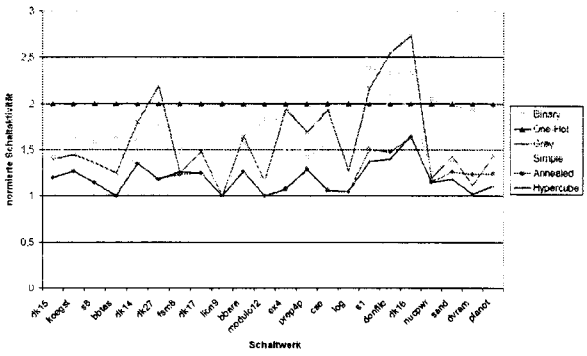


Bild 2: normierte Schaltaktivität

Für einige der Schaltwerke wurde mehr als ein Eingangsvektor erzeugt und ausgewertet. Die ermittelten Ergebnisse für die normierte Schaltaktivität unterscheiden sich dabei im Mittel nur um wenige Prozentpunkte. Somit können die für einen Pseudo-Random Eingangsvektor ermittelten Ergebnisse durchaus als typisch für das untersuchte Schaltwerk angenommen werden.

4. Leistungsabschätzung

Im vorigen Kapitel wurde auf die Untersuchung der reinen Schaltaktivität bei der Zustandskodierung eingegangen und gezeigt, dass die optimierenden Algorithmen eine Reduktion der Schaltaktivität ermöglichen. Die Schaltaktivität gibt allerdings keinen endgültigen Aufschluss über die Leistungsaufnahme eines Schaltwerks bei einer bestimmten Zustandskodierung. Hierzu fehlen Informationen zur Schaltaktivität im Überführungs- bzw. Ausgabeschaltnetz und somit zu der dort auftretenden dynamischen Leistungsaufnahme. Außerdem werden der Hardwareaufwand und die damit verbundene, durch Leckströme hervorgerufene, Leistungsaufnahme nicht berücksichtigt.

Unterschiedliche Zielplattformen beeinflussen sowohl die Umsetzung des Überführungs- bzw. Ausgabeschaltnetzes in Hardware bei der Synthese als auch den Ressourcenverbrauch. Darüber hinaus beeinflussen das Layout und das Routing die Leistungsaufnahme. Auch der Herstellungsprozess der Zielplattform hat, zum Beispiel über die Strukturgröße und die damit verbundenen Leckströme, Einfluss auf die Leistungsaufnahme.

Um bei der Leistungsabschätzung für FPGAs mit Xilinx XPower verwertbare Ergebnisse zu erzielen, wird jedes Schaltwerk mehrfach instanziert und mit identischen Eingangsvektoren simuliert. Damit liegt der Gesamtleistungsverbrauch der untersuchten Realisierung höher und auch relativ kleine Änderungen der Leistungsaufnahme eines Schaltwerks werden nicht durch gerundete Ausgaben des Programms maskiert.

4.1. Untersuchung für FPGAs

Der Ablauf einer Leistungsabschätzung mit XPower ist in Bild 3 dargestellt.

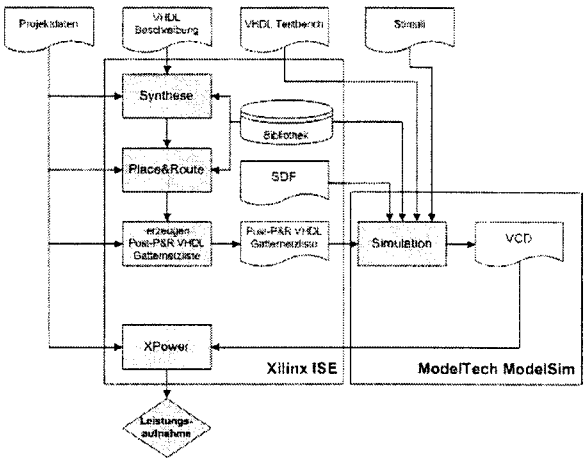


Bild 3: Leistungsabschätzung mit XPower

Die VHDL Beschreibung wird mit dem Xilinx Synthesewerkzeug XST synthetisiert. Nach dem Platzieren und Verdrahten erstellt die ISE eine VHDL Gatternetzliste als Simulationsmodell für die Timing Simulation. Bei der Simulation dieser Gatternetzliste mit ModelSim wird eine VCD Datei erzeugt, die Informationen zu den Schaltvorgängen der Signale enthält.

Auf Basis dieser Informationen bestimmt XPower für alle nicht in der VCD Datei enthaltenen Signale Werte für die Schaltaktivität. Neben anderen Projektdaten, wie der Verdrahtung, den Kapazitäten und dem Zielbaustein, dienen XPower die so ermittelten und die in der VCD Datei enthaltenen Schaltaktivitäten dazu, den Leistungsverbrauch zu ermitteln.

Die Ausgabe von XPower umfasst neben der Gesamtleistung die Teilleistungen der V_{ccint}, V_{ccout} und, sofern vorhanden, V_{ccaux} Netze und die zugehörigen statischen Leistungen. Diese Werte werden im Anschluss noch detaillierter aufgeschlüsselt nach einzelnen Teilen der Gatternetzliste ausgegeben. Für die spätere Auswertung wurde die dynamische Leistung des V_{ccint} Netzes betrachtet, da Schaltwerke meist für Steueraufgaben innerhalb der Schaltung eingesetzt werden.

Für die Untersuchung der Zustandskodierungen in FPGAs wurden dieselben 22 Automaten und 6 Kodierungen wie für die Untersuchung der reinen Schaltaktivität verwendet. Der Testvektor war jeweils derselbe, der auch für die Zustandskodierung mit MATLAB verwendet wurde. Dieser wird als typisches Betriebsverhalten des Schaltwerks angenommen. Wie schon ausgeführt, wurden die Schaltwerke mehrfach in den Bausteinen platziert, um besser auswertbare Ergebnisse zu erhalten. Je nach Komplexität des untersuchten Schaltwerks wurden zwischen 2 und 15 Instanzen erzeugt.

Als Zielbaustein wurden FPGAs der Spartan-II Reihe der Firma Xilinx gewählt. Stichprobenartig wurden auch Bausteine der Virtex-II Reihe untersucht. Zwischen den Ergebnissen beider Bausteine konnten keine wesentlichen Unterschiede festgestellt werden. Deshalb wurden nur für die Spartan-II Reihe alle Schaltwerke untersucht. Die Ergebnisse können aber auf andere FPGAs übertragen werden.

In Bild 4 ist für jedes Schaltwerk die ermittelte Ersparnis der dynamischen Leistung bezogen auf die Binärkodierung für eine Realisierung in einem Spartan-II FPGA dargestellt. Auf der x-Achse ist die Anzahl der Zustände aufgetragen. Dieses Diagramm kann, abhängig von der Anzahl der Zustände des Schaltwerks, in zwei wesentliche Bereiche unterteilt werden. Der erste Bereich umfasst alle Schaltwerke mit 16 und mehr Zuständen. Es ist wieder für alle untersuchten Kodierverfahren die gegenüber der Binärkodierung erzielte Ersparnis der dynamischen Leistung auf dem V_{ccint} Netz des FPGA dargestellt. Bei allen Schaltwerken mit 16 und mehr Zuständen erreicht man die größte Ersparnis gegenüber der Binärkodierung und somit die geringste Leistungsaufnahme des Schaltwerks durch eine One-Hot-Kodierung. In diesem Bereich ist mit einer One-Hot-Kodierung eine durchschnittliche Leistungsersparnis von 33 % möglich. Bei dem größten untersuchten Schaltwerk erreicht man bei einer One-Hot-Kodierung eine Leistungsersparnis von 51 % gegenüber der Binärkodierung.

Der zweite Bereich umfasst die Schaltwerke mit weniger als 16 Zuständen. Auch in diesem Bereich ist durch entsprechende Kodierung eine Leistungsersparnis von bis zu 26 % möglich. Allerdings kann aus den Ergebnissen kein klar zu favorisierendes Kodierverfahren ermittelt werden. Für Schaltwerke mit weniger als 16 Zuständen muss das beste Kodierverfahren individuell ermittelt werden. Hat das Schaltwerk zwischen 8 und 16 Zuständen kann das durchaus eine One-Hot-Kodierung sein. Bei weniger als 8 Zuständen liegen die anderen Kodierverfahren in Bezug auf die Leistungsersparnis vor der One-Hot-Kodierung.

Diese Ergebnisse decken sich mit den von Sutter et al. in [11] veröffentlichten Messergebnissen für FPGAs. Die Ursache für das gute Abschneiden der One-Hot-Kodierung bei einer Hardwarerealisierung der Schaltwerke in einem FPGA ist in der bei FPGAs eingesetzten Architektur zu suchen. In [10] sind die Quellen der Leistungsaufnahme eines FPGA basierend auf Layoutdaten der Firma Xilinx für die Virtex-II Baureihe untersucht worden. Dabei wurde die effektive Kapazität für die unterschiedlichen Komponenten eines FPGA ermittelt. Die effektive Kapazität einer LUT liegt dabei ungefähr um den Faktor neun über der eines Flip-Flops. Das bedeutet,

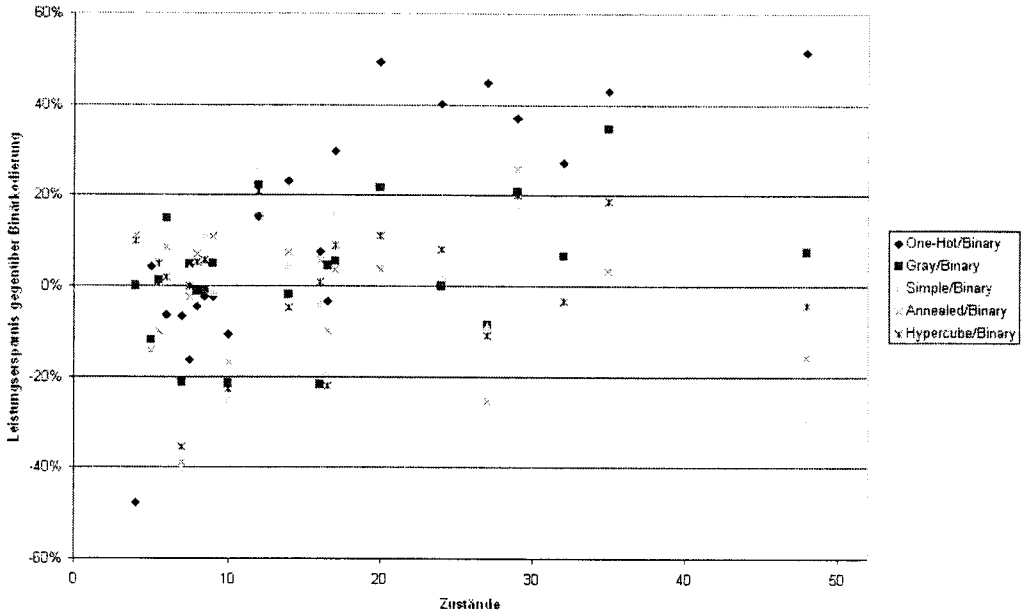


Bild 4: Leistungsersparnis gegenüber Binärkodierung

dass die Leistungsaufnahme einer LUT um den Faktor neun über der eines Flip-Flops liegt. Der Anteil des Routings an der dynamischen Leistungsaufnahme spielt aufgrund der relativ hohen Kapazitäten im Routing eine große Rolle [10].

Die für die One-Hot-Kodierung erzeugten Überführungs- und Ausgabeschaltnetze sind einfacher zu realisieren als für andere Kodierungen. Die Anzahl der benötigten Flip-Flops ist zwar groß, aber es werden weniger der leistungshungrigen LUTs benötigt. Dazu kommt, dass durch die weniger komplexen Überführungs- und Ausgabeschaltnetze auch das Routing auf dem FPGA einfacher wird. Insgesamt ist somit die One-Hot-Kodierung trotz größerer normierter Schaltaktivität für FPGAs mit großen Schaltwerken die Kodierung mit der geringsten Leistungsaufnahme. Durch die einfachen Überführungs- und Ausgabeschaltnetze ermöglicht die One-Hot-Kodierung außerdem höhere Taktfrequenzen.

4.2. Untersuchungen für ASICs

Die für FPGAs ermittelten Ergebnisse zeigen einen sehr starken Einfluss der spezifischen FPGA Architektur auf die Ergebnisse. FPGAs werden üblicherweise für Anwendungen mit kleinen Stückzahlen verwendet, wie zum Beispiel als Prototypen zum Funktionsnachweis. Für eine Serienproduktion mit großen Stückzahlen werden hingegen so genannte ASICs, Application Specific Integrated Circuits, eingesetzt. Diese weisen eine vollständig andere Hardwarearchitektur auf, die auf den Grundbauelementen der digitalen Logik und den so genannten Standardzellen basiert. So sind auch für die Leistungsabschätzung unterschiedlicher Zustandskodierungen andere Ergebnisse zu erwarten, als die im vorigen Kapitel gezeigten.

Deshalb wurden dieselben Automaten und Kodierungen wie zuvor auch für eine Realisierung als ASIC untersucht. Wieder wurden die bereits in MATLAB zur Kodierung verwendeten Eingangsvektoren als Stimuli benutzt.

Als Synthesewerkzeug diente der Synopsys Design Compiler. Für die Leistungsabschätzung wurden Synopsys PrimePower und als Simulator ModelSim von ModelTech eingesetzt. Simuliert wird mit einer direkt nach der Synthese erstellten VHDL Gatternetzliste. Die Simulation findet also noch vor dem Layout statt. In Bild 5 ist der Ablauf der Leistungsabschätzung dargestellt, wie sie für ASICs durchgeführt wurde.

Mit dem Synopsys Design Compiler wird die VHDL Beschreibung des Zustandsautomaten synthetisiert. Als Zielbibliothek wird dabei ein 0,5 µm Alcatel Mitec Semiconductor (AMIS) Prozess verwendet. Das nach der Synthese zur Verfügung stehende Design wird zum einen im Synopsys eigenen DB Format gespeichert, zum anderen als VHDL Gatternetzliste exportiert. Zuvor werden eventuell für VHDL nötige Anpassungen an Bezeichnungen vorgenommen. Die so erstellte VHDL Gatternetzliste wird mit den entsprechenden VITAL Bibliotheken, Stimuli und der zugehörigen Testbench in ModelSim simuliert. Dabei wird eine VCD Datei mit Informationen zu den Schaltaktivitäten aller Signale der Gatternetzliste erzeugt. Die VCD Datei, die leicht angepasste VHDL Gatternetzliste, die bei der Synthese erstellten DB Dateien sowie die Zielbibliothek dienen PrimePower zur Leistungssimulation.

Dieselben 22 Schaltwerke wurden wieder mehrfach in den Designs platziert. Bei der Auswertung wird die durch PrimePower ermittelte dynamische Leistung betrachtet. Für alle Schaltwerke wurde als Designziel ein Systemtakt von 50 MHz vorgegeben. Neben den ermittelten Ergebnissen für die dynamische Leistung wurden auch der Flächenbedarf sowie der kritische Pfad des synthetisierten Schaltwerks ausgegeben.

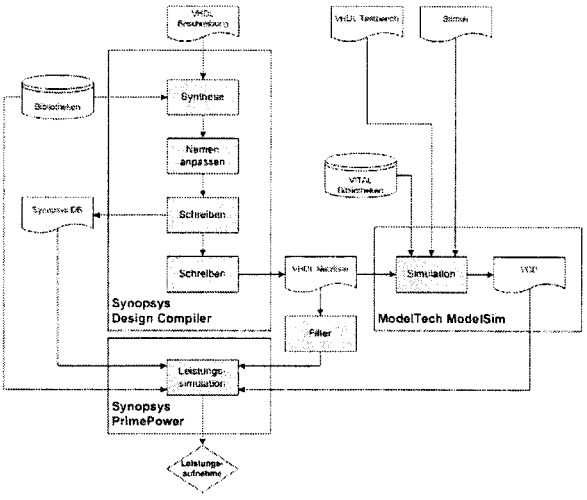


Bild 5: Leistungsabschätzung mit PrimePower

In Bild 6 ist die gegenüber einer Binärkodierung erreichbare Ersparnis an dynamischer Leistungsaufnahme für die untersuchten Kodierungen über der Anzahl der Zustände im Schaltwerk aufgetragen. Im Gegensatz zu den in Kapitel 4.1 vorgestellten Ergebnissen zeigt sich hier ein Ergebnis, das weitgehend unabhängig von der Anzahl Zustände im Schaltwerk ist. Die für die One-Hot-Kodierung erreichten Ergebnisse sind durchweg schlecht. Hier kann die One-Hot-Kodierung aufgrund der vollkommen anderen Hardwarearchitektur ihre Vorteile bei größeren Schaltwerken nicht ausspielen. Durchgehend gute Ergebnisse lassen sich mit den

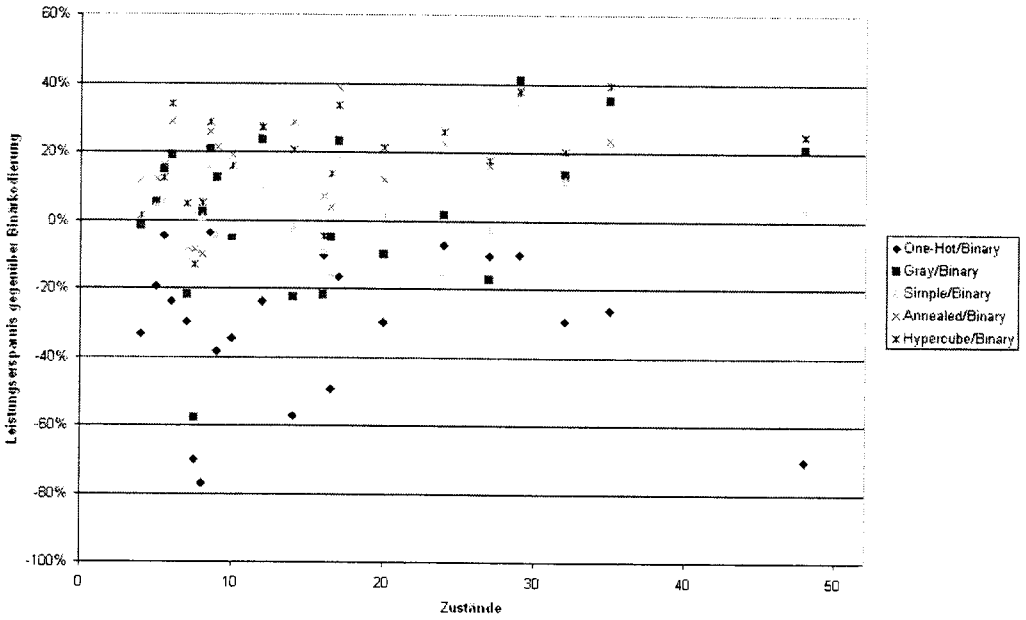


Bild 6: Leistungsersparnis gegenüber Binärkodierung

durch Simulated Annealing und Hypercube Embedding bestimmten Kodierungen erreichen.

Durch Simulated Annealing und Hypercube Embedding lässt sich die dynamische Leistungsaufnahme im Mittel um 17 % senken. Der Simple Algorithmus erreicht immerhin noch eine Reduktion um 5 %. Die One-Hot-Kodierung verschlechtert im Mittel die dynamische Leistungsaufnahme um 31 %. Maximal kann man durch Hypercube Embedding eine Reduktion von 40 %, durch Simulated Annealing von 39 % erreichen.

Bisher gab es bei der Synthese von Hardwarebeschreibungen die Möglichkeit, das Ergebnis hinsichtlich des Flächenbedarfs oder hinsichtlich der Geschwindigkeit zu optimieren. Eine Optimierung auf geringen Flächenbedarf geht mit einer Verschlechterung der maximal möglichen Geschwindigkeit einher und umgekehrt. Die Optimierung hinsichtlich des Leistungsbedarfs, zum Beispiel durch verlustleistungsoptimierte Zustandskodierung, ist ein dritter Parameter, der die beiden zuvor genannten beeinflusst. Deshalb wurde für die Realisierung als ASIC neben der Fläche auch das Timing der Hardwarerealisierung der unterschiedlichen Kodierungen untersucht. Im Mittel führt die Verwendung der durch die Optimierungsalgorithmen Simulated Annealing und Hypercube Embedding ermittelten Kodierungen zu einem um 7 % bzw. 8 % gestiegenen Flächenbedarf.

Für die ASIC Realisierung wurden alle Schaltwerke auf einen Takt von 50 MHz optimiert. Anhand der mittels report_timing erstellten Timing Berichte wurde für jede Kodierung basierend auf dem kritischen Pfad der maximal mögliche Takt ermittelt. Für Simulated Annealing und Hypercube Embedding lag dieser im Durchschnitt 2 % unter dem für die Binärkodierung bestimmten.

5. Zusammenfassung und Ausblick

Die Untersuchung von Zustandskodierungen hinsichtlich ihres Leistungsbedarfs geschah für 22 Schaltwerke und drei bekannte Standardkodierverfahren sowie drei mit unterschiedlichen Algorithmen optimierte Kodierungen. Der erste, sehr einfache, untersuchte Algorithmus wurde selbst entwickelt. Außerdem wurde jeweils ein Algorithmus aus den beiden wesentlichen Familien der in den Veröffentlichungen vorgeschlagenen Algorithmen untersucht. Die Algorithmen zur Zustandskodierung wurden in MATLAB implementiert. Sie gehören zu einer selbst erstellten MATLAB Umgebung zur Zustandskodierung. Mit dieser Umgebung kann auch die Schaltaktivität unterschiedlicher Kodierungen ausgewertet werden.

Die Leistungsabschätzung fand für unterschiedliche Hardwarearchitekturen statt. Dadurch bot sich zum einem der Vergleich der Zustandskodierung für die untersuchten Architekturen. Zum anderen konnten die beiden eingesetzten Leistungssimulatoren miteinander verglichen werden. Hier hat sich gezeigt, dass die kostenlos verfügbaren Werkzeuge für FPGAs aufgrund der Architekturunterschiede nicht für

eine Leistungsabschätzung bei anderen Architekturen, wie ASICs, eingesetzt werden können.

Als Zusammenfassung der Ergebnisse zeigt sich, dass die optimierenden Kodieralgorithmen zu einer Leistungsersparnis bei ASICs führen. Die Gesamtergebnisse der beiden wesentlichen untersuchten Algorithmen unterscheiden sich dabei nicht besonders stark. Für FPGAs mit größeren Schaltwerken hat sich gezeigt, dass die heute aufgrund des Timings bereits häufig eingesetzte One-Hot-Kodierung zu der geringsten Leistungsaufnahme führt. Bei kleineren Schaltwerken muss hier die beste Zustandskodierung für jedes einzelne Schaltwerk bestimmt werden.

Zusammenfassend lässt sich sagen, dass die Optimierung der Zustandskodierung durchaus zur Reduktion der dynamischen Leistungsaufnahme beitragen kann. Eine Implementierung direkt in Synthesewerkzeuge oder über zusätzliche Programme in den Entwurfsablauf scheint bei den in Kapitel 4.2 gezeigten Ergebnissen interessant. Bei der Integration in ein Synthesewerkzeug bestünde darüber hinaus auch noch die Möglichkeit, den Algorithmus zusätzlich den Flächenbedarf unterschiedlicher Kodierungen bewerten zu lassen. Damit könnte dann eine Optimierung hinsichtlich des Flächenbedarfes und somit der statischen Leistungsaufnahme sowie der Schaltaktivität und somit der dynamischen Leistungsaufnahme erreicht werden.

Literatur/Quellen

[1] De-Sheng, C.; Sarrafzadeh, M.; Yeap, G. K. H.: State Encoding of Finite State Machines for Low Power Design. In: *Proceedings of the 1995 IEEE International Symposium on Circuits and Systems*, Vol. 3, Mai 1995, S. 2309-2312

[2] Eberl, W.: *Verfahren zur nichtlinearen Optimierung* [online]. Letzte Aktualisierung: 15.04.1995, erhältlich im Internet unter: http://www.eberl.net/chaos/Skript/node48.html [Stand: 09.03.2005]

[3] Haase, G.: *Parallelisierung und Vektorisierung numerischer Algorithmen* [online]. Letzte Aktualisierung: 22.12.1998, erhältlich im Internet unter: http://www.numa.uni-linz.ac.at/Staff/haase/Lectures/parvor/parvor.html [Stand: 10.02.2005]

[4] Koegst, M.; Franke, G.; Feske, K.: State Assignment for FSM Low Power Design. In: *Proceedings of the Conference on European Design Automation*, 1996, S. 28-33

[5] LEMOS Konsortium. *LEMOS: Low-Power - Entwurfsmethoden für mobile Systeme, Beschreibung* [online]. Letzte Aktualisierung: 20.08.2004, erhältlich im Internet unter: http://lemos.offis.de [Stand: 13.06.2005]

[6] Microelectronics Center of North Carolina (MCNC). *MCNC '89, Logic Synthesis and Optimization Benchmarks.* Erhältlich im Internet unter: http://web.archive.org/web/19971024145958/www.cbl.ncsu.edu/CBL_Docs/lgs89.html [Stand: 09.03.2005]

[7] Natschläger, T.: *Maschinelles Lernen A : Simulated Annealing.* Graz, Technische Universität, Folien zur Vorlesung, Wintersemester 2001/2002

[8] Programmable Electronics Performance Corporation (PREP). *PREP Suite #1.3, PREP4 – Large State Machine VHDL.* Erhältlich im Internet unter: http://web.archive.org/web/19970129230005/http://www.prep.org/testbnch/synth.htm#prep [Stand: 09.03.2005]

[9] Roy, K.; Prasad, S.C.: Circuit Activity Based Logic Synthesis for Low Power Reliable Operations. In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 1, No. 4, Dezember 1993, S. 503-513

[10] Shang, L.; Kaviani, A. S.; Bathala, K.: Dynamic Power Consumption in Virtex™-II FPGA Family. In: *Proceedings of the 2002 ACM/SIGDA 10th International Symposium on Field-Programmable Gate Arrays*, Februar 2002, S. 157-164

[11] Sutter, G.; Todorovich E.; Lopez-Buedo, S.; Boemo, E.: Low-Power FSMs in FPGA: Encoding Alternatives. In: *12th Power and Timing Modeling, Optimization and Simulation Conference*, September 2002

Untersuchung von Verfahren zur Verlustleistungsoptimierung bei on-Chip Bussystemen

M. Strasser, M. Gaiser, F. Kesel
FH Pforzheim, Tiefenbronner Str. 65
Telefon: 07231 28-6567 (Prof. Dr.-Ing. Frank Kesel)

Die Minimierung der auf einem Chip umgesetzten Verlustleistung nimmt einen immer höheren Stellenwert bei der Entwicklung von hoch integrierten Systemen ein, wobei primäre Ziele zum einen die Verlängerung der Betriebsdauer von portablen Geräten durch Senkung der Stromaufnahme, sowie zum anderen die Beherrschbarkeit der Wärmeabführung zum Beispiel bei Hochleistungsprozessoren sind.

Versorgungsspannung, Taktfrequenz und die im Chip vorhandenen Kapazitäten sind als in höheren Entwurfsebenen nicht beeinflussbare Parameter in der Regel vorgegeben. Somit bleibt als einziger optimierbarer Parameter zur Verringerung der Leistungsaufnahme die Schaltaktivität auf den einzelnen Leitungen zu reduzieren. Auf Grund der um Größenordnungen höheren Lastkapazitäten von Leitungen außerhalb von Halbleitern im Vergleich zu chipinternen Leitungen wurden bisher vorrangig Buskodierverfahren entwickelt, die diese sehr hohen Leitungskapazitäten zu Grunde legen. Solche Verfahren auch für chipinterne Busse, so genannte on-chip Busse einzusetzen, wurde lange Zeit nicht in Betracht gezogen.

Diese Arbeit untersucht bekannte Verfahren und ermittelt bei den gegebenen Rahmenbedingungen Kenngrößen, die letztendlich zur Berechnung von Einsparpotentialen und Leitungslängen dienen.

1. Einleitung

1.1. Ursachen für Verlustleistung

Die in einem CMOS-Schaltkreis, z.B. in einem Inverter umgesetzte dynamische Verlustleistung ist gegeben durch:

$$P_V = \frac{1}{2} U_{dd}^{\ 2} f C_L \alpha$$

Wobei U_{dd} die Versorgungsspannung, f die Taktfrequenz, C_L die Lastkapazität und α die Schaltaktivität des Gatters darstellen.

Setzt man einen Inverter als Bustreiber ein, so stellt die Kapazität der angeschlossenen Busleitung sowie die Eingangskapazitäten aller an diese Leitung angeschlossenen Gatter die Lastkapazität C_L des Inverters dar. Die klassische Modellierung einer Leitung auf einem Chip als Plattenkondensator zur Ground-Platte kann bei heute gängigen Technologieprozessen so nicht mehr angewandt werden. Eine genauere Betrachtung der tatsächlichen Gegebenheiten ist nötig.

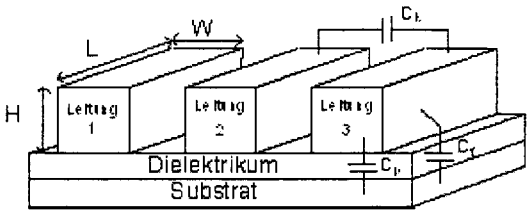


Abbildung 1.1: Modell für Leitungen auf einem Chip

Um den Widerstand einer Leitung, der durch das verwendete Material und den Querschnitt der Leitung bestimmt wird, nicht zu groß werden zu lassen, wird neben der Verwendung von geeigneten Materialien wie z.B. Kupfer, bei ständiger Reduzierung der Leiterbahnbreite W, die Höhe H immer mehr vergrößert. Dadurch entstehen auch Kapazitäten vom Rand der Leitung zur Ground-Platte, die so genannten Fringe[1]-Kapazitäten bzw. Randkapazitäten wie sie in Abbildung 1.1 dargestellt sind.

Zusätzlich liegen die einzelnen Leitungen immer dichter beieinander, weshalb die Koppelkapazitäten zwischen den Leitungen nicht mehr vernachlässigt werden dürfen. In modernen Prozessen nimmt die Koppelkapazität zwischen den Leitungen den größten Wert ein, gefolgt von den Randkapazitäten. Den

[1]Engl. Fringe: Rand, Randgebiet

kleinsten Beitrag zu der Gesamtkapazität einer Leitung liefert dann die Wire-to-Ground-Kapazität C_p.

1.2. Das Busmodell

Viele Veröffentlichungen wie z.B. [2] verwenden wie auch diese Arbeit ein Busmodell, um die gewonnen Daten über die Buskodierverfahren auf ein realistisches Szenario abbilden zu können.

Das klassische Busmodell setzt sich aus folgenden Komponenten zusammen:

1. Treiber
2. Leitung
3. Repeater
4. Leitung
5. Senke

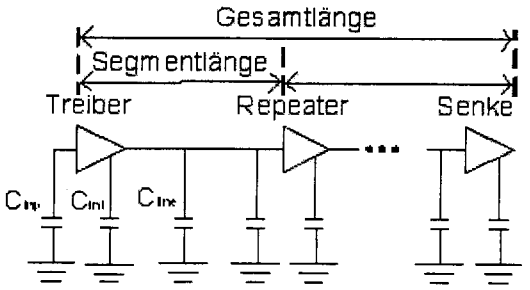


Abbildung 1.2: Schematisches Busmodell, nach [2]

Zur Vereinfachung wird oft davon ausgegangen, dass der Treiber, der Repeater und die Datensenke identisch aufgebaut sind, also auch identische Eingangskapazitäten C_{inp} aufweisen. Die Kapazität der Leitung C_{line} kann sich je nach Modell aus mehreren berücksichtigten Kapazitäten zusammensetzen.

Die Kapazitätswerte der einzelnen Komponenten hängen sehr stark von der verwendeten Technologie und der Dimensionierung der Bauteile ab. Die Gesamtkapazität des Busses pro Länge hängt wiederum von der Dimensionierung der Leitungen, der Treiber und deren Abstände zueinander ab. Es wurden drei Varianten eines Busses berechnet, die als minimale (min), durchschnittliche (avg) und maximale (max) Variante bezeichnet werden.

Mit Hilfe eines solchen Busmodells lassen sich dann Buslängen berechnen, ab denen der Einsatz eines Buskodierverfahrens in einem gegebenen System eine Leistungsersparnis erbringt oder nicht.

1.3. Kennzahlen zur Berechnung der Leistungsaufnahme

Eine Leistungsentnahme aus der Quelle findet nur bei einem Aufladevorgang der anliegenden Kapazitäten statt, also einem Zustandswechsel auf der Leitung von ‚0' nach ‚1'. Hier wird der Quelle schon die gesamte Energie entnommen, die für einen vollständigen Schaltvorgang, also entweder von ‚0' nach ‚1' nach ‚0' oder auch im anderen Fall von ‚1' nach ‚0' nach ‚1' benötigt wird.

Die auftretende **Leistungsentnahme** aus der Quelle bei einem Aufladevorgang ist dabei definiert als:

$$P_V = U_{dd}^{\ 2} f C_L \alpha$$

Im Folgenden wird immer die Leistungsentnahme aus der Quelle zu Grunde gelegt, da es einfacher ist, anstelle von jedem Zustandswechsel auf der Leitung nur Aufladevorgänge zu betrachten.

Bei gegeben Technologieparametern V,f und einer Lastkapazität pro Länge C/l kann mit der tatsächlichen Schaltaktivität α auf dem Bus bei gegebenen Eingangsdaten nun eine Verlustleistung pro Länge des unkodierten Busses angegeben werden. Die Angabe der Schaltaktivität (hier wieder bezogen auf Aufladevorgänge) liegt dabei zwischen 0,5 (Zustandswechsel bei jedem Takt, Aufladevorgang bei jedem zweitem Takt) und 0 (keine Zustandswechsel, keine Aufladevorgänge) und bezieht sich auf die über einen gewissen Zeitraum im Mittel aufgetretenen Aufladevorgänge pro Takt

Das α des durch ein Buskodierverfahren kodierten Busses, das einen deutlich geringeren Wert als beim unkodierten Bus aufweisen sollte, wird durch Messungen oder Simulationen ermittelt.

Die Angabe der **Kodiereffizienz E_α** setzt die Anzahl aller Aufladevorgänge auf dem unkodierten mit der Anzahl aller Aufladevorgänge auf dem kodierten Bus in Relation und stellt somit die Reduktion der Aufladevorgänge dar.

$$E_\alpha = 1 - \frac{\alpha_{coded}}{\alpha_{uncoded}}$$

Die **Leistungsaufnahmeersparnis pro mm** auf dem Bus berechnet sich aus:

$$\frac{P_{Bus}}{mm} = U_{dd}^{\ 2} \cdot f \cdot \left(\frac{C_{Bus,uncoded}}{mm}\alpha_{uncoded} - \frac{C_{Bus,coded}}{mm}\alpha_{coded}\right)$$

Die Kapazitäten des unkodierten und des kodierten Busses können sich aufgrund von zusätzlichen Leitungen unterscheiden. Dies muss bei der

Berechnung einer Leistungsaufnahmeersparnis berücksichtigt werden.

Die **Grenzbuslänge** stellt die Länge des Busses dar, ab der die auf dem Bus eingesparte Leistung genau der in den Codern zusätzlich benötigten Leistung entspricht. Ist der Bus länger als die Grenzbuslänge, liegt für das gesamte System eine Leistungseinsparung vor.

$$L_{Bus,Grenz} = P_{Coder} / (\frac{P_{Bus}}{mm})$$

Die **Grenzbuskapazität** drückt aus, ab welcher Gesamtkapazität aller Busleitungen sich der Einsatz des Buskodierverfahrens lohnt. Im Gegensatz zur Grenzbuslänge muss kein Modell eines Busses zugrunde gelegt werden.

$$C_{Grenz,bus} = P_{coder} / (U_{dd}{}^2 \cdot f \cdot (\alpha_{uncoded} - \alpha_{coded})$$

Die **Grenzleitungskapazität** stellt die Kapazität dar, die jede Leitung des Busses im Mittel aufweisen muss, wobei n_{lines} die Anzahl der Leitungen des unkodierten bzw. des kodierten Busses darstellt.

$$C_{Grenz,line} = P_{coder} / (U_{dd}{}^2 \cdot f \cdot (n_{lines,uncoded} * \alpha_{uncoded} - n_{lines,coded} * \alpha_{coded})$$

Die **nötige Buslänge** bei einer gegebenen Ersparnis x in Prozent errechnet sich nach

$$L_{Bus} = \frac{P_{Coder}}{((1 - \frac{x}{100}) \cdot P_{Bus,uncoded}) - P_{Bus,coded}}$$

1.4. Geeignete Buskodierverfahren

Aufgrund der besonderen Aufgabenstellung bei der Betrachtung von Buskodierverfahren auf On-Chip Bussystemen, wurden vier Verfahren zur weiteren Untersuchung ausgewählt. Auswahlkriterien dabei sind eine zu erwartende hohe Kodiereffizienz bei einer möglichst breit gestreuten Eingangsdatenverteilung sowie eine möglichst kompakte und effiziente Realisierungsmöglichkeit der Verfahren für ASIC Bibliotheken.

Zur Veranschaulichung der Funktionsweise eines typischen Buskodierverfahrens wird das Businvert-Verfahren detaillierter vorgestellt. Die drei anderen Verfahren sind das „spatially adaptive" oder kurz das SpatAd-Verfahren [1], dass „difference based mapping" oder kurz DBM in Kombination mit einem Invertfahren [4] und schließlich das „enhanced zone V2" oder kurz EZ2-Verfahren mit nachgeschaltetem Businvertverfahren [5].

1.4.1 Das Businvert-Verfahren

Das BI-Verfahren [3] stellt den Ausgangspunkt der Untersuchungen dar. Als eines der ersten veröffentlichten Verfahren bietet es einen entscheidenden Vorteil gegenüber den meisten anderen Verfahren: Encoder und Decoder mit einem sehr geringen Hardwareaufwand. Die Nachteile: nur mittelmäßige Kodiereffizienz sowie zusätzliche Signalleitungen nötig.

$$(B_{t+1}, INV_{t+1}) = \begin{cases} (b_t, 0) & \text{if } HD(b_t, B_t) + INV_t \le \frac{n}{2} \\ (\bar{b}_t, 1) & \text{otherwise} \end{cases}$$

B bezeichnet das aktuell auf dem Bus anliegende Datenwort, b das am Encoder anliegende Datenwort, HD die Hamming Distanz der in Klammern angegebenen Ausdrücke, INV den Zustand der Invertleitung, n die Breite des Busses inklusive der zusätzlichen Invertleitung.

Dem Encoder stehen zwei Möglichkeiten zur Verfügung, das neue Busdatenwort zu bilden: Entweder wird das anliegende Datenwort unverändert oder invertiert auf den Bus gelegt. Die Umschaltung des entsprechenden Multiplexers geschieht durch den Entscheider, der den größten Teil der benötigten Logik beansprucht.

Durch die zusätzlichen INV-Leitungen kann der BI-Decoder sehr hardwareeffizient aufgebaut werden. Er besteht nur aus einem Inverter und einem 2-fach Multiplexer pro Datenleitung. Die zu den entsprechenden Datenleitungen gehörende INV-Leitung schaltet alle Multiplexer zwischen dem unveränderten und dem invertierten Datenwort um.

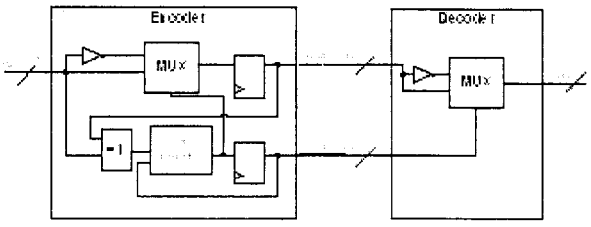


Abbildung 1.3: Blockschaltbild BI Buskoder

Realisiert wurde das BI-Verfahren mit zwei jeweils 8 Bit breiten, unabhängig voneinander arbeitenden Einzelcodern, jeweils einer für das High- und das Lowbyte des 16 Bit breiten Testdatenbusses. Diese Konfiguration führt zur besten Kodiereffizienz bei Einhaltung einer selbstgewählten Obergrenze von zwei zusätzlichen Leitungen. Die beiden Coder arbeiten unabhängig voneinander und benötigen jeweils eine eigene INV-Signalisierungsleitung.

Beim BI-Verfahren verschlechtert sich die Kodiereffizienz in der Regel mit größer werdenden Bitbreiten. Außerdem steigt der Aufwand für den Entscheider an.

2. Testmethodik

2.1. Verfahrensweise

Zur Ermittlung der Kodiereffizienz wurde in einem XILINX Spartan3 FPGA eine Testumgebung zur Messung der Kodiereffizienz der einzelnen Buskodierverfahren implementiert. Die Testdatensätze werden über eine serielle Verbindung von einem PC aus übermittelt. Die Auswertung geschieht auf dem PC unter MATLAB.

Die Berechnung der Leistungsaufnahme wurde mit PrimePower von Synopsys durchgeführt.

2.2. Testdatensätze

Die Grundlage aller Tests bilden die verwendeten Testdatensätze, die in Anlehnung an einen von der Firma Bosch verwendeten SOC-Datenbus entwickelt wurden.

Die Besonderheiten dieses Busses:

1. Es handelt sich um Audiodaten
2. Die Audiodaten liegen im Zweierkomplement (ZK) vor
3. Kein Busteilnehmer schreibt mehrere Datenworte am Stück auf den Bus (burst), nach einem Schreibvorgang eines Datenwortes folgt immer ein anderer Teilnehmer.

Durch diese Randbedingungen entsteht eine Datenstruktur auf dem Bus, in der kaum noch zeitliche Redundanz enthalten ist, da die aufeinander folgenden Datenworte aus unterschiedlichen Quellen stammen. Dies wurde bei der Auswahl der Kodierverfahren berücksichtigt.

Auf die Struktur der Daten wirkt sich weiterhin die Aussteuerung des Audiodatenstromes massiv aus. Eine geringe Aussteuerung und die Darstellung im Zweierkomplement haben eine sehr hohe Korrelation der oberen Bits zur Folge. Diese Eigenschaft kann von einigen Kodierverfahren sehr gut ausgenutzt werden, was sich in einer teilweise recht hohen Kodiereffizienz niederschlägt.

Test-daten-satz	Beschreibung
1	Nicht segmentiertes Audiofile, Popmusik
2	Nicht segmentiert, Sprache
3	Mischdatei, Beckenschläge und Popmusik
4	Mischdatei, Beckenschläge und Gesang
5	Mischdatei, Sprache und Beckenschläge
6	Mischdatei, Sprache und Akkustikgitarre
7	Mischdatei, Zufallsdaten und Popmusik
8	Mischdatei, Zufallsdaten, Sprache und Beckenschläge
9	Von Matlab generierte Zufallsdaten

Folgende drei grundlegende Blöcke lassen sich aus den Testdaten bilden:

1. Reine Audiodateien
2. Ineinander vermischte Audiodaten
3. Mit Zufallsdaten vermischte Audiodaten

Besonders die Datensätze des zweiten Blocks tragen der vorgegebenen Busarchitektur und der zu erwartenden Struktur der auf dem Bus liegenden Daten Rechnung.

Diese Kategorisierung wird bei den weiteren Betrachtungen zu Grunde gelegt.

3. Ergebnisse

3.1. Kodiereffizienz

Im Testsystem wurde die Kodiereffizienz durch das Aufaddieren aller Aufladevorgänge der unkodierten Eingangsdaten und aller Aufladevorgänge der Daten auf dem Bus ermittelt.

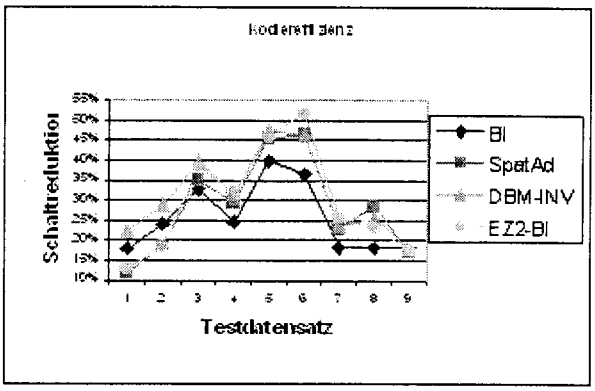


Abbildung 3.4: Kodiereffizienz

Der in Abbildung 3.4 dargestellte Verlauf der Kodiereffizienz der vier Verfahren über alle neun Testdatensätze zeigt eine deutliche Schwankung der Effizienz zwischen ca. 10 % und 50 %. Der Bereich mit der größten Relevanz zu dem von Bosch verwendeten Bus liegt im Block 2 (Testdatensätze zwei bis einschließlich fünf). Hier zeigen die getesteten Verfahren eine durchwegs gute Kodiereffizienz, im Schnitt liegen alle Verfahren hier bei ca. 38,9 %.

Es zeigt sich, dass die Kodiereffizienzen der einzelnen Verfahren doch erheblich voneinander abweichen. Im Block 2 beläuft sich die größte Abweichung von BI zu EZ2-BI auf absolute 8,5 %. Damit ist EZ2-BI in diesem Block 25 % effektiver als das BI-Verfahren. Diese Ergebnisse relativieren sich sehr schnell, wenn die im Kapitel 3.2 ermittelte Leistungsaufnahme der Coder mit berücksichtigt wird.

Interessant ist auch die im Block 3 durchgehend hohe Kodiereffizienz. Selbst bei reinen Zufallsdaten erreichen alle Verfahren noch eine Schaltreduktion von ca. 18 %. Schlechtere Ergebnisse werden nur bei den reinen Audiodaten erreicht. Dies kann mit der insgesamt sehr geringen durchschnittlichen Schaltaktivität der Eingangsdaten erklärt werden, wobei hier die Streuung der Kodierverfahren zueinander im Gegensatz zu z.B. reinen Zufallsdaten erheblich größer ist.

3.2. Leistungsbedarf der Coder

Die Leistungsaufnahme der Buskoder stellt eine wesentliche Größe in der Beurteilung eines Buskodierverfahrens dar. Die reine Betrachtung der Kodiereffizienz sagt nicht aus wie teuer (energetisch gesehen) diese erkauft wurde. Die Leistungsaufnahme eines Buskodierverfahrens hängt zum einen von der Komplexität der verwendeten Hardware und zum anderen von deren Schaltaktivität und damit von den Eingangsdaten ab. Aus diesem Grund wurde die Leistungsaufnahme für jedes Buskodierverfahren in Kombination mit jedem Eingangsdatensatz mit Hilfe von Synopsys PrimePower und einer 0,50 µm Zellbibliothek berechnet.

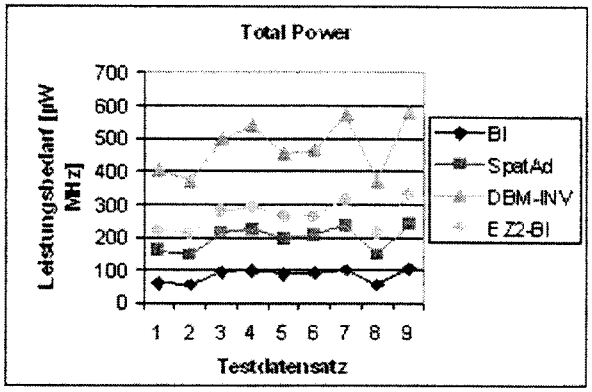


Abbildung 3.5: Auf µW / MHz normierte Leistungsaufnahme der Codecs

Deutlich stellen sich Unterschiede bei der in Abbildung 3.5 dargestellten Leistungsaufnahme der Buskoder bei den verschiedenen Testdatensätzen heraus. Das komplexe DBM-Verfahren benötigt z.B. im Schnitt die ca. fünffache Leistung des einfach aufgebauten BI-Verfahrens.

3.3. Benötigte Leitungslängen

Eine wichtige der ermittelten Größen stellt die Grenzleitungskapazität dar, weil sie die für jede Leitung eines Busses im Schnitt nötige Kapazität ausdrückt, ab der ein Buskodierverfahren Energie einsparen kann.

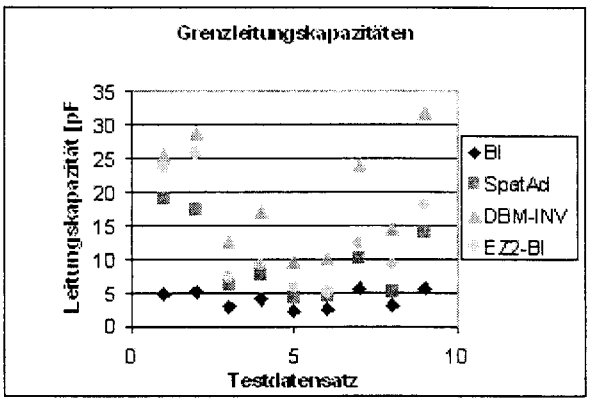


Abbildung 3.6: Grenzleitungskapazitäten aller Testdatensätze

Die in Abbildung 3.6 dargestellten Ergebnisse der Berechnung der Grenzleitungskapazitäten zeigen deutlich die Schwankung dieser Größe über die verschiedenen Testdatensätze und Buskodierverfahren hinweg. Bei den Datensätzen drei bis sechs weisen alle Verfahren eine relative geringe Grenzleitungskapazität auf, da sie auf diesen Bereich optimiert wurden. In den beiden anderen Testdatenblöcken kommt es zu sehr hohen und kaum realistischen Kapazitätswerten über 10 pF pro Leitung.

Dies verdeutlicht, dass die später über den Bus übertragenen Daten in Ihrer Struktur bekannt sein müssen, um ein optimales Verfahren bestimmen zu können. Die hier vorgestellten Buskodierverfahren arbeiten zwar auch bei Testdaten wie dem reinen Rauschen noch mit einer Reduktion der Schaltvorgänge, allerdings ist diese wesentlich niedriger.

Auf Basis des Busmodells lassen sich Grenzbuslängen errechnen, die ebenfalls stark von den Eingangsdaten abhängen. Die Angabe der Grenzbuslängen in Millimetern zeigt, in welchen Dimensionen sich die nötigen Längen für eine positive Energiebilanz eines solchen Systems bewegen. Selbst bei der diesen Daten zu Grunde liegenden 0,5 µm Technologie sind solche Buslängen kaum zu erreichen.

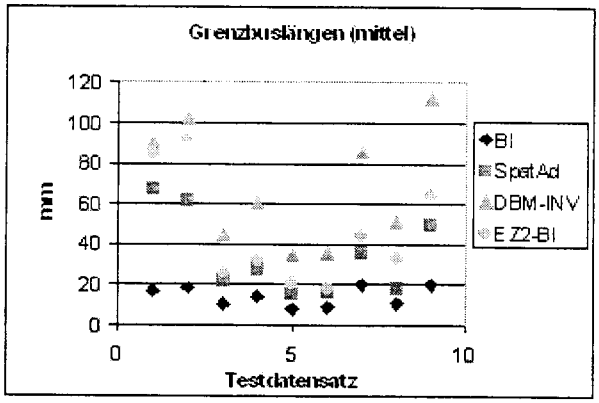


Abbildung 3.7: Grenzbuslängen auf Basis des durchschnittlichen Busmodells

Abbildung 3.7 zeigt das BI-Verfahren als Spitzenreiter bei allen Testdatensätzen. Aufgrund des sehr geringen Hardwareaufwandes in den Codern und einer im Schnitt mittelmäßigen Kodiereffizienz begnügt sich dieses Verfahren schon mit relativ kurzen Buslängen. Der Minimalwert wird beim Testdatensatz fünf (Mischdatei, Sprache und Beckenschläge) erreicht und beträgt 7,9 mm.

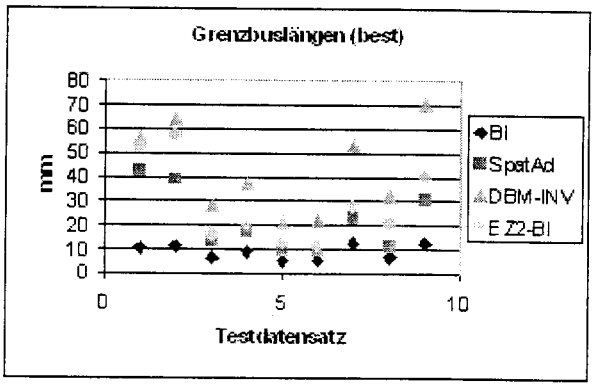


Abbildung 3.8: Grenzbuslängen mit „best case" Busmodell als Basis

Etwas anders stellt sich die Situation bei Betrachtung des „max" Busmodells mit der größten Leitungskapazität dar. Hier sinken die Grenzbuslängen bis auf einen Minimalwert von 5,0 mm.

4. Zusammenfassung

In dieser Arbeit konnte gezeigt werden, wie wichtig der Zusammenhang zwischen den über den Bus zu übertragenden Daten und die Auswahl des dafür passenden Buskodierverfahrens ist.

Die isolierte Betrachtung von Schaltvorgängen und Eigenkapazitäten einer Leitung stellt bei den heutigen Technologiegrößen im Halbleiterbereich keine zufrieden stellende Abbildung der Wirklichkeit mehr dar. Eine umfassende Betrachtung aller auftretenden Koppelkapazitäten, die auf die Busleitungen einwirken, ist allerdings viel zu komplex, um von einem Buskodierverfahren angemessen berücksichtigt werden zu können.

Der einzige Weg, die Leistungsaufnahme eines Bussystems deutlich zu verringern, stellt daher die Reduktion die Schaltvorgänge auf den Leitungen dar. Wie in weiteren Untersuchungen gezeigt werden konnte, senkt eine hohe Kodiereffizienz schon die Koppelvorgänge der Busleitungen untereinander in erheblichem Maße, ohne dass diese speziell berücksichtigt wurden.

Betrachtet man die für eine positive Energiebilanz nötigen Leitungskapazitäten, so befindet man sich bei der hier zu Grunde liegenden 0,5 µm CMOS Technologie bei Werten zwischen ca. 2 und 10 pF pro Leitung. Mit dieser Aussage kann ein Systemdesigner oder auch eine EDA Software entscheiden, ob für ein

vorliegendes Design ein Buskodierverfahren verwendet werden soll oder nicht.

Die Angabe von Grenzbuslängen ist hier nicht so aussagekräftig, da die Ergebnisse sehr stark vom verwendeten Busmodell abhängen. Die Längenangaben in Millimetern erlauben aber eine wesentlich bessere Vorstellung der nötigen Buslängen, die hier im Bereich von ca. 5 mm bis 40 mm für die betrachteten Testdatensätze liegen.

4.1. Ausblick

Es bleibt die Frage, wie sich die Gegebenheiten bei der heute stark fortschreitenden Technologieskalierung verhalten. Im High-Tech Bereich sind heute Strukturgrößen von 90 nm üblich, 65 nm sind der nächste Schritt. Für diese Technologiegrößen konnten keine Simulationen durchgeführt werden, da keine Zellbibliotheken zur Verfügung standen.

Zu erwarten ist aber, dass sich die Energiebilanz zu Gunsten der Buskodierverfahren hin verschiebt. Die dynamische Leistungsaufnahme in den Codern sinkt quadratisch mit der Abnahme der Versorgungsspannung. Dies gilt natürlich auch für die auf den Leitungen umgesetzte Verlustleistung. Hier ist aber eine zunehmende Gesamtkapazität durch die starke Zunahme der Koppelkapazitäten zu erwarten.

Immer komplexer werdende Chipdesigns führen zusätzlich trotz immer kleiner werdenden Strukturen zu relativ konstanten Chipgrößen und damit auch Buslängen.

Der rentable Einsatz von Buskodierverfahren scheint realistisch. Durch eine Leistungssimulation der Coder mit einer entsprechenden Bibliothek und einem Modell der Buskapazitäten lassen sich auch hier detaillierte Aussagen treffen.

Ein anderes Problem stellt allerdings die durch Leckströme in den Transistoren verursachte Erhöhung der statischen Stromaufnahme dar. Dieses Problem kann bei den hier vorgestellten Buskodierverfahren beispielsweise durch das Abschalten von nicht benötigten De- und Encodern reduziert werden.

4.2. Literaturverzeichnis

[1] A. Acquaviva, R. Scarsi, "A Spatially-Adaptive BusInterface for Low-Switching Communication", 2000

[2] C. Kretzschmar, A. Nieuwland, "Why Transition Coding for Power Minimization of on-Chip Buses does not work", 2004

[3] M. Stan und W. Burleson, „ Bus-Invert Coding for Low-Power I/O," 1995

[4] S. Ramprasad, N. Shanbhag, " A Coding Framework for Low Power Address and Data Busses", 1999

[5] T. Lang, E. Musoll, „Extension of the Working-Zone-Encoding Method to Reduce the Energy on the Microprocessor Data Bus", 1998

HMD – HEAD MOUNTED DISPLAY

Entwurf einer Wireless Kommunikationskomponente

Daniel Ziegler, Prof. Dr. Manfred Bartel
HTW Aalen, EDA Zentrum, Beethovenstraße 1, 73430 Aalen
Tel. 07361 / 576 - 247, Fax 07361 / 576 - 324
manfred.bartel@htw-aalen.de

Im Rahmen einer Diplomarbeit wurden der Aufbau und die Entwicklung eines Frequenzsynthesizers beschrieben, der in WLAN-Systemen zum Einsatz kommt. Dieser wurde mit Hilfe der Prozessfamilie ‚SGC25x' des Instituts für Halbleiterphysik, Frankfurt-Oder (IHP) realisiert, eine 0,25µm Prozessfamilie, die auf modernster SiGe:C BiCMOS-Technologie basiert. Verifiziert und auf Funktionstüchtigkeit überprüft wurde die Komponente mit dem Entwurfsbausatz (Design Kit) vom IHP, der wiederum auf dem Cadence™ EDA-System basiert. Der Bausatz ist optimiert für die Entwicklung von analogen Hochleistungs-Schaltkreisen wie zum Beispiel Glasfaser-kommunikationskomponenten oder drahtlosen Anwendungen.

In einer ersten Phase wurden potentielle Übertragungsstandards, die heute für eine drahtlose Datenübertragung zur Verfügung stehen sowie der Stand der WLAN-Technik untersucht. Anhand einer Spezifikation der Firma ZEISS wurde ein Standard gewählt, der alle Anforderungen erfüllt.

Danach wurden alle Schaltungsblöcke des Synthesizers entwickelt, um sie anschließend als Bibliothekselemente zur Verfügung zu stellen.

1. Einleitung

1.1. Motivation

Heutige drahtlose Übertragungssysteme erfordern hohe Datenraten, große Betriebsfrequenzbereiche, hohe Übertragungsqualitäten und viele Übertragungskanäle bei gegebener Bandbreite. Um den Vorteil der drahtlosen Übertragung nicht einzuschränken, sind noch weitere Faktoren wie geringer Stromverbrauch, geringes Gewicht, geringe Größe und hohe Integrationsdichte für den Erfolg einer Komponente von entscheidender Bedeutung. Erwünschenswert ist die Integration von möglichst allen Teilen auf einem Chip. Alle diese Anforderungen zu erfüllen machen den Entwurf einer Wireless Kommunikationskomponente zu einer großen Herausforderung. Am Ende des Prozesses steht eine SoC[1]-WLAN Lösung wie in Abbildung 1.

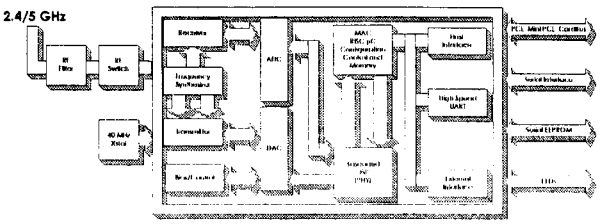


Abbildung 1 Blockschaltbild einer Basisband-SoC-WLAN-Lösung

Ein Wireless Local Area Network verwendet eine Funkfrequenztechnologie, um Daten per Radiowellen oder auch per Infrarotlicht drahtlos zu übertragen. Der heutzutage am häufigsten verwendete Standard zur drahtlosen Datenübertragung ist der Standard IEEE 802.11, der 1997 verabschiedet wurde, wobei die Erweiterungen 802.11a und 802.11b (1999 verabschiedet) im 5 GHz-Bereich bzw. im 2,4 GHz-Bereich arbeiten.

Das Blockschaltbild eines Transceivers[2], der Sender (Tx) und Empfänger (Rx) zugleich ist, zeigt Abbildung 2. Im Anschluss an die Antenne, die zum Abstrahlen bzw. Empfangen der Funkwellen dient, befindet sich ein Duplexer, vereinfacht als Schalter dargestellt. Dieser dient dazu, den sehr empfindlichen Empfangspfad während des Sendens von der Antenne und dem Sendepfad mit dessen enorm verstärkten Signalen abzukoppeln und umgekehrt, wenn empfangen werden soll. Ein gleichzeitiges Senden und Empfangen ist deshalb nicht möglich. Nachdem das empfangene Signal den Duplexer passiert hat, wird es zunächst von einem Bandpass gefiltert, der alle unerwünschten Frequenzanteile, also die die keine Informationen enthalten, entfernt und

[1] SoC – System on Chip

[2] engl.: transmitter and receiver (TxRx)

3.1. Phasen-/Frequenz-Detektor

Er besteht aus zwei flankengesteuerten, rücksetzbaren D-Flipflops (D-FFs) und einem UND-Gatter, das den Reset-Eingang der D-FFs treibt (siehe Abbildung 5). Die beiden Eingänge D sind an log. ‚L' angeschlossen. A und B dienen als Takteingänge der Flipflops. Dadurch setzt jedes D-FF bei steigender Flanke an seinem Takteingang CK seinen Ausgang Q auf log. ‚L'.

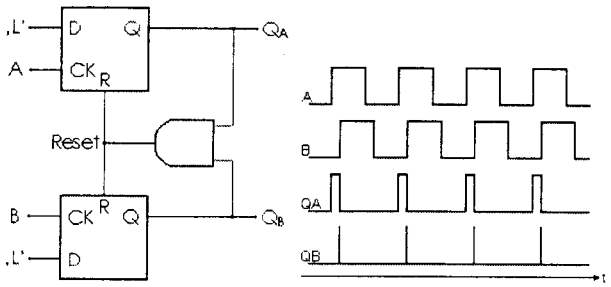


Abbildung 5 PFD: Implementierung und Zeitdiagramm [#RAZA01]

Wenn zunächst $Q_A = Q_B =$ ‚0' und A nach ‚L' wechselt, wechselt Q_A auch auf ‚L'. Falls dieses Ereignis gefolgt wird von einem Wechsel von ‚0' auf ‚L' des Eingangs B, wird Q_B ebenfalls ‚L', ein verbotener Zustand ist eingetreten. Infolgedessen sorgt das UND-Gatter für einen ‚Reset' der beiden Flipflops (Q_A und Q_B gehen auf ‚0' zurück), d.h. Q_A und Q_B sind beide für eine kurze Zeit log. ‚high', was an der unvermeidbaren Verzögerungszeit des UND-Gatters und der spezifischen Zeit, die ein D-FF benötigt, um sich rückzusetzen liegt.

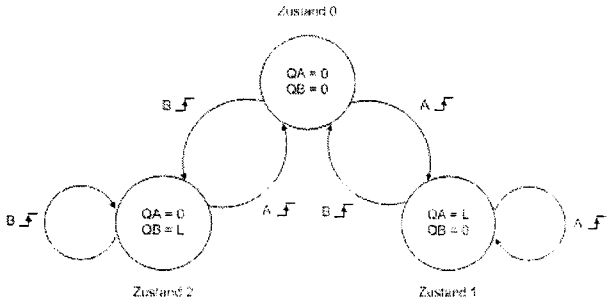


Abbildung 6 Zustandsdiagramm des PFD [#RAZA98]

Die Breite der Pulse ist gleich dem Phasenunterschied zwischen den beiden Eingängen A und B. Dieses Verhalten des PFD gilt auch analog für Frequenzunterschiede an seinen beiden Eingängen. Am Ausgang Q_A erscheinen also immer dann positive Pulse, wenn B gegenüber A hinterherhinkt oder wenn $\omega_A > \omega_B$.

Der Schaltkreis verwendet sequentielle Logik, um drei Zustände zu erzeugen.

3.2. Ladungspumpe / Schleifenfilter

Die Ladungspumpe (CP) dient dazu, die beiden digitalen Ausgangssignale UP und DOWN des PFD in Ladungsströme umzuwandeln, deren Größe proportional zum Phasenfehler ist. Ein passiver Filter formt dann das Ausgangs-Stromsignal der CP um, um die wertlosen Informationen, die in diesem Signal enthalten sind zu unterdrücken.

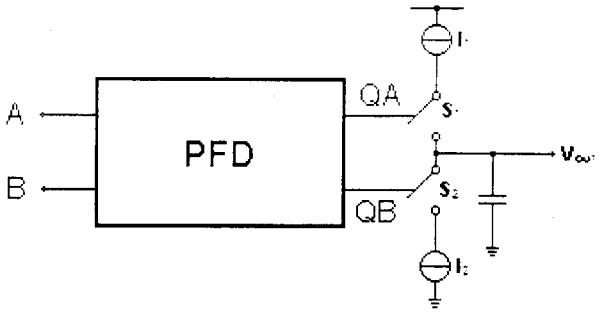


Abbildung 7 Blockschaltbild des PFD mit Ladungspumpe und Schleifenfilter [#RAZA98]

Wenn A eine höhere Frequenz als B hat oder wenn A die gleiche Frequenz als B hat, aber gegenüber B vorauseilt, lädt die CP einen konstanten Strom I_1 durch den Schalter S_1 in den Kondensator. Währenddessen vergrößert sich die Ausgangsspannung V_{out} ständig.

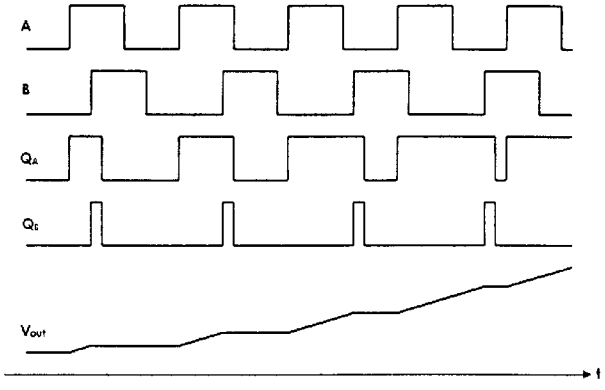


Abbildung 8 Zeitbereich-Antwort der Ladungspumpe mit Schleifenfilter [#RAZA98]

Umgekehrt, wenn A eine kleinere Frequenz als B hat oder gegenüber B hinterherhinkt, wird V_{out} ständig

abnehmen. Q_A steuert dabei den Schalter S_1 und Q_B den Schalter S_2, wobei log. ‚L' den Schalter geschlossen und log. ‚0' den Schalter geöffnet hält. Wie vorher gezeigt, ist es jedoch unvermeidbar, dass Q_A und Q_B beide für eine kurze Zeit log. ‚L' sind, was eine Fehlfunktion des PFD darstellt. In diesem Fall sind dann sowohl S1 als auch S2 geschlossen. Sind die beiden Stromquellen I_1 und I_2 jedoch so dimensioniert, dass der Wert ihrer Stromstärken gleich ist, wird der Strom, der von I_1 geliefert wird von I_2 komplett aufgenommen. Dadurch wird der Fehler des PFD korrigiert und bei $Q_A = Q_B$ = ‚L' fließt kein Strom durch den Kondensator und V_{out} bleibt unverändert (konstant), als ob S_1 und S_2 geöffnet wären.

Die Implementierung (siehe Abbildung 9) besteht aus zwei identisch aufgebauten Teilen für jeweils das up- und down-Signal.

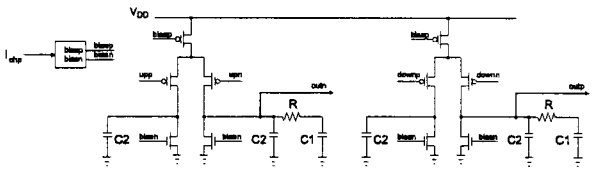


Abbildung 9 in dieser Arbeit verwendete Ladungspumpe mit Schleifenfilter [#GUST04]

Das Schleifenfilter wird von C_1, C_2 und R gebildet. Der VCO wird von out_n und out_p gesteuert.

3.3. VCO

Den verwendeten VCO zeigt Abbildung 10. Er besitzt einen differentiellen Eingang (V_+ und V_-) mit einem breiten Abstimmbereich.

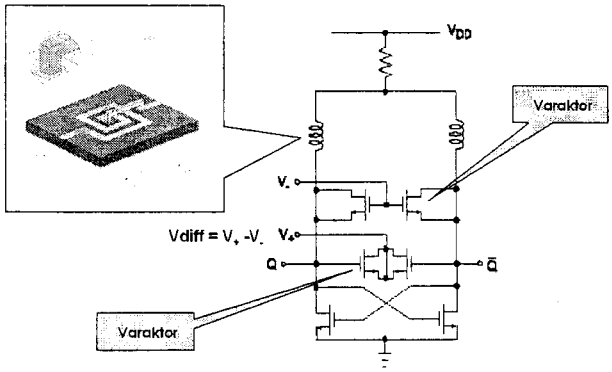


Abbildung 10 Schaltkreis des VCO [#GUST04]

Ring-Oszillatoren werden üblicherweise in RF-Systemen nicht verwendet, da sie die hohen Anforderungen an Phasenrauschen, Seitenbänder und Einrastdauer einer WLAN-Komponente nicht erfüllen. Stattdessen werden LC- oder auch LCR-Oszillatoren verwendet. Die Frequenz f_{out}, mit der ein LC-Oszillator schwingt, ist nur von den Werten der Spulen(n) und (dem) Kondensator(en) abhängig. Der Wert monolithischer Spulen auf dem Chip lässt sich jedoch nicht verändern, deshalb können letztendlich nur Kapazitätswerte verändert werden, um den VCO einzustellen. Es werden deshalb Kondensatoren benötigt, deren Kapazitätswerte spannungsabhängig sind. Diese werden Varaktoren genannt. Ein MOSFET, bei dem Drain, Source und Bulk (D, S, B) miteinander verbunden sind, bildet eine Kondensatorstruktur in CMOS-Technologie mit einem Kapazitätswert, der von der Spannung V abhängt, die zwischen D S B und Gate (G) anliegt Die Varaktoren werden aus NMOS-Transistoren gebildet. An den Ausgängen der Schaltung ist zusätzlich jeweils ein Inverter angeschlossen, um das Sinus-Signal, das der VCO erzeugt in ein annäherndes Rechteck-Signal umzuwandeln. Um die nachfolgende Stufe, den Teiler M zu treiben, ist ein Signal mit relativ steilen Flanken nötig, um zwischen den Pegeln zu unterscheiden. Die Anstiegszeit und das Tastverhältnis des Signals, das am Ausgang des Inverters anliegt wurden gemessen (mit dem Werkzeug „Calculator" des verwendeten EDA-Systems). Das Tastverhältnis beträgt 53 % : 47 % und die Anstiegszeit circa 14 ps.

3.4. Teiler M

Die Zusammenschaltung von Dual-Modulus Prescaler und Pulse/Swallow Counter ist in Abbildung 11 zu sehen.

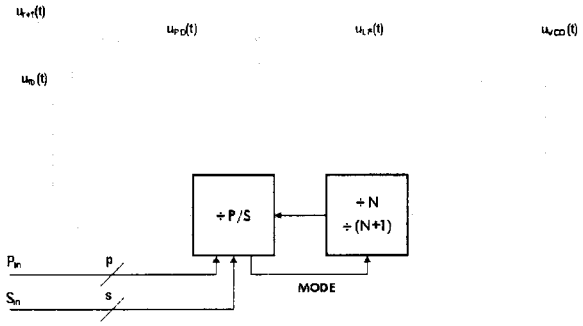


Abbildung 11 Blöcke des Teilers M [#GUST04]

Der Pulse/Swallow Counter steuert den Dual-Modulus Prescaler über das Steuersignal ‚MODE' und bestimmt somit, ob der Prescaler durch N oder durch N+1 teilen soll.

3.4.1 Dual-Modulus Prescaler

Beim Entwurf besteht ein Kompromiss zwischen Stromverbrauch und Geschwindigkeit bezüglich der Zusammenschaltung der einzelnen Flipflops. Deswegen muss zwischen zwei verschiedenen Varianten von Frequenzteilern unterschieden werden:

- Synchron arbeitende Frequenzteiler
- Asynchron arbeitende Frequenzteiler

Bei synchronen Teilern besteht keine zeitliche Verzögerung zwischen den Ausgängen der beteiligten Stufen, da das zu verarbeitende Signal (Eingangssignal) zeitgleich an den Takteingängen aller beteiligten Flipflops anliegt.

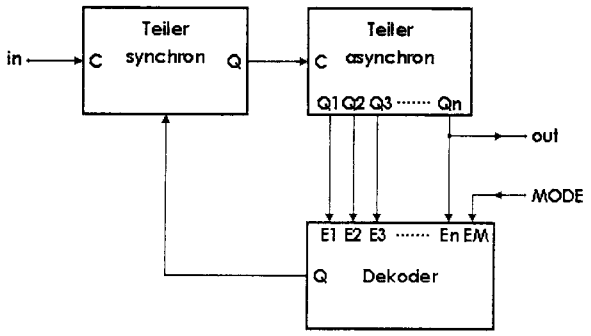


Abbildung 12 Dual-Modulus Prescaler: Allgemeines Blockschaltbild

Es tritt also nur eine Verzögerung zwischen dem Eingangssignal und den Ausgangssignalen der einzelnen Stufen auf. Diese Verzögerung wird bestimmt von der Verzögerungszeit eines einzelnen Flipflops. Jedoch müssen alle beteiligten Flipflops die maximale Eingangsfrequenz verarbeiten, was zur Folge hat, dass ein synchroner Frequenzteiler eine höhere Stromaufnahme hat wie ein asynchroner. Bei asynchronen Teilern wird jeweils der Takteingang C mit dem Ausgang Q und dem Eingang D der vorhergehenden Stufe (Flipflop) verbunden. Mit jeder Stufe wird die Frequenz halbiert, d.h. nur die erste Stufe muss die maximale Eingangsfrequenz verarbeiten und hat somit den größten Stromverbrauch. Entsprechend sinkt der Stromverbrauch mit jeder weiteren Stufe. Allerdings tritt bei asynchronen Teilern eine größere Verzögerung auf als bei synchronen, da das Eingangssignal alle Stufen nacheinander durchlaufen muss. Die Gesamt-Verzögerung setzt sich bei asynchronen Teilern also aus den Verzögerungen der einzelnen Stufen zusammen. Um die Vorteile eines synchronen Frequenzteilers mit denen eines asynchronen zu verbinden, werden beim Dual-Modulus Prescaler ein synchroner und ein asynchroner Frequenzteiler verwendet. Die Verarbeitung des ‚MODE'-Signals übernimmt ein Dekoder, der den synchronen Teiler steuert und somit entscheidet, ob durch N oder durch N+1 geteilt werden soll.

Um den Teilungsfaktor N der Schaltung zu ermitteln, sei m die Anzahl Stufen des synchronen Teilers und n die Anzahl Stufen des asynchronen Teilers. Der Teilungsfaktor des synchronen Teilers ist dann 2^m bzw. 2^m+1. Mit n Stufen kann der asynchrone Teiler 2^n verschiedene Ausgangszustände erzeugen. Der Teilungsfaktor N lässt sich wie folgt berechnen:

$$N = 2^m + (2^n - 1) * 2^m = 2^m * 2^n$$

Bei dem 32/33 Prescaler soll der Stromverbrauch möglichst gering gehalten werden.

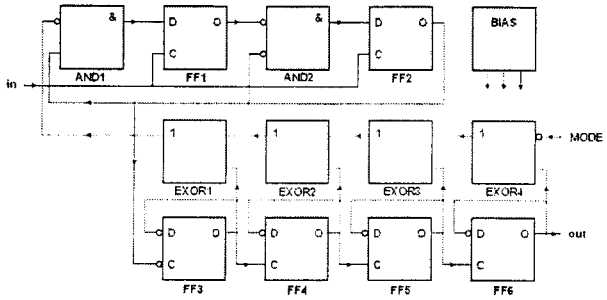


Abbildung 13 Blockdiagramm des 32/33 Prescalers [#GUST04]

Der synchrone Teiler, der Dual-Modulus Block genannt wird, besteht deshalb aus der Mindestanzahl von zwei Flipflops (FF1 und FF2). Er hat also nur eine Stufe, die von FF2 gebildet wird (FF1 wird nur verwendet, wenn der Modus N+1 gewählt wird). Der Teilungsfaktor des Dual-Modulus-Block ist also $2^1 = 2$ bzw. $2^1+1 = 3$ (m = 1). Somit ist der Dual-Modulus Block ein, mithilfe des Signals ‚MODE', umschaltbarer Teiler, der durch 2 (MODE = ‚0') oder durch 3 (MODE = ‚L') teilt und wird von den synchronen Flipflops FF1 und FF2 sowie den beiden UND-Gattern AND1 und AND2 gebildet. Da die Anzahl Stufen des synchronen Teilers im Hinblick auf geringen Stromverbrauch feststeht, bleibt als einziger Faktor, der noch ermittelt werden muss, um den Teilungsfaktor von 32 bzw. 33 zu erhalten, die Anzahl Stufen n des asynchronen Teilers. Für N = 32 erhält man n = 4 Stufen. Es ist also ein vierstufiger asynchroner Teiler notwendig. Diese Teilerkette wird von den Flipflops FF3 bis FF6 gebildet. Um den Dekoder zu realisieren, werden EXOR-Gatter verwendet. Es muss für jede Stufe des asynchronen Teilers ein EXOR-Gatter zur Verfügung stehen, bei dem hier gezeigten Prescaler sind es demnach vier Stück (EXOR1 bis EXOR4). Der Ausgang des Dekoders (Ausgang EXOR1) steuert den synchronen Teiler. Wenn dieser Ausgang log. ‚L'

(high) ist, ist der Ausgang von AND1 log. ‚0' (low) und somit der Ausgang von FF1 ebenfalls log. ‚0'. Der Ausgang von AND2 ist in diesem Falle immer dann log. ‚L', wenn der Ausgang von FF2 log. ‚0' ist. D. h. immer wenn der Ausgang des Dekoders log. ‚L' ist, bilden AND2 und FF2 einen Teiler, der durch zwei teilt. Andernfalls teilt der synchrone Teiler durch 3.

3.4.2 Pulse/Swallow Counter

Im Ausgangszustand[9] teilt der Prescaler durch N+1 (MODE ='L'), bis der Swallow Counter überläuft (Wert S ist erreicht). Dann wechselt das Signal ‚MODE' seinen logischen Pegel.

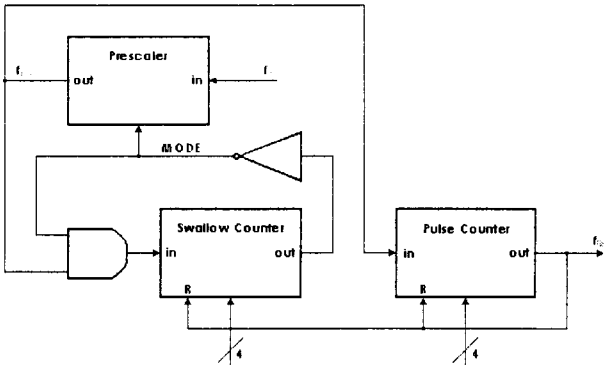


Abbildung 14 Teiler M: Gesamtblockschaltung

Der Prescaler teilt anschließend durch N, bis wiederum der Pulse Counter beim Zählerzustand P angelangt ist.

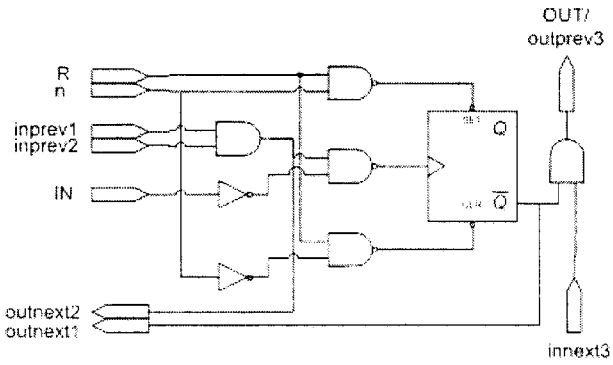


Abbildung 15 Zählerzelle des Pulse und Swallow Counters

Anschließend setzt dieser sich selbst und den Swallow Counter zurück. Der Ausgang hat einen kompletten Zyklus für $M = N*P_{in}+S_{in}$ Zyklen am Eingang erzeugt. Danach wiederholt sich der gesamte Vorgang. Der Swallow Counter und der Pulse Counter sind gleich aufgebaut. Pro Bit verwenden beide jeweils eine Zählerzelle (siehe Abbildung 15). Bei R = ‚L' kann die Zelle programmiert werden, indem das Flipflop je nach Programmiereingang n gesetzt oder rückgesetzt wird. Sobald R auf ‚0' wechselt, fängt der Zähler an zu zählen.

Die hier verwendeten Zähler sind jeweils 4 Bit breit und setzen sich dementsprechend aus jeweils vier Zählerzellen zusammen.

[9] engl.: initial reset state

4. Literatur

[GUST04] Hans Gustat, Frank Herzel and Igor Shevchenko, A Fully-Integrated Low-Power Low-Jitter Clock Synthesizer with 1.2 GHz Tuning Range in SiGe:C BiCMOS, in Proc. International SiGe Technology and Device Meeting, Frankfurt (Oder), Germany, May 2004, pp. 270-271

[RAZA98] Behzad Razavi, RF Microelectronics, Prentice Hall PTR, ISBN 0-13-887571-5

[RAZA01] Design of Analog CMOS Integrated Circuits, McGRAW-HILL International Edition 2001, Electrical Engineering Series, ISBN 0-07-118815-0 (softcover edition), ISBN 0-07-118839-8 (hardcover edition)

[RESE02] SiGe Bipolar Integrated Circuits up to 100 GHz, Research Trends, Special Edition / October 2002

Device Charakterisierung am Prozess AMIS_C05M

V. Lange, A. Friesen und G. Higelin
Fachhochschule Furtwangen, Robert Gerwig Platz 1, 78120 Furtwangen
Tel.: 07723-9202505, Fax: 07723-9201109, vl@fh-furtwangen.de

Zusammenfassung:

In der vorliegenden Arbeit wurden digitale und analoge Schaltungen zur Device-Charakterisierung eines MOS Prozesses entworfen und charakterisiert.
Im einzelnen handelt es sich um : NMOS- und PMOS-Transistoren, Ringoszillatoren, sowie Bandgap Spannungsreferenz und Photodioden.
Die integrierte Schaltung wurde gefertigt in einem 0.5 µm CMOS Prozess mit drei Metalllagen (AMIS_C05M).

1. Einführung

Transistoren sind das Grundelement aller integrierter Schaltungen. Ihre SPICE – Parameter sind die Grundlage für die Simulation der damit entworfenen komplexen Schaltungen.

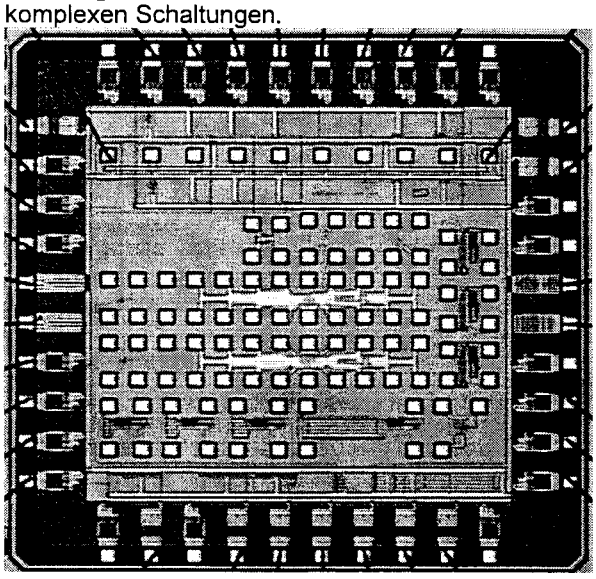


Abbildung 1: Gesamtchip

Je genauer diese spezifischen Modellparameter bekannt sind, umso genauer können komplexe Schaltungen während der Entwurfsphase und somit vor der Fertigung simuliert werden. Der Chiphersteller stellt unterschiedliche Parametersätze zur Verfügung: typische Werte sowie Grenzwertdaten. Diese Datensätze werden im folgenden als "FAB"-Parameter bezeichnet, während die im Labor, aus Messungen am Transistor gewonnenen Parametersätze, "LAB"-Parameter genannt werden. Die Auswirkung unterschiedlicher Parametersätze auf die Transistorkennlinien wird in Abschnitt 4.1 diskutiert.
Ringoszillatoren liefern auf einfache Weise Informationen über charakteristische Eigenschaften ihrer Grundbausteine, die Verzögerungszeiten der Inverter und anderer logischer Gatter. Außerdem können Rückschlüsse auf Einflüsse parasitäter Elemente, z.B. Kapazitäten gezogen werden. Diese parasitären Größen können bereits während des Entwurfs durch die Möglichkeit der „Back Annotation" bestimmt und in der Simulation berücksichtigt werden. Damit können mit diesen Strukturen neben dem Einfluß der SPICE-Parameter auch die Auswirkung parasitärer Kapazitäten und Widerstände auf die Device-Performance untersucht werden.
An weiteren CMOS Grundschaltungen wie Operationsverstärker und spannungsgesteuerten Oszillatoren (VCO) soll der Einfluß der SPICE-Parameter auf das Verhalten gängiger Schaltungen demonstriert werden.
Temperatureigenschaften einer Prozesstechnologie lassen sich an Spannungsreferenzen untersuchen. Zu diesem Zweck befindet sich auf dem Chip eine Bandgap Spannungsreferenz.
Die Lichtempfindlichkeit von pn-Übergängen ist die Grundlage für die Möglichkeit in einem CMOS-Prozess optische Sensoren zu realisieren.
Abbildung 1 zeigt die Gesamtansicht des Chips. In der Mitte sind die NMOS- und PMOS-Transistorflöten angeordnet, die über Testpads mit dem Nadelprober kontaktiert werden können. Rechts

und unterhalb von den Transistorflöten befinden sich Ringoszillatoren unterschiedlicher Länge und mit unterschiedlichen Grundzellen, die ebenfalls über Testpads erreichbar sind. Für diese Zellen wurden eigene Treiber entworfen, die den Mietec-Bondpad-Treiber „bu8ma" ersetzen. Ganz oben und ganz unten befinden sich dann alle Schaltungen, die über die Bondpad-Treiber von Mietec im gehäusten Chip kontaktierbar sind. Auf diese Weise können die Strukturen über Bondpads im gehäusten Chip und/oder Testpads mit dem Waferprober kontaktiert werden.

2. Arbeitsumgebung

2.1 Technologie

Der Chip wurde im Standard CMOS Prozess AMIS 0.5µm gefertigt. Schaltungen in dieser Technologie sind mit einer Versorgungsspannung von V_{dd}=3.3V zu betreiben. Die Oxiddicke beträgt t_{ox}=10nm und die minimale Kanallänge 0.5µm. Die wichtigsten Eigenschaften der NMOS- und PMOS-Transistoren sind der Tabelle 1 zu entnehmen.

	NMOS	*PMOS*
Min. Kanallänge [µm]	0.5	0.5
Min. Kanalweite [µm]	0.8	0.8
VT0 [V]	0.6	-0.59
KP (Transconductance) [µA/V²]	134	34
GAMMA (Body Factor) [V^1/2]	0.65	0.7

Tabelle 1: Transistor-Eigenschaften des AMIS 0.5µm CMOS Prozesses

Weitere Informationen zu diesem Prozess befinden sich auf der Europractice Homepage [1].
Die Fertigung erfolgte im Januar 2004 bei Europractice im Run 944

2.2 Chip - Entwurf

Der Entwurf des Chips wurde im Labor für IC-Entwurf der FHF unter Mentor Graphics Version C2 durchgeführt. Der Entwurf des Full Custom Chip wurde in den folgenden Schritten durchgeführt (Abbildung 2):

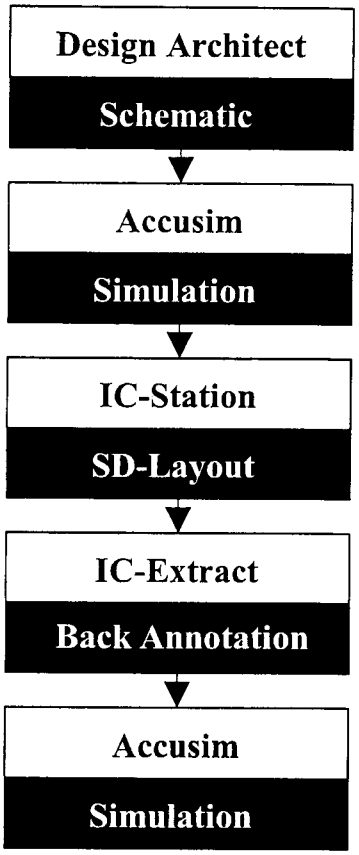


Abbildung 2: Design Flow

Mit dem Mentor Graphics Werkzeug Design Architect wurde zunächst für jede Teilschaltung der Schaltplan (Schematic) erstellt, der im Anschluß mit Accusim und den typischen „FAB"-Parametern simuliert wurde. Ausgehend vom Schematic wurde mit der IC-Station das Layout erstellt (Schematic Driven Layout, SDL). IC-Extract wurde danach zur Verifizierung des Layout und zur Bestimmung parasitärer Größen (Kapazitäten und Widerstände) eingesetzt mit anschließender nochmaliger Simulation unter Accusim.

2.3 Mess – System

Messungen und insbesondere die Bestimmung der SPICE-Parameter der Transistoren erfolgten im Labor für Halbleitercharakterisierung der FHF. Die Messung der DC-Transistorkennlinien erfolgte auf Waferlevel mit einem Süss MP4 Nadelprober (Abbildung 3).

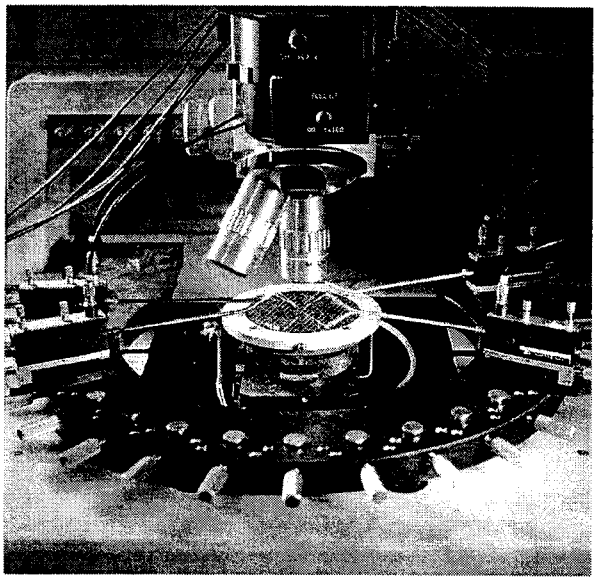


Abbildung 3: Süss-MP4 Nadelpober

Die Steuerung der Messgeräte (HP4142 Source Monitor System) erfolgt vom PC gesteuert über den GPIB-Bus unter dem „Integrated Circuit Characterization and Analysis Program" IC-CAP (Version 2004) [2] von Agilent Technologies. IC-CAP stellt Instrumententreiber zur Verfügung, um eine Vielzahl von Messgeräten zur Messdatenerfassung einzusetzen. Darüber hinaus ist es möglich über selbstgeschriebene Macro's mit Messgeräten über den GPIB-Bus zu kommunizieren. Für alle gängigen Halbleiterbauelemente stehen Programmpakete bereit für die Bestimmung der SPICE-Parameter. Zugleich ist die Simulation der Kennlinien zum visuellen Vergleich mit den Messungen möglich.
Die SMU's HP4142 können gleichzeitig als Quelle und Messgerät verwendet werden. Die Auflösung als Strommessgerät beträgt 20fA. Damit können mit hoher Genauigkeit auch Subthreshold-Ströme von MOS-Transistoren gemessen werden. Des Weiteren stehen eine HP4284 LCR-Messbrücke zur Verfügung mit der u.a. Kapazitäten im Bereich von 0.01fF bis 10F mit einer Messunsicherheit von <0.5% im Frequenzbereich von 20Hz bis 1MHz gemessen werden können. Eine Temptronic 315A Temperatursteuerung ermöglicht die Messung von Temperatureffekten im Temperaturbereich von Raumtemperatur bis zu 200°C über einen heizbaren Chuck auf Waferlevel oder am gehäusten Chip.

3. Strukturen

3.1 Transistorflöten

Die Bestimmung der Transistor DC SPICE Parameter erfolgt aus der Messung der DC-Kennlinien an Transistoren unterschiedlicher Geometrie. Ziel war die Bestimmung der BSIM3 (Berkeley Shortchannel IGFET Model [3]) DC-Parameter. Dieses Modell ist geeignet für Transistoren mit einer minimalen Kanallänge von ca. 0.15µm, benötigt jedoch wegen der großen Zahl von Modellparametern (>130) eine große Zahl Transistoren unterschiedlicher Geometrie (Abbildung 4). Neben den sogenannten Corner-Devices "small", "short", "narrow" und "large" werden Transistoren zur Längen- und Weitenskalierung benötigt. Von Interesse ist auch, ob die für eine ca. 5µm-Technologie ausgelegten UCB MOSFET Model Level 2 und Level 3 [4] für die Beschreibung dieser Transistoren geeignet sind. Diese Modelle benötigen nur die "short"- , "narrow"- und "large"- Transistoren zur Parameterbestimmung, wodurch sich der Meß- und Analyseaufwand drastisch vermindert.

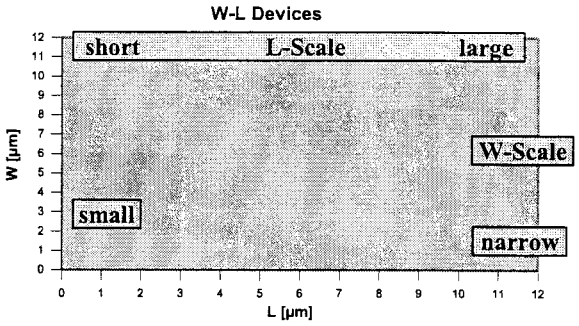


Abbildung 4: BSIM3 Transistorgeometrien

Auf dem Chip befinden sich zwei Transistorflöten (NMOS, PMOS) mit jeweils 14 Transistoren.
Abbildung 5 zeigt eine NMOS-Flöte mit den Transistorgeometrien W[µm]/L[µm] (von links nach rechts): 0.8/0.5 (small) 0.8/1.2 0.8/3 0.8/5 0.8/10 (narrow) 1.2/10 3/10 5/10 10/10 (large) 10/3 10/1.5 10/1.2 10/0.8 10/0.5 (short).
Die Transistoren einer solchen Flöte haben gemeinsame Gate-, Bulk- und Source- Anschlüsse, die einzelnen Transistoren werden durch den individuellen Drain- Anschluß selektiert.

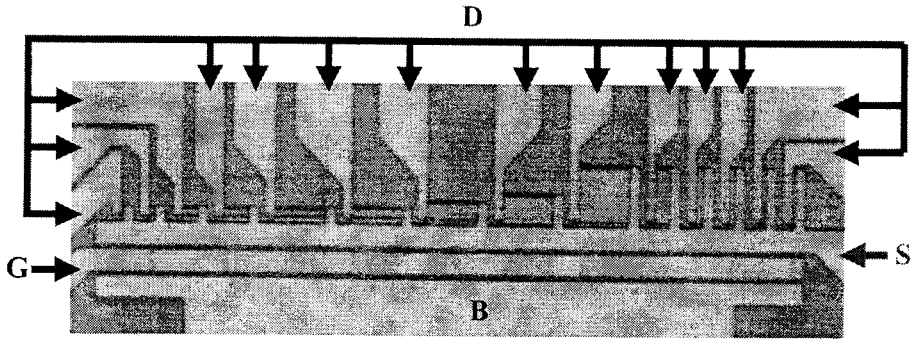


Abbildung 5: NMOS-Transistorflöte

3.2 Ringoszillatoren

Ringoszillatoren bestehen aus einer geraden Anzahl von Invertern und einem NAND (Abbildung 6).

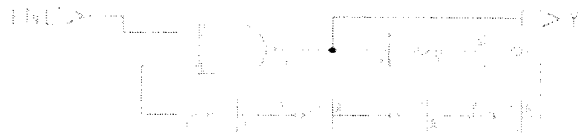


Abbildung 6: Ringoszillator

Ein Eingang des NAND wird zum Aktivieren des Oszillators benötigt, an den zweiten Eingang wird der Ausgang des letzten Inverters der Inverterkette angeschlossen, wodurch der Ringoszillator geschlossen wird. Aus der gemessenen Oszillatorfrequenz f_{osc} und der Anzahl der Inverter n kann die Verzögerungszeit der Inverter bestimmt werden:

$$f_{osc} = \frac{1}{n(t_{PHL} + t_{PLH})}$$

Dabei sind die Zeiten t_{PHL} und t_{PLH} die Verzögerungen beim Ein- und Ausschalten des Inverters:

$$t_{PHL} = R_P * C_{OUT} \quad \text{bzw.} \quad t_{PLH} = R_N * C_{OUT}$$

Auf diese Weise kann sehr einfach der Einfluss parasitäter Größen (Widerstand und Kapazität) bestimmt werden.
Es wurden Ringoszillatoren unterschiedlicher Länge (7 bzw. 67 invertierende Elemente) realisiert. Zusätzlich wurden Oszillatoren mit langen Verbindungsleitungen zwischen den Invertern entworfen zur Untersuchung des Einflusses parasitärer Leitungseffekte.
Wegen der hohen Oszillationsfrequenz des kurzen Oszillators wurde ein 4-stufiges JK-Flip-Flop entworfen, das die Frequenz mit dem Faktor 16 teilt. Das NAND und die Inverter der Ringoszillatoren wurden selbst entworfen. Dabei wurden bei den Invertern Transistoren minimaler Kanallänge verwendet mit symmetrischem Schaltverhalten.

3.3 Grundschaltungen

Von den auf dem Chip vorhandenen Analogschaltungen (Bandgap, VCO und Operationsverstärker) soll an dieser Stelle die Bandgap Spannungsreferenz vorgestellt werden, da für die anderen Schaltungen gegenwärtig noch keine Messungen vorliegen.
Bei der Bandgap handelt es sich um eine temperaturstabilisierte Referenzspannungsquelle. Das Prinzip beruht darauf, daß Dioden und Widerstände einen Temperaturkoeffizienten mit unterschiedlichem Vorzeichen haben. Das Schematic der auf dem Chip befindlichen Bandgap ist in Abbildung 7 zu sehen.
In der Schaltung werden die Dioden durch PMOS Transistoren realisiert, die als Dioden geschaltet sind. Diese Elemente haben einen negativen Temperaturkoeffizient. Die Widerstände mit positivem Temperaturkoeffizient werden auf dem Chip durch "High Ohmic Poly" realisiert.

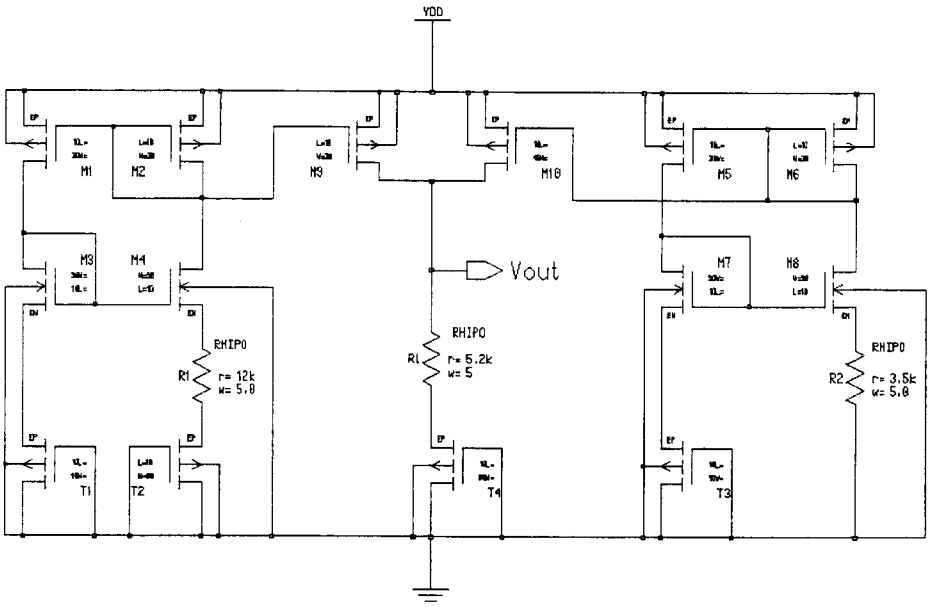


Abbildung 7: Schaltplan der Bandgap Spannungsreferenz

Der in den linken und rechten Zweigen der Schaltung erzeugte möglichst temperaturkonstante Summenstrom wird wieder über einen Widerstand und eine Diode geleitet, die sich in der Temperatur kompensieren, und erzeugen auf diese Weise eine konstante Spannung (Simulation: 1.48V).

3.4 Optisches CMOS Sensorelement

In der verwendeten Technologie lassen sich Dioden zum p-Substrat auf zwei unterschiedliche Weisen realisieren. Im einen Fall wird für das n-Gebiet die n^+-Diffusion, im anderen Fall die n-Wanne des Prozesses verwendet. Abbildung 8 zeigt ein solches Element (oben: n^+-Diffusion, unten: n-Wanne).

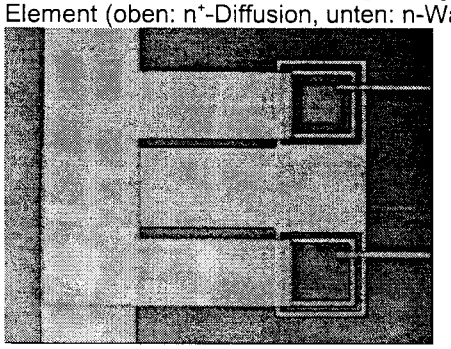


Abbildung 8: Optische Sensorelemente

Das Prinzip dieser optischen Sensorelemente ist die Ladungsträgergeneration in der Raumladungszone durch Licht. Das optische Verhalten wird beschrieben durch die spektrale Empfindlichkeit $\mathcal{R}$:

$$\mathcal{R} = \eta\,\lambda[\mu m]/1.24 \quad [A/W]$$

mit dem Wirkungsgrad η :

$$\eta = \eta(NA,ND,VDiode,r(\lambda),\alpha(\lambda))$$

Damit ist die spektrale Empfindlichkeit proportional zur Wellenlänge, zudem besteht eine indirekte Wellenlängenabhängigkeit über die optischen Materialeigenschaften (Reflexionskoeffizient r und Absorptionskoeffizient α). Prozesseigenschaften beeinflussen die spektrale Empfindlichkeit über die Dotierungen der n- und p-Gebiete. Zusätzlich beeinflusst die an der Diode anliegende Spannung die Weite der Raumladungszone und dadurch die spektrale Empfindlichkeit [5].
Neben der optischen Charakterisierung kann an den Dioden die elektrische Charakterisierung (Diodenkennlinie) und die Kapazitäts-Charakterisierung des np-Übergangs durchgeführt werden.

4. Ergebnisse

4.1 Transistoren

Die Bestimmung der BSIM3-DC-Modellparameter erfolgte unter ICCAP mit dem von der Firma AdMOS erstellten ICCAP Setup, unter dem der vollständige Ablauf der Messung, Parameterbestimmung und Simulation für die komplette Transistorflöte implementiert ist [6]:

1) Transistoren mit großer Geometrie werden mit den aus den Messungen extrahierten SPICE-Parametern sehr gut beschrieben (Abbildung 9).

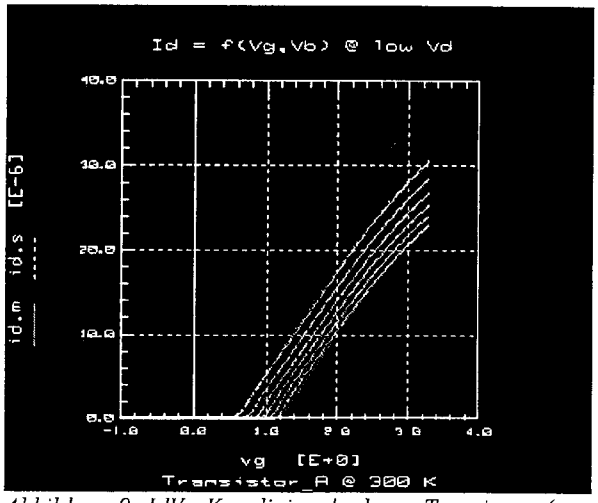


Abbildung 9: IdVg-Kennlinien des large-Transistors (rot: Messung, gelb: Simulation)

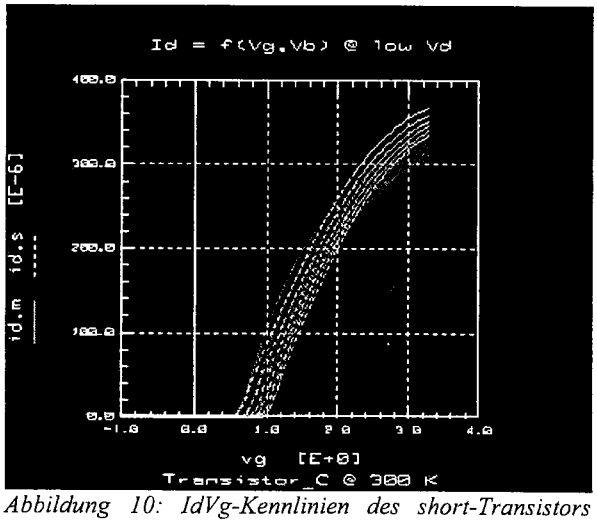


Abbildung 10: IdVg-Kennlinien des short-Transistors (rot: Messung, gelb: Simulation)

2) Transistoren mit minimaler Länge und / oder Weite zeigen geringe Abweichungen zwischen Messung und Simulation (Abbildung 10).

3) Die in dieser Arbeit bestimmten SPICE-Parameter (LAB-Parameter) zeigen generell eine bessere Übereinstimmung von simulierten und gemessenen Kennlinien wie die mit den FAB-Parametern erzeugten Kennlinien (Abbildung 11).

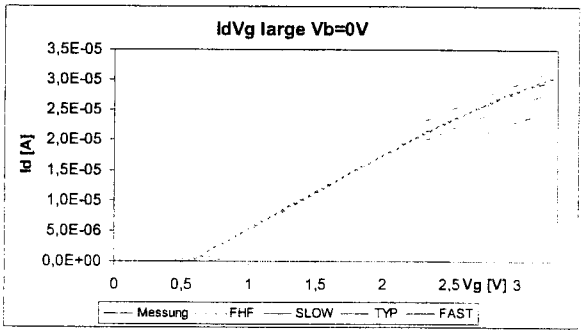


Abbildung 11: IdVg-Kennlinien des large-Transistors; Messung: rot, Simulation: FHF-grün, FAB-Parameter blau (von unten nach oben: slow, typ, fast)

4) Die LAB-Parameter liegen meistens in dem von den FAB-Parametern (slow-typical-fast) vorgegebenen Bereich.

5) Die Transistoren der 0,5µm-Technologie lassen sich auch durch die UCB Modelle Level2 und LEVEL3 erfolgreich beschreiben (Abbildung 12).

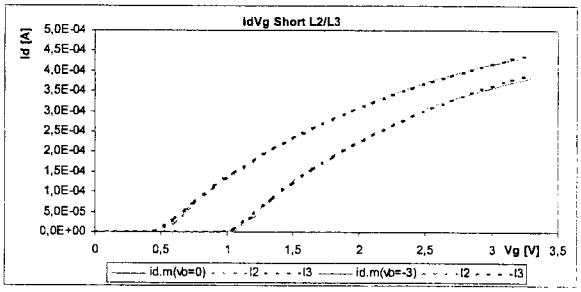


Abbildung 12: IdVg-Kennlinien des short-Transistors für Vb=0V bzw. -3V. Messung: rot; Simulation: UCB-Modell Level2 (grün) und Level3 (blau)

4.2 Ringoszillatoren

Neben den Leitungskapazitäten und Leitungswiderständen zwischen den Elementen des Ringoszillators sind die parasitären Transistorkapazitäten von Bedeutung. In Tabelle 2 sind die BSIM3 FAB-Parameter für die Gate-Source und Gate-Bulk Kapazitäten angegeben. Die Abweichungen liegen bei bis zu 10%:

	SLOW	*TYP*	*FAST*
CGSO	126pF	138pF	142pF
CGBO	326pF	345pF	367pF

Tabelle 2: FAB-Parameter von Transistorkapazitäten

Die Auswirkung dieser Parametervariation auf die Oszillatorfrequenz eines aus 67 Elementen bestehenden Ringoszillators wurde mit Accusim am Schematic untersucht. Die Analyse ergibt Abweichungen von über 20% gegenüber dem mit den typischen Parametern simulierten Oszillator (Tabelle 3). Die Simulation mit den aus Transistormessungen ermittelten BSIM3–DC-Parametern (Abschnitt 4.1) und den typischen FAB-Parametern für die Kapazitäten ergibt eine Frequenz von 143,5MHz.
Mit IC-Extract wurde die Back Annotation des Layout des Ringoszillators durchgeführt und somit parasitäre Leitungskapazitäten bestimmt. Die für diese Situation ermittelte Oszillatorfrequenz liegt bei 96,8MHz und zeigt deutlich den Einfluß der parasitären Elemente und damit die Notwendigkeit der Back Annotation.
Messungen an diesem Oszillator ergeben einen Wert von 74MHz. Dieser nochmals gravierende Unterschied zeigt, daß für eine exakte Analyse auch aus Messungen gewonnene Transistorkapazitäten notwendig sind und die Simulation nach der Back Annotation mit den anderen Parametersätzen noch durchgeführt werden muß.

	SLOW	*TYP*	*FAST*
f [MHz]	99.8	123	156
	DC:FHF C:TYP	*Back Annot. TYP*	*Messung*
f [MHz]	143,5	96,8	74

Tabelle3: Oszillatorfrequenzen

4.3 Spannungsreferenz

Die Accusim Simulation der Bandgap ergibt einen Wert für die Ausgangsspannung von 1,48 V mit einer Genauigkeit von 4mV im Temperaturbereich von 0 bis 100 °C. Messungen an der Bandgap Schaltung zeigen deutliche Abweichungen von diesem Wert (Abbildung 13).

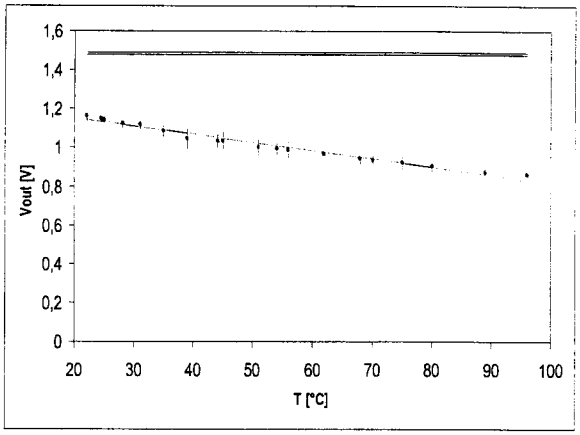


Abbildung 13: Ausgangsspannung als Funktion der Temperatur der Spannungsreferenz

Weder der Absolutwert noch die Temperaturkonstanz stimmen mit den Simulationsergebnissen (blaue Linie) überein.
Eine Regressionsanalyse der gemessenen Daten ergibt einen Steigungskoeffizienten von -4mV/°C.
Referenzspannungswert und Temperaturkonstanz dieser Schaltung werden wesentlich beeinflußt durch die Parameter des verwendeten ohmschen Widerstandes, High Ohmic Poly (HIPO) (Tabelle 4).

HIPO	*MIN*	*TYP*	*MAX*
Rsh [Ω/sq]	900	1000	1100
T_{CL} [ppm/°C]	-1500	-1250	-1000
T_{CQ} [ppm/°C²]	2	3	5

Tabelle 4: FAB-Parameter des High Ohmic Poly

Die Abweichungen gegenüber dem typischen Wert betragen 10% beim Widerstand Rsh und 20% beim linearen Temperaturkoeffizient T_{CL}. Simulationen mit diesen unterschiedlichen Parametern sollen zeigen, ob damit die Meßwerte reproduziert werden können.

4.4 Optische Sensoren

Die Messungen des spektralen Verhaltens der Dioden wurden auf Waferlevel durchgeführt. Die Größe der Dioden beträgt 10µm*10µm. Die Einkopplung des Laserlichtes erfolgte über die Optik des Beobachtungsmikroskops, wobei das Mikroskopobjektiv zur Fokusierung des Lichtes auf den Chip verwendet werden konnte. Zur Bestimmung der spektralen Empfindlichkeit muß die auf das lichtempfingliche Element einfallende Lichtleistung bekannt sein. Die Leistungsmessung konnte mit ausreichender Genauigkeit durchgeführt werden und lag typisch bei 10-100µW. Problematisch ist die Bestimmung der Gesamtfläche auf dem Chip, auf die diese Leistung auftrifft. Erst bei möglichst genauer Kenntnis dieser Fläche kann die Leistung auf der Diode bestimmt werden. Dieser Wert beträgt bei den jetzt durchgeführten Messungen nur einige nW.
Messungen zeigen das zu erwartende Verhalten: Mit zunehmender Lichtleistung steigt der Kurzschlußstrom linear (Abbildung 14). Aus der Steigung kann die Empfindlichkeit bestimmt werden (8,3A/W).

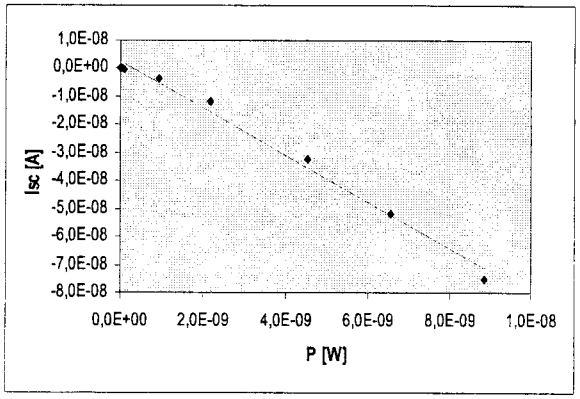


Abbildung 14: Kurzschlussstrom der np-Diode als Funktion der Lichtleistung (λ=488nm)

Dieser ungewöhnlich hohe Wert ist auf Fehler in der Flächenbestimmung zurückzuführen. Hier ist eine Verbesserung der experimentellen Bedingungen notwendig.
Die Open Loop Spannung steigt mit zunehmender Leistung nichtlinear auf einen Wert von 0,51V.
Neben der oben diskutierten Problematik bei der Flächenbestimmung wird die Zielsetzung für die Zukunft sein, die eingestrahlte Lichtleistung durch eine verbesserte Lichteinkopplung zu erhöhen.

5. Zusammenfassung

In der vorliegenden Arbeit wurde die Device Charakterisierung am Prozess AMIS_C05M erfolgreich durchgeführt. In einem ersten Schritt wurde insbesondere gezeigt, daß
die aus Messungen extrahierten BSIM3-Parameter (LAB-Parameter) eine wesentlich bessere Übereinstimmung von Simulation und Experiment ergeben als die FAB-Parameter;
die 0,5µm-Technologie auch mit dem UCB Level2 und Level3 gut beschrieben werden können;
für die korrekte Beschreibung von Ringoszillatoren neben den Leitungsparasiten, die über die Back Annotation ermittelt werden können, auch die parasitären Transistorkapazitäten genau bekannt sein müssen;
die entworfenen analogen Grundschaltungen sind funktionsfähig:
die Dioden dieses Standardprozesses als photoempfindliche Elemente verwendet werden können.

6. Literatur

[1] http://www.europractice.imec.be/europractice/on-line-docs/prototyping/ti/ti_mtc0u5.html

[2] Agilent 85190A IC-CAP 2004 User's Guide (2004)

[3] P.K.Ko and C.Hu „BSIM3 Version 3.0 Manual" University of California, Berkeley, CA94720 (1995)

[4] A.Vladmirescu and S.Liu „The Simulation of MOS Integrated Circuits Using SPICE2" UCB/ERL M80/7 University of California at Berkeley, CA94720 (1980)

[5] S.M.Sze „Physics of Semiconductor Devices" Chapter 13 John Wiley & Sons (1981)

[6] Agilent IC-Cap 2004 Nonlinear Device Models Vol.1 Chapter 3 (2004); http://www.admos.de

Danksagung:

Die Autoren danken Herrn Achim Bumüller für die umfangreiche Unterstützung beim Entwurf des Chips und dem Land Baden Württemberg für die Projektfinanzierung im Rahmen des MPC-Verbundes.

Integration, Implementation und Verifikation eines SOC zur Sensordatenerfassung im Rahmen des Projektes WEARLOG

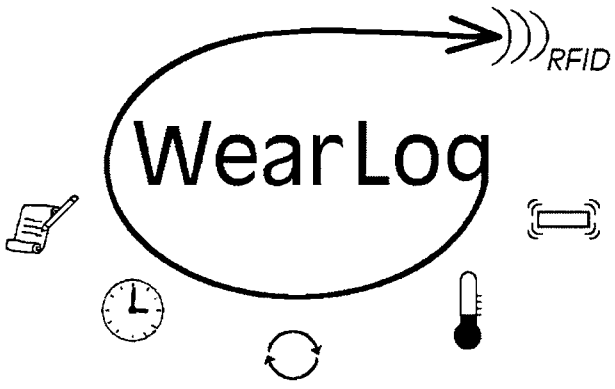
Stefan Mescheder, Prof. Dr.-Ing. Dirk Jansen, ASIC Design Center
Hochschule Offenburg, Badstrasse 24, 77652 Offenburg
Tel.: 0781/205-274, Fax: 0781/205-174, E-Mail: mescheder@planet-interkom.de

Abstract

This paper gives an overview about a project called "WearLog". This is a wireless system-on-chip (SOC) solution for measurement and information purposes.

The different components and aspects of implementation to create such a complex system are shown. The overall chip area of the system is estimated to be 10 mm² for a 0.35 µm technology.

There are high requirements for verification in order to achieve a first-time silicon design. In this project this is realized by a testing board with analog components and a FPGA with the digital part for whole system emulation and confirmation of the measurement process.



1. Einleitung

1.1. Das Projekt WearLog

Durch den integrierten Schaltkreis WearLog (Wear = Verschleiß und Log steht für Logger/Aufzeichner) soll es möglich sein, lebensdauerbestimmende Parameter von Produkten innerhalb der Gewährleistung zu überwachen und aufzuzeichnen. Das System wird über eine Batterie versorgt und verbraucht wegen seines hohen Integrationsgrades kaum Platz und Energie. Es werden durch eine Zeit- und Ereignissteuerung Messwerte der Sensoren aufgenommen, vom integrierten Mikrocontroller (FHOP - First Homemade Operational Processor) verarbeitet und in einem Flash-Speicher abgelegt. Diese Daten können über eine genormte RFID-Schnittstelle (Radio Frequency Identification, ISO 14443) ausgelesen werden.

1.2. Integration

Es handelt sich hierbei um einen hoch integrierten ASIC (Application Specific Integrated Circuit), bei dem sowohl sämtliche digitalen Schaltungsteile, wie auch die analogen Schaltungsteile zu einem „Mixed Signal"- Schaltkreis zusammengefasst wurden. Einzig der Flash-Speicher wurde aus Kosten- und Flexibilitätsgründen ausgelagert. Es können jedoch über das „Serial Peripheral Interface" (SPI) seriell ansprechbare Speicherbausteine verschiedener Hersteller und bis zu Größen mehrerer Mega Byte angeschlossen werden.

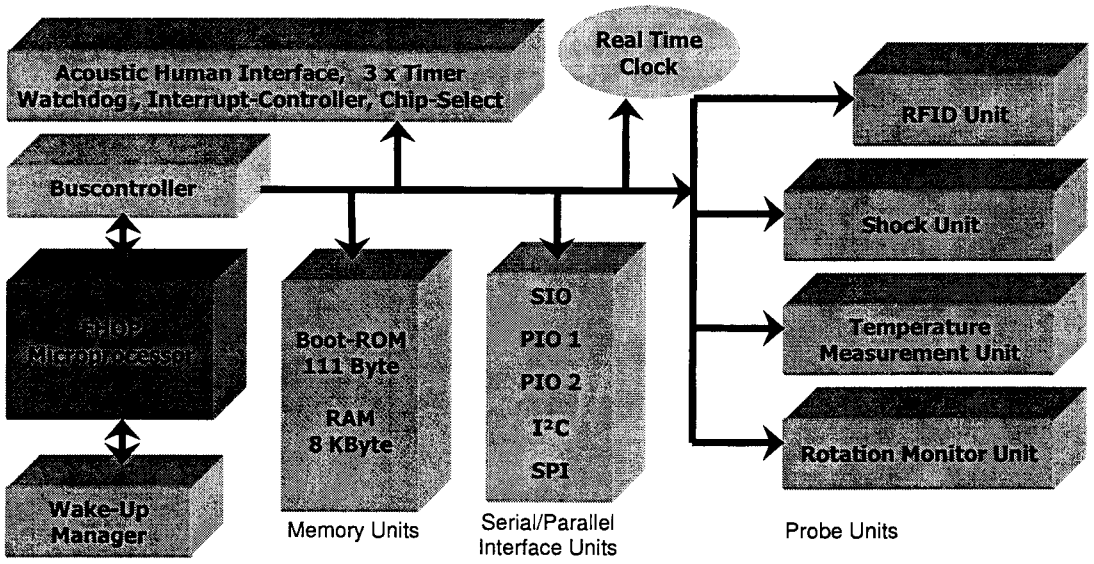


Abbildung 1 : Systemübersicht WearLog (Abkürzungen siehe Text.)

2. Aufbau des Systems

2.1. Digitale Schaltungsteile

Das Herzstück des Systems ist ein an der Hochschule Offenburg entwickelter 16 Bit Mikroprozessor. Das Zugreifen auf die Peripheriekomponenten geschieht über einen Buscontroller, der zusammen mit einer „Chip-Select" Einheit den Zugriff auf Adress- und Datenbus mit Tristate-Buffern regelt. Über diesen ist auch ein Zugriff auf die Speichereinheiten geregelt.

Als interner Speicher steht zum einen ein Boot-ROM zur Verfügung, das den Code zum Laden des Bios aus dem externen Flash in den internen 8 kByte RAM enthält. Dieser Speicherbereich wird beim Systemstart ausgeführt und ermöglicht es so, austauschbare Biosprogramme einzusetzen.

Zur drahtgebundenen Kommunikation steht eine serielle Schnittstelle (SIO), zwei parallele Ports (PIO1/PIO2) zur Ein- und Ausgabe binärer Zustände so wie eine I²C-Schnittstelle (Inter Integrated Circuit) zur Verfügung. Das „Serial Peripheral Interface" (SPI) wird zur Anbindung des externen Flash-Bausteins verwendet.

Über das „Acoustic Human Interface" können Signaltöne auf einem kleinen Lautsprecher ausgegeben werden. Diese Einheit ist insbesondere bei der Systementwicklung von großer Bedeutung, um Systemereignisse kenntlich zu machen.

Speziell für das WearLog Projekt ist ein „Wake-Up Manager" entworfen worden, der dafür sorgt, dass das System bei definierten Ereignissen seine Arbeit aufnimmt und danach wieder in einen Ruhemodus geschaltet wird. Diese Komponente trägt somit maßgeblich dazu bei, den Energieverbrauch zu minimieren.

2.2. Schaltungsteile mit analogen Komponenten

Ein integrierter Oszillator erzeugt eine Frequenz von 32 kHz. Wahlweise kann auch ein externer Frequenzgenerator verwendet werden. Diese Frequenz wird für die Echtzeituhr und die Komponenten genutzt, die im stand-by Betrieb des Gesamtsystems auf Ereignisse reagieren müssen. Nach dem Aufwecken des Systems steht eine durch eine PLL (Phase Locked Loop) generierte Frequenz von ca. 11 MHz zur Verfügung, die für genügend Leistungsreserven im Prozessorsystem sorgt.

Ein analoges Frontend der „RFID Unit" sorgt nach Anschluss einer auf 13,56 MHz abgeglichenen Spule für die drahtlose Kommunikation mit einem Lesegerät nach dem Proximity-Card Standart.

Das Aufnehmen von Vibrationen und Schockereignissen ist durch die „Shock Unit" über einen Piezosensor möglich. Auch kann das System durch Schockereignisse aufgeweckt werden.

Über die „Temperature Measurement Unit" ist eine autonome Temperaturmessung direkt auf dem Chip realisiert.

Über eine Spule und relativ bewegten Magneten, ermöglicht die „Rotation Monitor Unit" eine Messung von Rotationen. Dadurch lassen sich auch Betriebsstunden ermitteln. Die Einheit verfügt über zwei Spuleneingänge, über welche Magnetfeldschwankungen aufgenommen werden können.

3. Problematiken des Entwurfs und der Gesamtvalidierung

Sowohl das Entwickeln wie auch das Fertigen von komplexen ASIC's, ist mit hohen Kosten verbunden. Da das ASIC Design Center wie auch die meisten anderen Hochschuleinrichtungen und eine steigende Anzahl von Unternehmen nicht über die Möglichkeit verfügt, selbst integrierte Schaltkreise herzustellen, können keine Tests an Prototypen erfolgen. Das Design muss vollständig über Softwaretools und andere Testmittel verifiziert werden.

Um heutzutage in einem zeitlich vertretbaren Rahmen ein attraktives System-On-Chip zu entwickeln, ist es außerdem unabdingbar auf bereits entwickelte und bewährte Schaltungsteile zurückzugreifen. Beim Mikroprozessorkern und einem Teil der Peripherie handelt es sich um so genannte ReUse-Blöcke, die schon einmal in anderen Projekten zum Einsatz gekommen sind. Ihre Funktionen wurden in diesen ICs ausreichend verifiziert. Auch die neuen Komponenten sind einzeln getestete IP's (Intellectual Properties).

Jedoch sollen diese Blöcke zusammengesetzt ein funktionierendes Ganzes ergeben, was nicht zwangsläufig gewährleistet werden kann. Dafür müssen teilweise Änderungen und Anpassungen durchgeführt werden, die eine fortlaufende Verifikation erfordern.

Das Gesamtsystem ist auch deshalb schwierig zu testen und validieren, da es sich um eine Kombination aus analogen und digitalen Schaltungsteilen handelt, die aufgrund ihrer Komplexität auch mit leistungsfähigen Serversystemen nicht als Gesamtsystem simuliert werden können.

Das Einsetzen von „Black Boxes", „Stimuli-Files" und alternativen Verhaltensbeschreibungen birgt die Gefahr den ausgeklammerten Schaltungsteilen nicht gerecht zu werden. Eine andere Möglichkeit ist, eine Trennung von rein digitale Schaltungsteilen und analogen Komponenten durchzuführen und den digitalen Teil in einem FPGA (Field Programmable Gate Array) zu implementieren, während analoge Komponenten z.B. diskret aufgebaut werden. Diese Vorgehensweise wird später noch näher beschrieben.

Zu beachten sind bei dem Design auch die verschiedenen internen und externen Frequenzen und Domänen, die anders als in der Simulation in der Realität durch Versatz und Frequenzschwankungen zu Timingproblemen führen können.

4. Design Flow

Da es sich um ein System handelt, in dem ein Programm abgearbeitet wird, ist ein Hardware-Software Co-Design nötig. Für den Mikrocontroller (FHOP) existiert dafür ein FHOP-Desig-Kit, das verschiedene Softwaretools zur Verfügung stellt. Unter anderem auch einen Compiler, der es ermöglicht ein Assembler-Programm in den benötigten Code für den FHOP zu wandeln. Mit einem weiteren Programm kann der Code automatisch in ein VHDL-File für das Boot-ROM konvertiert werden. Das hat den Vorteil, dass das ROM technologie-unabhängig und „hard coded" ist. Wie angesprochen, enthält das Rom nur eine kurze Initialisierung und Bios-Laderoutine. Damit ist das Rom sehr klein und die Programme laufen im Ram-Speicher.

Im Gegensatz dazu sind die analogen Schaltungsteile relativ systemunabhängig und können sowohl einzeln entworfen wie auch getestet werden. Im ASIC-Entwurf werden diese Schaltungsblöcke erst beim „Place & Route" eingebunden.

Einen Überblick über den „Entwicklungsfluss" und einen Teil der verwendeten Tools gibt die folgende Abbildung.

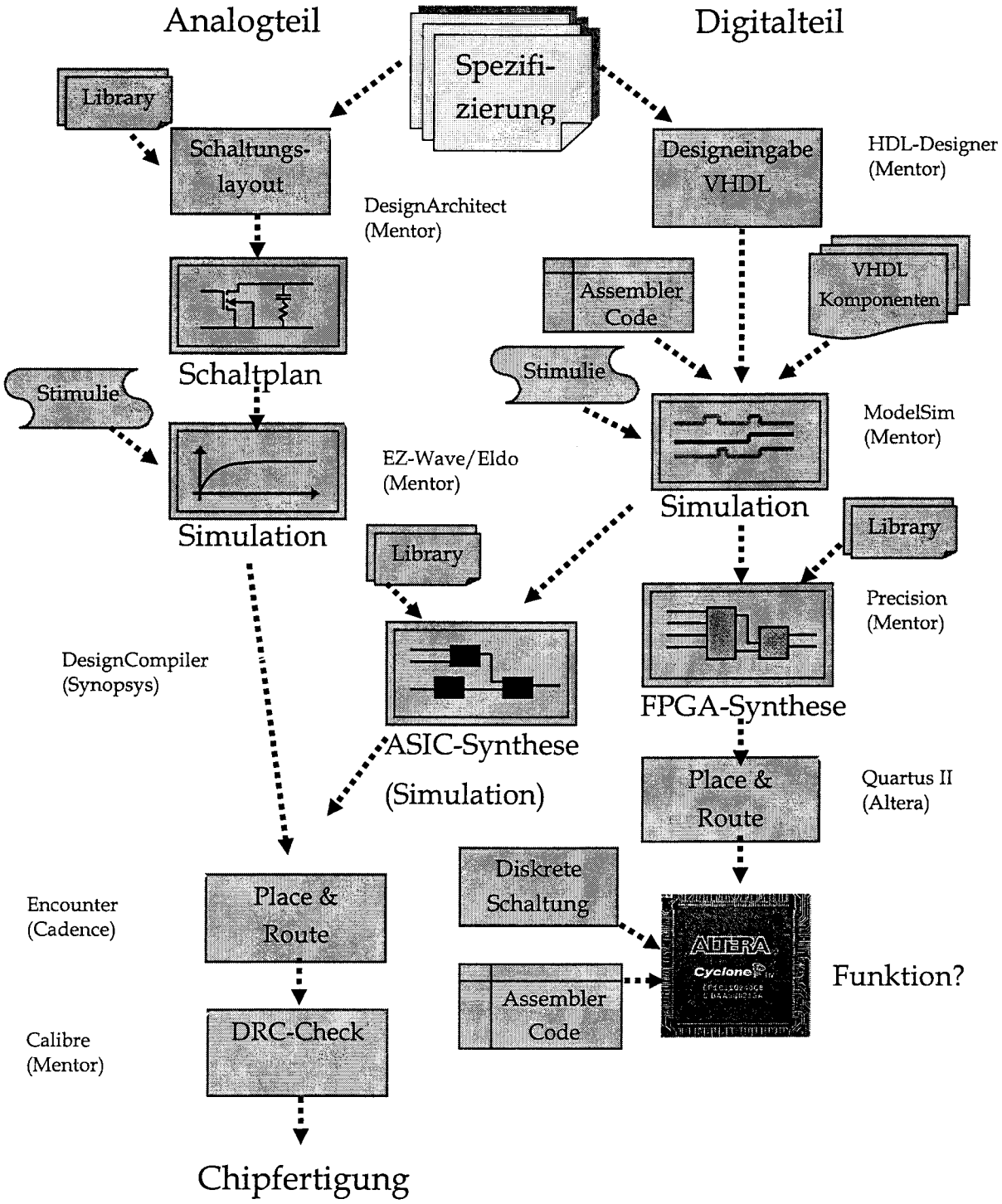
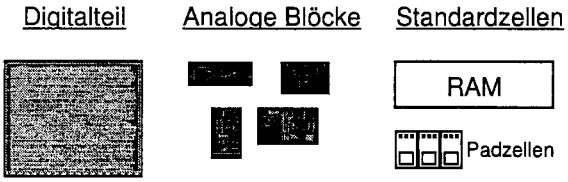


Abbildung 2 : Design Flow WearLog Projekt

5. Projekt-Clusterung

Die Schaltungsblöcke, die sich später auf dem fertigen Chip wieder finden, kann man in folgende drei Kategorien aufteilen:



Diese Schaltungsteile werden auch im Chipdesign unterschiedlich behandelt.

Bei den Standardzellen handelt es sich um Komponenten, die vom Hersteller bereits für eine bestimmte Zieltechnologie entwickelt wurden. Diese können nun sehr einfach z.B. in das ASIC Design übernommen werden. Für andere Zielplattformen wie FPGA müssen diese entsprechend ersetzt werden.

6. Projektierung

Die analogen Schaltungsteile wurden im DesignArchitect (Mentor) entworfen und verwaltet. Die digitalen Schaltungsteile sind im HDL-Designer (ebenfalls von Mentor) zusammengefasst worden. Diese Tools bieten die Möglichkeit der graphischen Darstellung von unterschiedlichen Hierarchieebenen, was die Übersicht solch komplexer Projekte gerade auf höheren Ebenen durch Abstraktion stark vereinfacht. Auch ist es so im HDL-Designer möglich, verschiedene Ansichten zu erstellen oder abzuleiten, die für eine Verifikation (z.B. auf einem FPGA) geeignet sind. Dabei greifen die Ansichten auf eine gleiche Codebasis der Schaltungsblöcke zurück und garantieren so, dass für Simulation, FPGA- und ASIC Synthese die gleichen aktuellen VHDL-Codes verwendet werden. Auch eine komfortable Verwaltung von Schaltungsteilen und Bibliotheken ist im Programm gegeben.

Eine weitere Möglichkeit, die das Tool zur Verfügung stellt, ist das Einbinden und Aufrufen von anderen EDA-Tools (Electronic Design Automation Tools). Es wird dabei automatisch ein Script generiert, was Anweisungen und Übergabeparameter enthält. So wird der komplette rechte Zweig im Design Flow (siehe Abbildung 2) von der VHDL-Designeingabe bis zur FPGA-Verifikation durch den HDL-Designer unterstützt.

7. VHDL-Entwurf

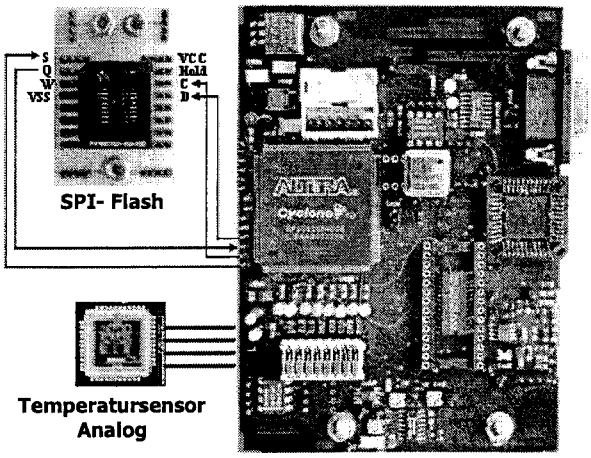
Bei der digitalen Schaltungsbeschreibung in VHDL muss auf einen synthetisierbaren Programmierstil geachtet werden, der sowohl für die FPGA- als auch die ASIC-Synthese geeignet ist. Es können darüber hinaus über Attribute Syntheseanweisungen und Constrains für ein besseres Ergebnis in den Code eingebunden werden. Auch eine Einbindung von Pin-Definitionen für das spätere FPGA-Design vereinfacht die weitere Vorgehensweise erheblich.

Maßgeblich für eine deterministisch einwandfreie Funktion ist jedoch ein synchrones Design. Asynchronitäten müssen daher vermieden werden.

Anfällig für Timing-Probleme ist auch das gemeinsame Bussystem. Alle Komponenten müssen über die gleiche Busarchitektur verfügen. Dies wurde im Projekt über einen Tristate-Bus und eine Chip-Select-Einheit realisiert, die genau definiert, welche Komponente zu welchem Zeitpunkt auf den Bus zugreifen darf.

8. Verifikation auf einem Testboard

Um das Gesamtsystem zu testen, wurde ein Testboard entwickelt.



Die digitalen Komponenten werden auf einem FPGA (Field Programmable Gate Array, hier ein Altera Cyclone) nachgebildet. Die analogen Komponenten sind als diskrete Schaltung oder bereits existierende integrierte Schaltungsteile ausgeführt. Desweiteren befinden sich Schnittstellen, Frequenzgeneratoren, Piezolautsprecher (Rückseite) und Spannungsregler auf dem Board, so dass alles vorhanden ist, um das Gesamtsystem in Betrieb zu nehmen. Das digitale FPGA-Design kann nun geladen und das WearLog

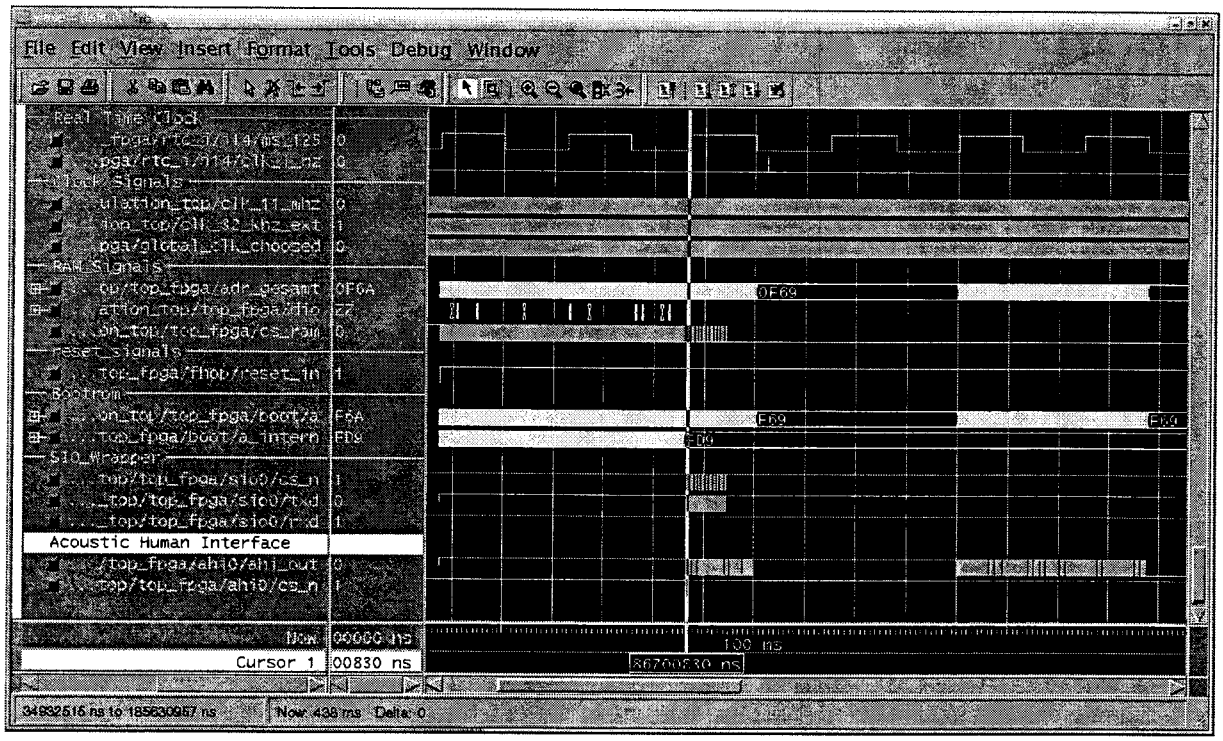


Abbildung 3 : Simulation WearLog (Digitalteil)

Projekt unter realitätsnahen Bedingungen getestet werden.

Gerade in diesem Verifikationsschritt werden immer wieder Fehler und Verbesserungsmöglichkeiten gefunden, die in einem späteren ASIC nicht ohne weiteres korrigiert werden können.

9. System Simulation

Das System lässt sich unter Ausklammerung der analogen Komponenten und Verwendung von Simulations-Files rein digital simulieren und verifizieren. Es können durch den vorhandenen Mikroprozessor sehr einfach verschiedene Programme ausgeführt werden.

Die Überprüfung kann auch durch spezielle Testprogramme (Testbenches) vereinfacht werden, indem eine korrekte Ausführung über Systemereignisse kenntlich gemacht wird oder die Ergebnisse ausgegeben werden. Dies erübrigt teilweise langwieriges Debuggen der Programme.

Abbildung 3 zeigt die Simulation des Ladevorganges aus dem Flash. Dieser Vorgang wird mit zwei Piepstönen quittiert, die auch in der Simulation durch die Schwingungen am „Acoustic Human Interface" zu sehen sind.

10. Stand der Entwicklung

Es sind mittlerweile alle benötigten Schaltungsteile für das System entwickelt und angepasst worden.

Das System kann (wie oben beschrieben) erfolgreich simuliert werden.

Für den späteren ASIC ist eine erfolgreiche Synthese unter Synopsys durchgeführt worden.

Derzeit wird versucht das Gesamtsystem auch im Hinblick auf die Sensorik weiter zu testen und eine einwandfreie Funktion sicher zu stellen.

Nach erfolgreichen Tests wird der Schaltkreis über einen so genanten Chip-Broker gefertigt.

11. Referenzen

[1] Dipl.-Ing.(FH) Wolfgang Vollmer, Prof. Dr. D. Jansen: FHOP-DESIGN-KIT, FH-Offenburg, 1999

[2] Handbuch der Electronic Design Automation, Prof. Dr.-Ing. Dirk Jansen, Carl Hanser Verlag, 2001

Geschwindigkeits-Steuerung eines Modells mit FPGA

Florian Grandmontagne, Andreas Bayer,
Peter Rieger, Tobias Kennerknecht
Betreuender Assistent: Dipl. -Ing.(FH) Christoph Weber
Hochschule Ravensburg-Weingarten, Doggenriedstraße, 88250 Weingarten
Fon: 0751/501-9686, Fax: 0751/501-9876
weber@hs-weingarten.de

Aus der Vorlesung „Computer Aided Engeneering" entstand das Projekt, einen Tempomat mit Hilfe eines FPGA's zu entwerfen und ein vorführbares Modell zu entwickeln.

Das Programm wurde in VHDL entworfen. Es handelt die Verarbeitung eines Drehgebers so wie weitere Steuersignale zur Bedienung ab, übernimmt die Regelung und gibt ein entsprechendes Steuersignal, so wie den Soll-Geschwindigkeitswert aus.

Als Vorführobjekt wurde ein Modellauto auf ein Gestell aus Plexiglass montiert, ein Drehgeber an einer Achse befestigt, so wie ein Bremsblock zur Simulation von Berg- und Talfahrten montiert. Der Motor wird durch eine modifizierte Motorsteuerung betrieben.

1. Einleitung

Die Basis des gesamten Aufbaus stellt ein Modellauto mit einem 6 Volt Elektromotor, einer Bremsvorrichtung, die als Störsignal eingreift sowie Bergfahrt und Gegenwind simuliert, einem Drehimpulsgeber, der auf der Antriebsachse des Modellautos befestigt ist, einer Motorsteuerung, welche die Drehzahl des Elektromotors steuert, einem digitalen Potentiometer und einem Prototypen-Board mit einem Spartan2 FPGA der Firma Xilinx.

Die eigentliche Steuerung des Tempomaten wurde in VHDL entworfen, seine Funktion in der Simulation geprüft und später mit dem Synthesetool „Leonardo" für den Baustein XCv250 - ein FPGA der Familie Spartan2 der Firma Xilinx - zur Implementierung vorbereitet. Der FPGA befindet sich auf einem Prototypen-Board der Firma XESS (XSA Board V1.2). Hier ist es mit Hilfe der XESS-Downloadtools möglich den FPGA sowohl direkt über die parallele Schnittstelle zu konfigurieren, als auch ein EEPROM (ebenfalls auf dem Prototypen-Board) mit den Konfigurationsdaten zu beschreiben. Die EEPROM-Variante hat den Vorteil, dass die Konfiguration bei Spannungsverlust im EEPROM gespeichert bleibt und beim nächsten Einschalten vom FPGA wieder geladen wird.

Mit Hilfe von Tastern, die sich auf dem XILINX-Board befinden wird ein Geschwindigkeits-Sollwert vorgegeben. Dazu sind zwei vorprogrammierte Festwert-Schalter sowie 10er- und 1er-Schritt-Schalter im Design vorgesehen.
Nun vergleicht der XILINX den eingestellten Sollwert mit dem vom Drehzahlsensor übermittelten Istwert.
Im Falle einer Abweichung gibt der XILINX eine 8bit Kombination auf die Eingänge des digitalen Potentiometers, welches daraufhin ihren Widerstand anpasst und somit die Motorsteuerung beeinflusst - den Motor also schneller oder langsamer laufen lässt.

2. Hardware

2.1.Motorsteuerung

Durch die Motorsteuerung wird die Drehzahl des Elektromotors beeinflusst. Aus Gründen der Zeitersparnis wurde diese als fertiger Bausatz bestellt und musste nur noch aufgebaut werden. Die Steuerung kann mit 12-24V betrieben werden und liefert maximal 10A Ausgangsstrom. Die Motorsteuerung arbeitet mittels einer Puls-Weiten-Modulation und besitzt zur manuellen Regelung der Drehzahl, ein analoges Potentiometer, welches für unsere Zwecke durch ein selbst aufgebautes digitales Potentiometer ersetzt worden ist.

Änderungen des Widerstandswertes am Potentiometer bewirken also eine Pulsbreitenänderung.

2.2.Digitales Potentiometer

Um die vom XILINX ausgegebenen 8bit Werte für die Motorsteuerung brauchbar zu machen, musste das vorhandene analoge Potentiometer zur Einstellung der Drehzahl durch einen Digital/Analogwandler ersetzt werden.

Die Grundfunktion ist eine Reihenschaltung einzelner Widerstände, die je nach eingegebenem Bitwert durch IC-Relais in die Widerstandsreihe hinzu- oder weggeschaltet werden.

3. VHDL-Entwurf

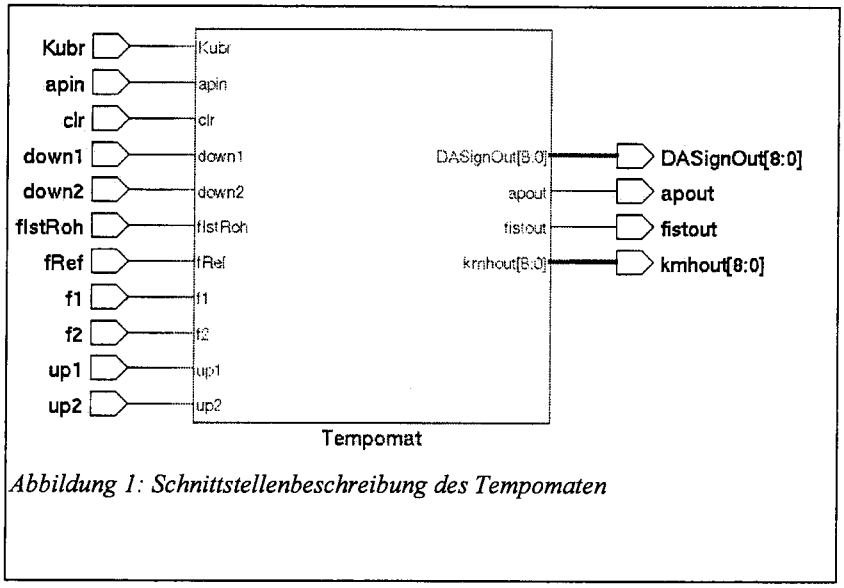


Abbildung 1: Schnittstellenbeschreibung des Tempomaten

3.1.Struktur

Der Tempomat besitzt die in „Abbildung 1" dargestellten Ein- und Ausgangs-Signale.

Die Einstellung der Soll-Geschwindigkeit erfolgt über eine Reihe von Schalter. Hierfür stehen zwei Festwertschalter „f1" und „f2" zur Verfügung, so wie vier weitere Schalter, um die Geschwindigkeit in zwei verschiedenen Stufen zu erhöhen und zu verringern. Dabei sind die vier Schalter so programmiert, dass sie je nach Dauer des Drückens, um so höher regeln. Die beiden Signale „apin" und „Kubr" können den Tempomat passivieren. „Kubr" steht hierbei für Signale die von der Kupplung oder der Bremse kommen. Mit dem Aktiv/Passiv – Wechselschalter „apin" kann der Tempomat aktiviert oder passiviert werden. Das Signal vom Drehgeber wird der Schaltung über „fIstRoh" zugeführt. Die letzten beiden Eingänge versorgen die Schaltung mit einem Referenztakt (fRef) und einem Reset-Eingang (clr).

Zur Motorsteuerung wurde das 8-bit breite „DA Signal" parallel ausgegeben. Die Signale „kmhout" und „apout" wurden verwendet, um den eingestellten Geschwindigkeitswert so wie den Zustand des Tempomat (Akiv/Passiv) anzeigen zu lassen. Das normierte Drehsensor-signal wurde zu Analysezwecken ebenfalls am Ausgang verfügbar gemacht.

Intern setzt sich das Programm aus folgenden Hauptblöcken zusammen(siehe Abbildung 2): Vorteiler, Eingabeteil (input), Vergleicher und Output. Des weiteren wird ein OR-Gatter verwendet.

Das Sensorsignal (f_{Ist}) wird im ***Vorteiler*** so normiert, dass eine Umdrehung der Achse einem Takt am Ausgang ergibt.
Dies ermöglicht die Verwendung beliebiger Sensoren am Eingang bei minimaler Änderungen im Programm, da diese sich auf den Vorteiler beschränken.

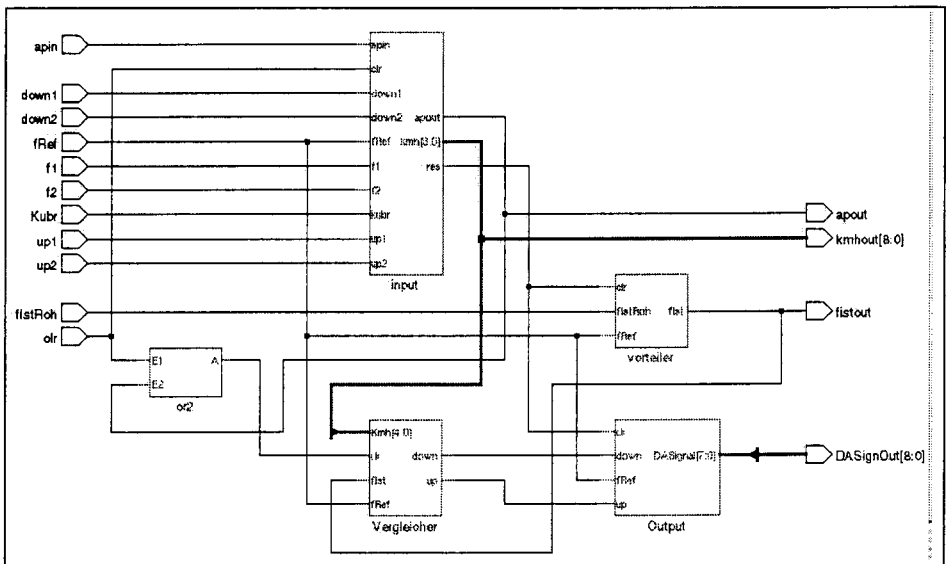


Abbildung 2: Interne Blöcke des Programmes

Liefert der Sensor bei einer Umdrehung 16 Takte, so wird intern ein Takt erzeugt, der um den Faktor 16 langsamer läuft.Der ***Eingabeteil*** bildet die Schnittstelle zum Anwender, so wie zum Rest des Automobils. Hier werden alle Schalter zur Einstellung der Soll-Geschwindigkeit, so wie zur Aktivierung und Passivierung des Tempomaten verarbeitet. Die Schalter für die schrittweise Geschwindigkeitsänderung verändern diese linear mit der Dauer des Drückens. Um den Tempomat zu aktivieren oder zu passivieren ist ein Schalter vorgesehen, der diesen bei jeweiligem Drücken in den anderen Modus schaltet (von aktiv zu passiv oder umgekehrt).
Weitere wichtige Signale, die zur Passivierung des Tempomaten führen, sind unter dem Signal „kubr"(~Kupplung/Bremse) zusammengefasst worden.
Als Ausgangssignal produziert der Eingabeteil einen internen Wert für die Km/h Zahl und übergibt diesen im Integerformat an den Vergleicher. Um die Tempomat-Regelung zu passivieren, wird ein „res"-Signal ausgegeben, welches den Vergleicher passivieren kann, und damit die Regelung der Geschwindigkeit ausser Kraft setzt. Um dem Anwender den aktuellen Status des Tempomaten anzuzeigen, wird das aktiv/passiv Signal „apout" aus der Schaltung heraus und an eine LED geführt.

Im Falle des aktiven Tempomaten findet im ***Vergleicher*** die Auswertung der aktuellen Geschwindigkeit statt. Die Passivierung läuft hierbei über ein „OR"-Gatter, welches dem „clr"-Eingangssignal vorgeschaltet ist. Als Eingänge besitzt das Gatter das allgemeine „clr" Signal so wie das interne Signal „res", welches vom Eingabeteil geliefert wird. Beide können somit also den Vergleicher passivieren.
Prinzipiell erfolgt die Regelung nach folgendem Schema:
Der Referenztakt f_{ref} ist mit 1 MHz vorgegeben. Mit diesem Takt wird eine halbe Periode von f_{IstInt} abgetastet. Die Anzahl der dafür benötigten Takte so wie weitere Informationen über die Dimensionen des drehenden Reifens machen die Berechnung der Geschwindigkeit möglich. Nach anschließendem Vergleich mit dem (Km/h)Soll-Wert ist der Vergleicher in der Lage, als Ausgangssignal eine Erhöhung, Verringerung oder das Gleichbleiben der Geschwindigkeit durch die Signale „up" und „down" anzugeben (Prinzip: Phase Locked Loop).

Im Wesentlichen besteht der Programmblock ***Output*** aus einem 8bit Zähler. Dieser wird entsprechend den eingehenden up/down-Signalen erhöht oder verringert und der Zählstand an den Ausgang weitergegeben.
Prinzipiell galt es den Ausgabeteil flexibel zu halten um, bei entsprechender Umrüstung im Hardware Teil, möglichst einfach das Programm neu anpassen zu können (Beispielsweise bei Einsatz eines 16bit D/A-Wandlers). Änderungen wären somit nur im „Output"-Teil vorzunehmen, ohne dass die weiteren Programmgruppen davon betroffen wären.
Die Schnittstelle zum „Vergleicher" ist sehr

einfach gestaltet. Hierbei ist eine Änderung der Wertübergabe denkbar. In der jetzigen Realisierung wird der Zähler linear verändert. Denkbar wäre eine nicht lineare Steuerung der Ausgangsgeschwindigkeit, abhängig von der Geschwindigkeitsdifferenz zwischen Soll- und Ist- Wert.

3.2.Geschwindigkeitsberechnung

Allgemeine Formel der konstanten Geschwindigkeit:

$$s = v_{Ist} \cdot t \rightarrow v_{Ist} = \frac{s}{t}, \quad wobei$$

$v_{Ist} \Rightarrow Geschwindigkeit$

$s \Rightarrow Strecke$

$t \Rightarrow verstrichene\ Zeit$

Berechnung der zurückgelegten Strecke 's' bei einer Umdrehung:

$$s = 2\pi r = d \cdot \pi, \quad wobei$$

$r \Rightarrow Radius\ des\ benutzten\ Reifens$

$d \Rightarrow Durchmesser\ des\ Reifens$

Berechnung der Umdrehungsdauer (T_{IstInt})

Messung des normierten Sensortaktes mit Hilfe der Referenzfrequenz (1 MHz).

$$f_{Ref} = 1MHz\left[\frac{1}{s}\right]; \quad mit$$

$$f_{Ref} \cdot T_{Ref} = 1$$

$$T_{Ref} = \frac{1}{f_{Ref}}$$

$$n \cdot T_{Ref} = T_{IstInt} = n \cdot \frac{1}{f_{Ref}} = T_{IstInt}; \quad mit$$

$n \Rightarrow$ *Anzahl der gezählten Tackte von* f_{Ref} *über eine Periode von* $f_{IstInt.}$

Damit ergibt sich für v:

$$v_{Ist} = d\pi \cdot \frac{f_{Ref}}{n} = \frac{d \cdot \pi \cdot f_{Ref}}{n}, \quad \left[\frac{m}{s}\right]$$

$$v_{Ist} = 3{,}6 \cdot \frac{d \cdot \pi \cdot f_{Ref}}{n}, \quad \left[\frac{Km}{h}\right]; \quad wobei$$

d in [m]

f_Ref in [Hz]

4. Zusammenfassung

Die generelle Funktionalität des Tempomate ist getestet und sichergestellt worden. Eingehende Signale von den Schaltern werden korrekt verarbeitet und sind über die Ausgänge hin verifizierbar. Das Zusammenwirken von Sensor und FPGA sowie FPGA und Motorsteuerung läuft planmäßig.

Als Erweiterungen sind eine komfortable Benutzerschnittstelle so wie ein kompakter Aufbau des Modells geplant.

Untersuchungen anderer Regel-Algorythmen zu Demonstrationszwecken sind vorstellbar.

Insgesamt bietet das Projekt „Tempomat" Spielraum für weitere Entwicklungen. Es ist ein vorzügliches Modell um eine auf einem FPGA realisierte Schaltung, in anschaulicher Art und Weise zu visualisieren und anfassbar zu machen.

Design Automation Conference, DAC 2005

in Anaheim, Kalifornien 13. – 17.06.2005

Prof. Dr.-Ing. Walter Lindermeir

FHT-Esslingen

Flandernstr. 101, 73732 Esslingen

Die Design Automation Conference (DAC) ist neben der International Conference on Computer Aided Design (ICCAD) weltweit die wichtigste Konferenz auf dem Gebiet der Entwurfsautomatisierung digitaler sowie analoger integrierter Schaltungen. Fünf Mitglieder der MultiProjekt-Chip-Gruppe (MPC) besuchten die DAC 2005.

Dieser Reisebericht skizziert einerseits Tendenzen der langfristigen Entwicklung in der Halbleiterindustrie. Darüber hinaus soll andererseits eine Auswahl aktueller Forschungsthemen vorgestellt werden, die sich voraussichtlich bald auf am Markt verfügbare Electronic Design Automation (EDA)-Tools auswirken werden.

1. Allgemeines zur DAC

Die DAC 2005 fand dieses Jahr im Anaheim Convention Center in Los Angeles statt. Sie war die 42. Konferenz ihrer Art, die seit 1963 jährlich durchgeführt wird.

Die Qualität der Konferenz misst sich an der niedrigen Akzeptanzrate der eingereichten Papers. Sie lag in diesem Jahr bei rund 23%, d.h. von 780 Einreichungen wurden 180 Beiträge akzeptiert.

Die Konferenzbeiträge wurden in einem Begutachtungsverfahren von international anerkannten Wissenschaftlern auf den betreffenden Fachgebieten ausgewählt, so dass sich die angenommenen Beiträge durch ein hohes Maß an Qualität auszeichnen.

Die Veranstaltung gliedert sich in zwei Teile: Zum einen das technische Programm, in welchem die angesprochenen wissenschaftlichen Beiträge in fünf parallelen Sessions vorgestellt werden. In dieses Programm sind mehrere Diskussionsforen eingebettet, in denen Vertreter von Hersteller- und EDA-Firmen sowie Wissenschaftler aktuelle Fachthemen diskutieren.

Zum anderen findet parallel zu den technischen Sessions eine Ausstellung von Firmen rund um die EDA- und Halbleiterbranche statt. In diesem Jahr präsentierten sich 250 Firmen. Dies waren deutlich mehr als in den vergangenen beiden Jahren, was als Indikator für eine gewisse Aufbruchstimmung in der Halbleiterindustrie gedeutet werden kann. Im Rahmen der Ausstellung finden zahlreiche Präsentationen und Vorführungen statt, die hier sowohl neue Produkte als auch Erweiterungen bestehender Produkte vorstellen.

Der erste und letzte Tag der Konferenz sind üblicherweise Tutorien gewidmet. So wurde u.a. ein Tutorium mit dem Titel „Statistical Performance Analysis and Optimization of Digital Circuits" angeboten. Diesem Thema, das eine grundlegend neue Methodik berührt, wird wegen seiner Aktualität auf dem nächsten MPC-Workshop ein eigener Vortrag gewidmet sein.

Nähere Informationen insbesondere zum Programm der Konferenz kann der interessierte Leser unter http://www.dac.com/ finden.

2. Hauptvortrag

Bernard S. Meyerson (IBM Fellow, Vice President and Chief Technologist, Systems and Technology Group, IBM Corp.) war der Hauptredner der Konferenz. Sein Vortrag trug den Titel: How Does One Define "Technology" Now That Classical Scaling Is Dead (and Has Been for Years)?

Meyerson stellte in seinem Vortrag heraus, dass die Hauptzielgröße der Halbleiterindustrie in den letzten vier Jahrzehnten die Performance der Halbleiterschaltungen war. Dies kann am Beispiel der Erhöhung der Taktfrequenz von Prozessoren nachvollzogen werden. Die Performance-Verbesserungen wurden durch den sog. Shrink-Pfad erreicht. Shrinken bedeutet, dass alle Dimensionen der Halbleiter um einen bestimmten Faktor im Verhältnis zueinander verändert werden (siehe Abbildung 1).

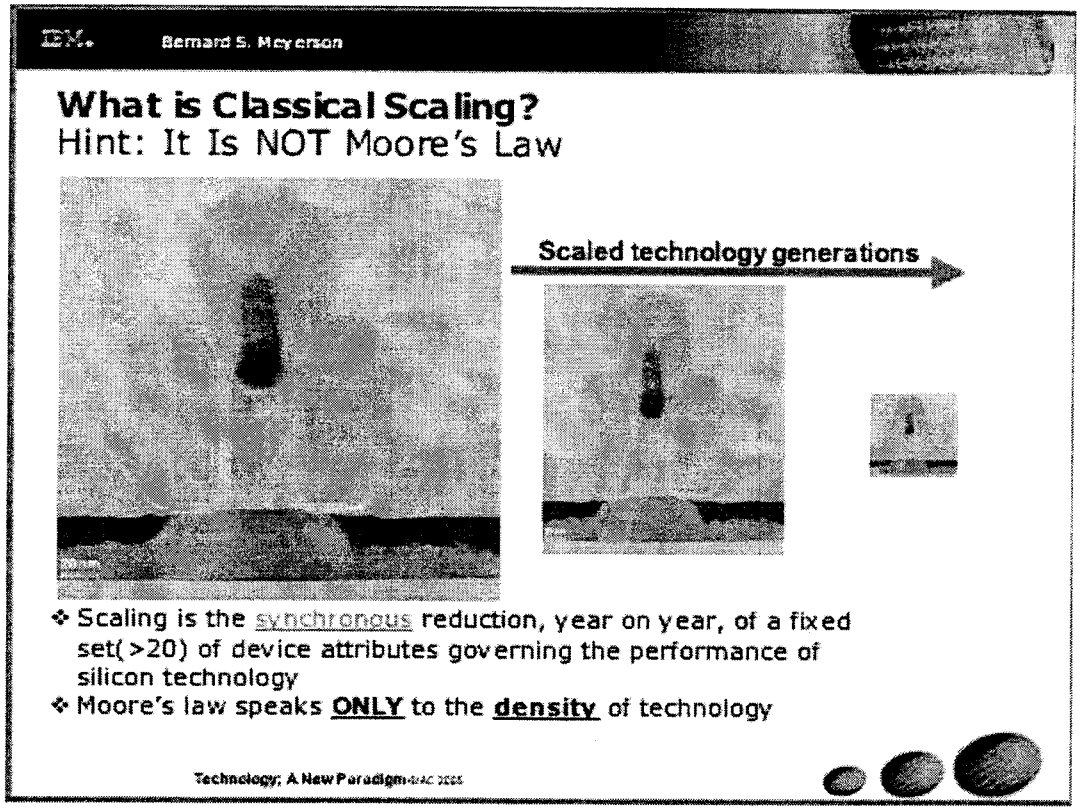


Abbildung 1: Klassisches Shrinken

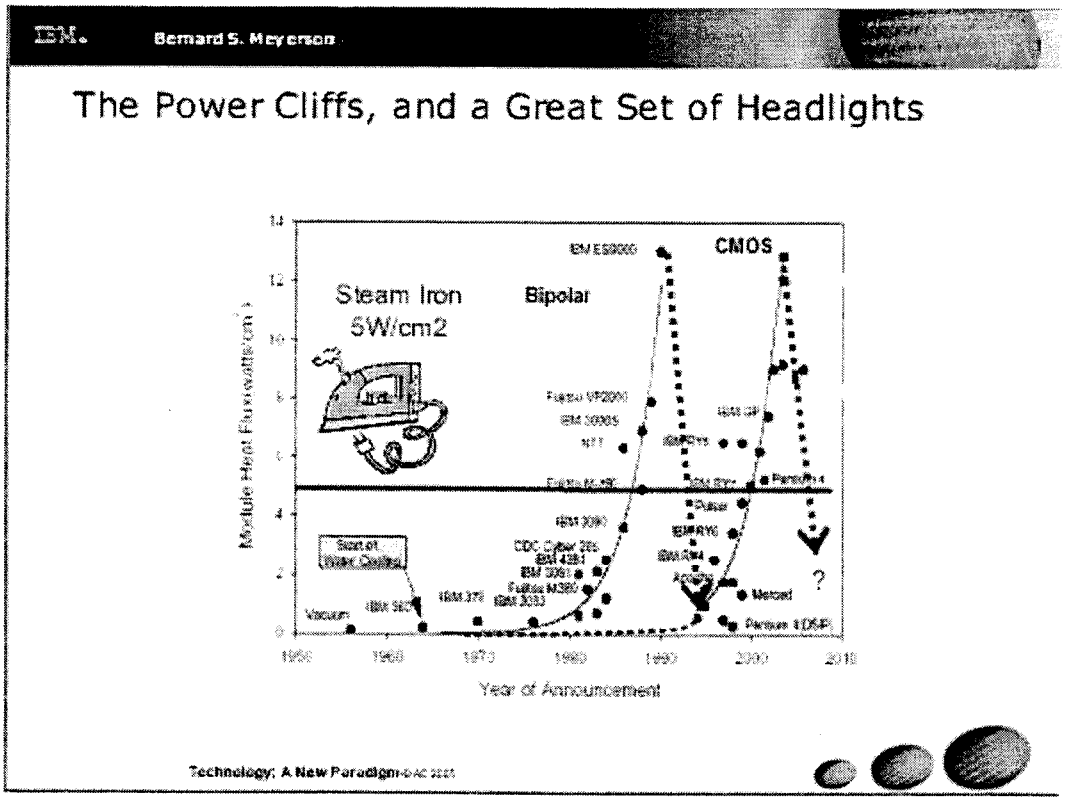


Abbildung 2: Power Cliffs

Neben den Dimensionen muss auch die Versorgungsspannung entsprechend verringert werden, um die Feldstärken konstant zu halten. Der Vorgang des Skalierens von Halbleiterschaltungen funktionierte bis zur 0,25µm-Technologie hervorragend. Durch das Scaling wurden Geschwindigkeit, Kosten, Verlustleistung sowie die implementierbare Komplexität der Schaltungen zugleich verbessert. Die Verkleinerung der Geometrien wurde nur durch die technologische Beherrschbarkeit der Herstellungsprozesse begrenzt.

Seit der 0.18µm-Technologie allerdings muss der Shrink-Pfad verlassen werden: Die Versorgungsspannung und die Oxiddicke können nicht einfach weiter nach den einfachen Shrink-Regeln verringert werden.

Die Ursache für das Abweichen vom Shrink-Pfad liegt darin, dass die Atome im Vergleich zu den Dimensionen, die auf dem Halbleiter realisiert werden, nicht mehr als klein angesehen werden können. So besteht das Gateoxid in aktuellen Technologien nur noch aus etwa 5-6 Atomschichten. Das bedeutet bei einem Defekt eines Atoms bereits eine erhebliche Änderung der elektrischen Eigenschaften des Transistors. Dies ist auch der Grund dafür, dass die Ansätze, das Laufzeitverhalten von Signalen auf dem Chip deterministisch vorherzusagen, hier zu kurz greifen und betont die Wichtigkeit der Verfahren des „Statistischen Entwurfs".

Die Verlustleistungsdichte löst als Zielgröße die Taktfrequenz ab. Es wird nicht damit gerechnet, dass Prozessoren mit wesentlich höheren Taktfrequenzen als heute auf den Markt kommen. Die Verlustleistungsdichte war Ende der 80er Jahre der Grund für das Ende der Bipolartechnologien im Bereich der Logikschaltungen. Nun stellt auch die Verlustleistung und dabei vor allem die statische Verlustleistung die Begrenzung für die CMOS-Technologie dar (siehe Abbildung 2). Meyerson betont in seinem Vortrag mit Nachdruck, dass es von großer Wichtigkeit ist, in dieser Situation ein starkes Engagement im Bereich Forschung und Entwicklung zu betreiben, weil Lösungen nicht mit weniger als fünf bis zehn Jahren Vorlauf erzielbar sind.

Als Beispiel für eine erfolgreiche Forschungsaktivität in dieser Richtung wird die Technologie „Strained Silicon" genannt. Durch Spannungen im Silizium wird die Beweglichkeit der Ladungsträger um bis zu 35.000% vergrößert, was die elektrischen Eigenschaften der Bauelemente überaus stark verbessert. Dies ist bereits in den 90nm-Technologien implementiert. Die einfache Skalierung der Transistoren muss also durch Innovationen und Forschungsaktivitäten abgelöst werden. Weitere Beispiele hierfür sind: Ultra-Thin SOI, High-k-Gate Dielectric, Double Gate CMOS uvm..

Als wichtige Richtung für weitere Innovationen betont Meyerson, dass nicht die Performance einzelner Prozessoren sondern die Leistungsfähigkeit von Systemen im Zentrum steht. Die Metriken mit denen Prozessoren verglichen werden, müssen geändert und durch applikationsspezifische Maßstäbe ersetzt werden. Als Beispiel hierfür dient der zurzeit leistungsstärkste Supercomputer BlueGene/L, der mit einer Rechenleistung von 70 Teraflops den bis dahin stärksten Earth Simulator (36 Teraflops) übertrifft. Hervorzuheben ist, dass dieser Supercomputer aus Chips aufgebaut ist, die aus je zwei PowerPC-Prozessoren mit einer Taktfrequenz von nur 800 MHz arbeitet. Die Gesamtzahl der Prozessoren dieses Supercomputers liegt bei etwa 131.000. Die Kommunikation der Prozessoren basiert auf einem innovativen Kommunikationssystem. Besonders zu betonen ist, dass BlueGene/L nur 1% der Größe und nur ein Achtundzwanzigstel der Verlustleistung des Earth Simulators benötigt. Dies unterstreiche die Notwendigkeit und die Erfolgsaussichten des „Holistic Design", des Entwurfs von Systemen aus Gesamt-Systemsicht.

3. Zusammenfassung einiger ausgewählter technischer Inhalte

Im Folgenden soll nur eine sehr begrenzte Auswahl an Themen berührt werden. Neben den hier angesprochenen Gebieten wurden auf der Konferenz auch wichtige Themen wie Signal Integrity, Low Power Design, hochfrequente integrierte Schaltungen, Wireless und viele andere mehr angesprochen. Nähere Informationen zu den technischen Beiträgen sind auf der angesprochenen Internetseite der DAC zu finden.

3.1. Electronic System Level (ESL)

In Session 36 und in einer Diskussionsrunde mit Firmenvertretern wurde über die aktuelle Anwendung von Methoden auf Electronic System Level diskutiert. Dabei geht es neben der funktionalen Beschreibung vor allem auch darum, die Performance und die Verlustleistung möglicher Implementierungen abzuschätzen und dadurch eine Machbarkeits- bzw. Marktchancenanalyse zu unterstützen.

Die momentan für diese Aufgaben zur Verfügung stehenden Hilfsmittel sind im Wesentlichen Erfahrung von entsprechenden Designern und Exeltabellen. Es

wurde herausgestellt, dass Matlab/Simulink sich im Wesentlichen zur Beschreibung des Datenpfades,

```
CASE sel[1:0] IS
WHEN "00" =>
  z <= a;
WHEN "01" =>
  z <= b;
WHEN "10" =>
  z <= c;
WHEN "11" =>
  z <= d;
END CASE;
```

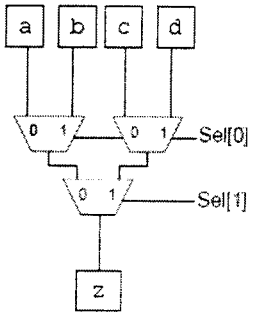


Abbildung 3: Multiplexer-Baum durch Case-Anweisung

```
IF c1 THEN
  z <= a;
ELSIF c2 THEN
  z <= b;
ELSIF c3 THEN
  z <= c;
ELSE
  z <= d;
END IF;
```

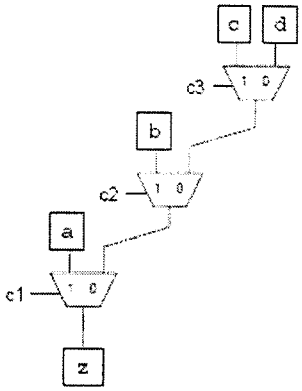


Abbildung 4: Multiplexer-Baum durch If-Then-Else Anweisung

nicht aber für Kontroll-Logik eignet. Alle Referenten betonten, dass die Verfeinerung in Richtung Hardware-Beschreibungssprache nicht automatisiert erfolgt (abgesehen von vorimplementierten Blöcken, die in einer Entwurfsbibliothek abgelegt sind). Vielmehr wird das abstrakte Modell als ausführbare Spezifikation betrachtet und dient dazu, die HDL-Implementierung zu verifizieren. Natürlich besteht hier auch das Problem, dass diese Modelle meist nicht zyklengenau sind. Der Aufwand der HDL-Implementierung ist im Vergleich zu den durchzuführenden Verifikationen von untergeordneter Bedeutung, so dass die Notwendigkeit zur Automatisierung dieses Schrittes nicht gesehen wird. Zur Verifikation werden Simulationsumgebungen verwendet, die das modulweise Austauschen der Modelle auf verschiedenen Abstraktionsebenen erlauben, wodurch die benötigte Rechenzeit in Grenzen gehalten werden kann. Es gibt zurzeit keine Möglichkeit, automatisiert vom ESL-Level aus auf anderen als den bereits in Firmen vorhanden Methoden auf die spätere Performance oder Verlustleistung einer späteren Realisierung zu schließen.

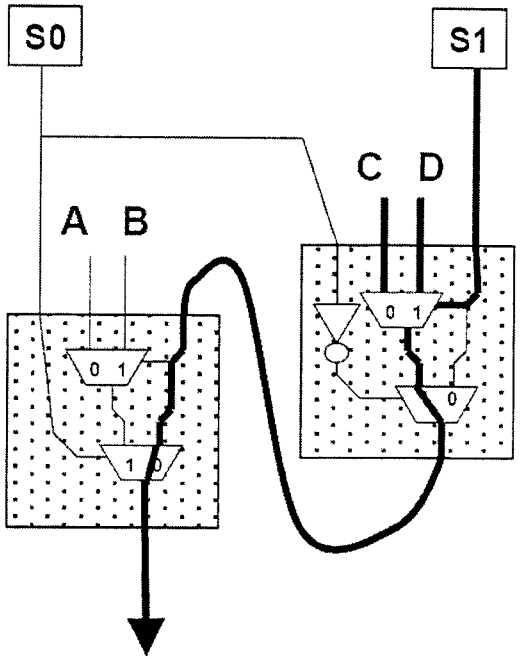


Abbildung 5: 4:1 Multiplexer in zwei 4-Input-LUTs

3.2. Neuerungen in Bezug auf FPGAs

In Session 27 wurde ein Ansatz zur Verbesserung der Synthese-Ergebnisse bei FPGA-Architekturen mit z.B. 4-Input-LUTs vorgestellt. Dabei wurde die Wichtigkeit von Multiplexer-Bäumen hervorgehoben. Diese Multiplexer-Bäume entstehen bei der Synthese von Case/Switch sowie If-Then-Else-Strukturen in HDLs (siehe Abbildungen 3 und 4). Beim Mapping dieser Strukturen auf 4-Input-LUTs wird jeder 2:1 Multiplexer auf ein 4-Input-LUT abgebildet. Dabei werden nur jeweils drei der vier Eingänge verwendet. Ein Eingang bleibt ungenutzt. Dies führt zu einem hohen Bedarf an logischen Elementen. Es wurde ein Ansatz vorgestellt, der es ermöglicht, einen 4:1 Multiplexer, der aus drei

2:1 Multiplexern aufgebaut ist, statt in drei 4-Input-LUTs nur in zwei 4-Input-LUTs abzubilden (Siehe Abbildung 5).

Die Effizienz des Verfahrens steigt, falls es auf Multiplexer-Busse angewendet wird. Da diese Multiplexer-Busse in den oben angeführten Case bzw. If-Then-Else-Strukturen auftauchen, falls in den verschiedenen Zweigen Signalzuweisungen an Busse stattfinden, ergeben sich für das Verfahren sehr gute Ergebnisse. Diese für das Verfahren günstige Konstellation tritt in HDL-Codes sehr häufig auf. Ergebnisse des Verfahrens sind in Abbildung 6 dargestellt.

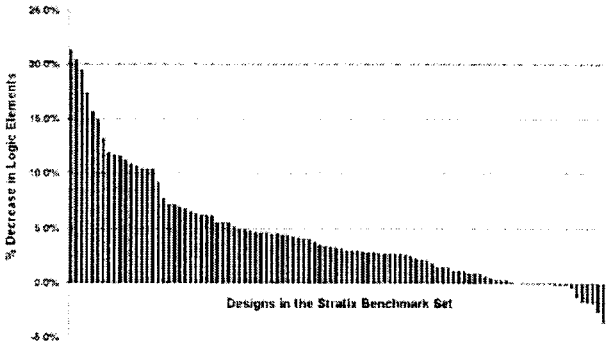


Abbildung 6: Ergebnisse des Syntheseverfahrens

Hervorzuheben ist, dass die erreichbare Performance der Entwürfe im Bereich ±10% schwankt, jedoch keine klare Tendenz erkennbar ist, so dass hier keine Korrelation zu Signallaufzeiten erkennbar ist.

In Session 44 wurde ein Verfahren vorgestellt, das durch gezielte Verringerung der Anzahl der in der Synthese nutzbaren Inputs/Outputs von LUTs die Anzahl der benötigten LUTs für einen Entwurf vergrößert, dabei aber die Verdrahtbarkeit des Entwurfes verbessert. Diese Begrenzung der Anzahl der Inputs/Outputs wird nicht global sondern nur in Regionen mit hoher Verdrahtungsdichte durchgeführt. Dadurch kann mit Hilfe dieses Vorgehens für einen Entwurf die Anzahl benötigter LUTs gegen die Anzahl benötigter Verdrahtungskanäle abgewogen werden.

Das Verfahren besitzt zwei Anwendungen: Zum einen, falls ein Entwurf in einem gegebenen FPGA nicht verdrahtbar ist, aber noch ungenutzte LUTs zur Verfügung stehen. Zum anderen, falls dies nicht möglich ist, kann das Vorgehen mit einem größeren FPGA derselben Familie wiederholt werden und man ist nicht notwendig gezwungen auf die nächst komplexere FPGA-Familie auszuweichen. Dies ist deshalb hervorzuheben, weil die Anzahl von Verdrahtungskanälen innerhalb einer FPGA-Familie konstant ist und nur die Anzahl LUTs erhöht wird. (Siehe Abbildung 7)

3.3. System on a Chip

In Bezug auf Tendenzen im Bereich der Systems on a Chip (SOC) wurden in Session 34 und in Tutorial 6 „Design of SOC with Embedded Processors" unter anderem auch folgende Aspekte dargestellt:

Kommunikationsstrukturen

Da die Anzahl von auf SOCs implementierten Prozessoren und Subsystemen immer weiter zunimmt, müssen verbesserte Konzepte zur Kommunikation dieser Komponenten bereitgestellt werden. Die Bandbreite reicht hier von applikationsspezifischen FIFO-Pufferspeichern für z.B. datenpfadlastige Anwendungen über evtl. hierarchische Bussysteme bis hin zu paketorientierten Router-Strukturen.

Zur optimalen Konfiguration hierarchischer Bus-Systeme werden automatisierte Verfahren auf hoher Abstraktionsebene entwickelt, die simulationsgestützt Entwurfkriterien wie Anzahl der Busse, Platzierung der Bridges, Bitbreiten, Bus-Taktfrequenz, Bus-Protokoll, Burstlängen etc. untersuchen.

In Bezug auf die paketvermittelten Kommunikationsstrukturen wurden Ansätze vorgestellt, die dynamisch konfigurierbare Router-Algorithmen speziell für On-Chip-Gegebenheiten anpassen.

Wichtig hierbei sind Cash-Coherence-Protokolle, da die verwendeten Prozessoren häufig über Caches verfügen. Hier werden bekannte Ansätze aus dem Bereich der Multiprozessoren auf die besonderen Gegebenheiten adaptiert.

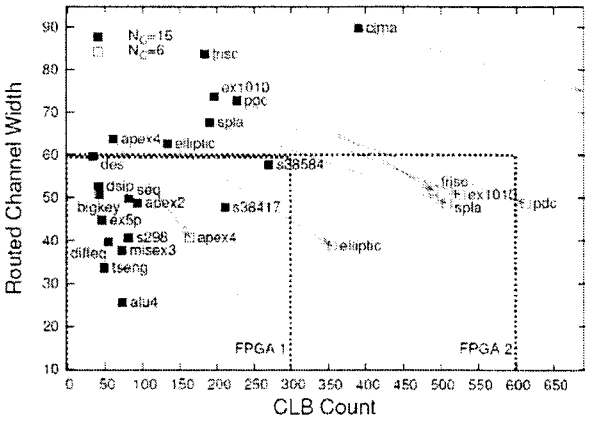


Abbildung 7: Erreichbare Trade-Offs

Konfigurierbare Prozessoren

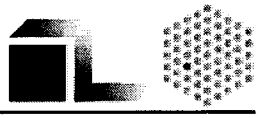
Im Rahmen von Architekturentscheidungen wird für Systeme ausgewählt, welche Funktionalitäten in Hardware und welche auf Prozessoren durch Software realisiert werden sollen. Dabei werden heute meist Standardprozessoren mit RISC-Architekturen verwendet. Dabei muss bei der Softwarerealisierung im Vergleich zur Hardwarerealisierung ein erhöhter Leistungsverbrauch, eine schlechtere Performance und ein größerer Flächenbedarf in Kauf genommen werden. Es wurde hier ein Ansatz vorgestellt, der auf konfigurierbaren Prozessoren basiert. Dabei wird das Instruction-Set des Prozessors gezielt so auf die Zielanwendung zugeschnitten, dass nur die für die zu realisierende Funktionalität benötigten Instruktionen implementiert werden. Dies bedeutet das Aufgeben von einigen Vorteilen der Software-Realisierung. Allerdings können die oben angesprochenen Nachteile stark gemildert werden, wobei entscheidende Vorteile der höheren Flexibilität und kürzeren Entwurfszeiten erhalten bleiben. Problematisch bleibt hier allerdings die Verifikation dieser Systeme.

3.4. Entwurf analoger Schaltungen

In der Session 51 wurden Ansätze für Verfahren der Synthese analoger integrierter Schaltungen dargestellt. Besonders hervorzuheben sind hier Ansätze, die zum einen in Richtung Performance-Modeling gehen, zum anderen in Richtung Macro-Modellierung/Response-Surface-Modelle. Es wurden Verfahren vorgestellt, die zu einer gegebenen Topologie einer Schaltung unter Berücksichtigung von schaltungstechnischen Nebenbedingungen ein Gebiet im Raum der interessierenden Eigenschaften der Schaltung bestimmen, das durch Parametrisierung der untersuchten Schaltungstopologie implementiert werden kann. Falls nun mehrere Schaltungstopologien für eine Aufgabenstellung auszuwählen sind, kann aus den Gebieten im Eigenschaftsraum erkannt werden, ob die betreffende Topologie die gestellte Aufgabe erfüllen kann. So kann z.B. ein passender Operationsverstärker für eine gegebene Spezifikation (Bandbreite, Slewrate, usw.) aus verschiedenen Topologien ausgewählt werden. Darüber hinaus wurden Verfahren vorgestellt, die es erlauben in einem Top-Down-Ansatz gegebene Spezifikationen für eine Schaltung auf die nächste Verfeinerungsebene abzubilden und später Bottom-Up die Schaltungen zu parametrieren. Zur Verifikation kann dann das entworfene Gesamt-System durch entsprechende Verhaltensmodelle simuliert werden.

4. Zusammenfassung

Der Besuch der DAC ermöglicht Einblick in aktuelle Entwicklungen auf dem Gebiet der Entwurfsautomatisierung integrierter Schaltungen. Die präsentierten Neuerungen betreffen im Wesentlichen inkrementelle Verbesserungen der Entwurfsmethodik und beziehen sich dabei auf praktisch alle im Entwurfsprozess auftretenden Ebenen. Besonders hervorzuheben ist eine Erweiterung des Entwurfs hin zu höheren Abstraktionsebenen (ESL) und das Aufkommen des statistischen Entwurfs integrierter Schaltungen. Die an die Konferenz angeschlossene Ausstellung eröffnet den Besuchern einen Einblick in die Aktivitäten der EDA-Industrie und zeigt die Möglichkeiten von momentan am Markt verfügbaren Tools.



Versatile Search Processor Array (VeSPA)*

Avi Epstein

EMBL, Scientific Core Facilities, Services and Technology Unit

Meyerhofstrasse 1, 69117 Heidelberg

(06221) 387-8349, epstein@embl.de

Abstract

Many biological applications require the comparison of large genome strings. General-purpose computers do not suffice as efficient search machines because of their limited throughput. The exponential growth of databases, which by far exceeds the evolution of general-purpose processors, imposes a severe challenge to computer architecture design.

A novel systolic array architecture for complex database searches is introduced here. It is intended for the implementation of very fast complex motif searches in biological databases. It is, however, a promising tool for all fields requiring extreme throughput of data and flexible motif definition like comparative genetics or high throughput screening.

A prototype ASIC of such a systolic array was implemented in 0.35μ CMOS technology. It is expected to achieve 200 Giga operations per second. Systems based on this architecture can achieve, using available 90nm CMOS technology, performance in the range of ExaOPS (10^{18} operations per second).

1. Introduction

The search engine described here is of particular importance for several application areas in the field of molecular biology and genomics. The features that make it attractive for such applications are the possibility to define a complex motif and search for it at high speed in large databases. The most obvious application is for motif searches in the gene database [1].

1.1. Bioinformatics

Bioinformatics is an interdisciplinary research area that deals with the computational management and analysis of biological information. Many bioinformatics projects deal with structural and functional aspects of genes and proteins, and many are related to the human genome project [2]. Data produced by research teams all over the world are collected in databases (*e.g.* EMBL nucleotide database [3]). Computational tools are then needed to analyze the collected data. One example is the homology search between genes of different species, performed in order to trace biological function resemblance or to trace the evolutionary process. However, the exponential growth of databases storing biological information makes it harder to search for useful information. Furthermore, the computing power required to perform data mining in a database is proportional to the database's size squared. Therefore, the current evolution of general-purpose processor performance cannot keep up with the required computing power. For example, the above-mentioned EMBL nucleotide database is growing faster than 1.7 times yearly. This calls for tripling of computing power yearly. The only way to deal with such growth is by the development of dedicated computer architectures.

1.2. Application to Bioinformatics

Proteins are chains of twenty different types of amino acids that are represented as strings of 20 different letters. The amino acids can be linked in any linear order. The sequence of amino acids (the primary structure of the protein) determines the 3-D structure of the protein and its function. Portions of the amino acid sequence form functional units known also as domains (or motifs).

The sequence information of proteins is stored in cells in the form of a DNA (Deoxyribonucleic Acid) molecule. The DNA molecule is a polymer consisting of a sequence of four types of sub-units or bases (adenine, guanine, cytosine and thymine).

The coding of the sequence of amino acids of a protein is by triplets of DNA bases. There are 64 possible combinations of the bases in a triplet and therefore there is a redundancy in the coding of DNA triplets to protein. An additional redundancy exists due to the fact that only some sections of the DNA

* This work was possible thanks to the Institute for integrated circuits at the Fachhochschule Mannheim, which allowed us to use their chip design facilities for the ASIC design.

(called exons) are used for coding of sequences of proteins. The sections in between exons are called introns and occupy about 97% of the length of the DNA. The collection of all genetic information of an organism (including both exons and introns) is called a genome.

Structural and functional information about proteins can be found in the Swissprot and TrEMBL databases [4]. The Swissprot database contains currently about 300,000 sequences with an average length of about 300 amino acids. Genome information of many species is collected in gene databases. There are hundreds complete genomes with lengths of 1.6 million to 3 billion bases [5]. The protein and gene databases require only 20 and 4 types of letters respectively for the representations of the different amino acids and bases.

Recently, methods using unsupervised machine learning have been developed for the detection of all maximal sequence patterns in any arbitrary set of proteins (*e.g.* the full non-redundant protein database of more than 800,000 entries) without alignment or enumeration [6]. These patterns, called seqlets, can be used to cluster protein families [7] or detect weak signals in a very sensitive and exhaustive manner [8].

The number of seqlets discovered in large datasets is overwhelming (in the order of several millions) and searching with them can be prohibitively slow. The amount of search patterns produced makes it impractical to perform database searches using computer programs. Thus, a hardware platform that can search the protein database and further identify patterns discovered in the learning set is of significant value.

1.3. Complex Motif Searches

A complex motif specifies a string of data units with more than one possible correct data unit in each location. Table I shows an example of a motif.

Table I

A complex motif description example

Position	Character
1	T
2	S or P
3	E or M or B or L
4	Q or R or S
5	S
6	* (any character)
7	not X and not Y
8	A..H

This example shows a motif of 8 positions (characters) with two possibilities in the second position (S or P), four at the third position (E or M or B or L), and so on. The number of possible combinations that match to this description (not counting the wild card in position six) is the multiplication of all the possibilities in all the motif positions (4608 possible strings in the example of Table I).

1.4. Hardware Acceleration

At present, implementation of complex motif searches is commonly done by running sequential programs on general-purpose computers (*e.g.* [9] and [10]) or general-purpose processor arrays (*e.g.* the MasPar [11]). The problem with such implementations is that the number of instructions grows linearly with the motif's complexity.

The problem is worse in applications where complete databases are compared against themselves or to other databases (*e.g.* [5]). Here the number of operations grows by the square of the database size.

The Versatile Search Processor Array (VeSPA) [12] is a massively parallel systolic array architecture. This parallelism is two-dimensional:

1. It simultaneously performs the comparison of all motif positions to a window with a length equal to the motif length.

2. For each character position, it considers all the alternative characters defined by the complex search motif simultaneously.

An additional level of parallelism can be achieved by populating search machine boards with multiple systolic array integrated circuits analyzing the database in parallel and comparing it to different search motifs. Furthermore, the calculation of the score and the threshold comparison operations that are also necessary (as described below) are also pipelined with the comparison operations of the individual character positions.

The search processor systolic array described here is not restricted to characters, but can be used for numerical values (*e.g.* signal levels). The scope of applications is therefore not limited to text searches and extends to any signal recognition and triggering applications as well as to high throughput screening applications (*e.g.* activity and toxicity profiles of lead compounds in high throughput drug screening [13]).

2. The Systolic Array

Figure 1 illustrates schematically the operation of the systolic array. Data from the database is shifted through a shift-register alongside a register containing the motif definition. At each clock cycle, each data in the shift register is compared to the character definition at the neighboring motif register. A match flag is generated for each motif position, which signifies if the data currently residing in the data register matches the definition. The number of active match flags is compared to a threshold and according to the result, a trigger signal is generated if the complex match condition is met.

The VeSPA acceleration of search assignments is achieved due to three unique features:

1. Coding of the alphabet that allows for fast parallel detection of complex patterns. The code allows the definition of multiple character values for each motif position (see 2.1).

2. Hardware implementation of a systolic array that permits evaluation of all motif positions in parallel.

3. The calculation of scoring values and comparison against a threshold enables on-the-fly signalling of a hit.

2.1. Character Coding

The operation of the VeSPA is based on redundant character code. The code is n-bit wide, where n is the number of possible characters in the character-set. Each character is represented by a single active bit-position in the code.

The motif code allows for multiple possibilities at a single character position by setting all the corresponding bit positions. As an example, in order to describe a code for the Boolean expression (A or C or Y) the motif code would be (where the LSB is representing the character 'A' and the MSB the character 'Z'):

"0100... ...0101"

Similarly, exclusion in the search motif can be defined by a code with 1s in the places of the allowed alternative characters and 0s in the places of the 'unwanted' characters. For example, the code to define a search for (not B and not D and not Y), is:

"1011... ...1010"

This coding simplifies the design of the search engine by enabling the parallel detection of matches in complex expressions.

2.2. The Single Character Element

An element for comparing a single character with a location in the motif (which may consist of up to 2^n options) is described in figure 2.

The character code (still in normal code, *e.g.* ASCII) is stored in the character register (n-bit wide). It is translated into a 2^n-bit code by an encoder. The motif character is stored in the motif-register (2^n-bit wide). The comparator is performing a bit by bit comparison of both 2^n-bit words. If there is a match in one or more of the bits, it produces an active match signal.

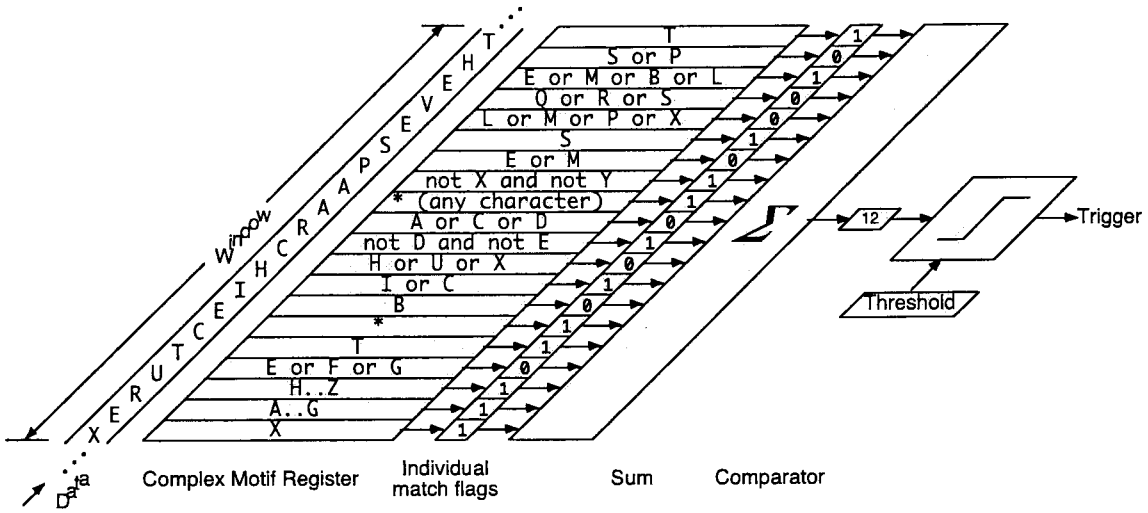


Figure 1: Schematic representation of the operation of the search processor systolic array with a numerical example.

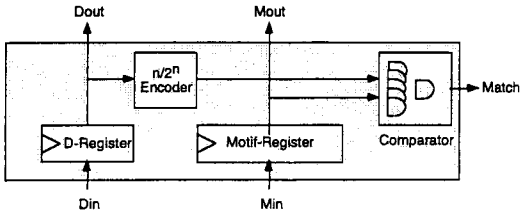


Figure 2: The basic single character element. *Din* is the data input, *Dout* the data output, *Min* the motif input and *Mout* the motif output.

The character element can alternatively be implemented using a 2^n wide D-register. That way the n/2n encoder is required only once (at the input to the element array). Such an implementation is shown in figure 3.

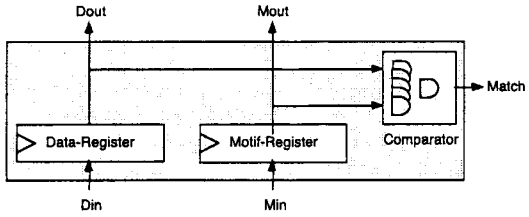


Figure 3: The modified single-character element with a wider data register and simplified logic.

The alternative wide D-register implementation eliminates the requirement for a n/2 ^{n} decoder at each character element at the cost of more flip-flops. Another advantage is the more regular array consisting of 2^n pairs of flip-flops and NAND gates (figure 4).

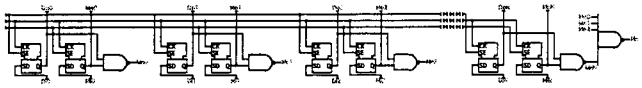


Figure 4: The implementation of the single character element. *Din*, *Don*, *Min* and *Mon* are data input n and output n lines and motif input n and output n lines respectively.

The flip-flops used in figure 4 are scan flip-flops, consisting of a regular D flip-flop and a multiplexer (figure 5). They contain a shift-enable (SE) input to control the data flow. Data is shifted in on the clock rising edge when SE is active and is frozen when SE is inactive. The actual implementation in the ASIC was using the DFS1 standard cell, which was intended for boundary-scan application.

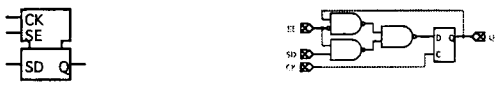


Figure 5: The internal circuit (right) of the scan flip-flops (left) used in the implementation of the single character element.

2.3. The Element Array

The main array of the VeSPA consists of a linear array of single character elements as shown in figure 6. The (complex) search motif is first shifted into the M-registers. Subsequently, the database flows through the D-registers.

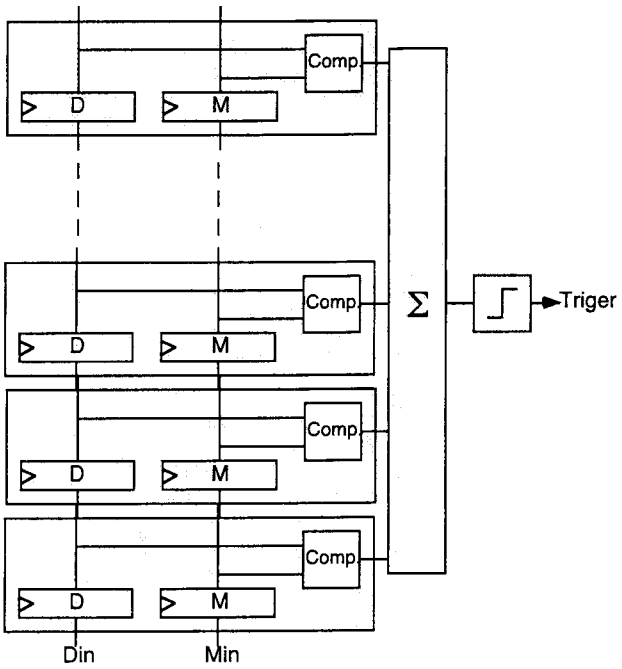


Figure 6: Block diagram of the systolic array.

The comparator output of each single character cell indicates if there is a match on that character. The sum of the match signals is than the score for this particular position in the database. In other words, it indicates how many characters match with the search motif. The threshold comparator unit can then generate a trigger signal. The schematic of the resulting array is shown in figure 7. The main array consists of 2048 scan flip-flops and 1024 2-input NAND gates. There are only short signal paths except for the clock, and data shift enable signals that require adequate low-skew tree structure.

2.4. Adder Tree and Wide OR-Gate Implementation

In order to fit the addition time into the clock cycle, a pipelined tree implementation similar to the one described in [14] can be used. A similar structure can also be used for the large OR of each character element in order to reduce the number of horizontal lines to be routed.

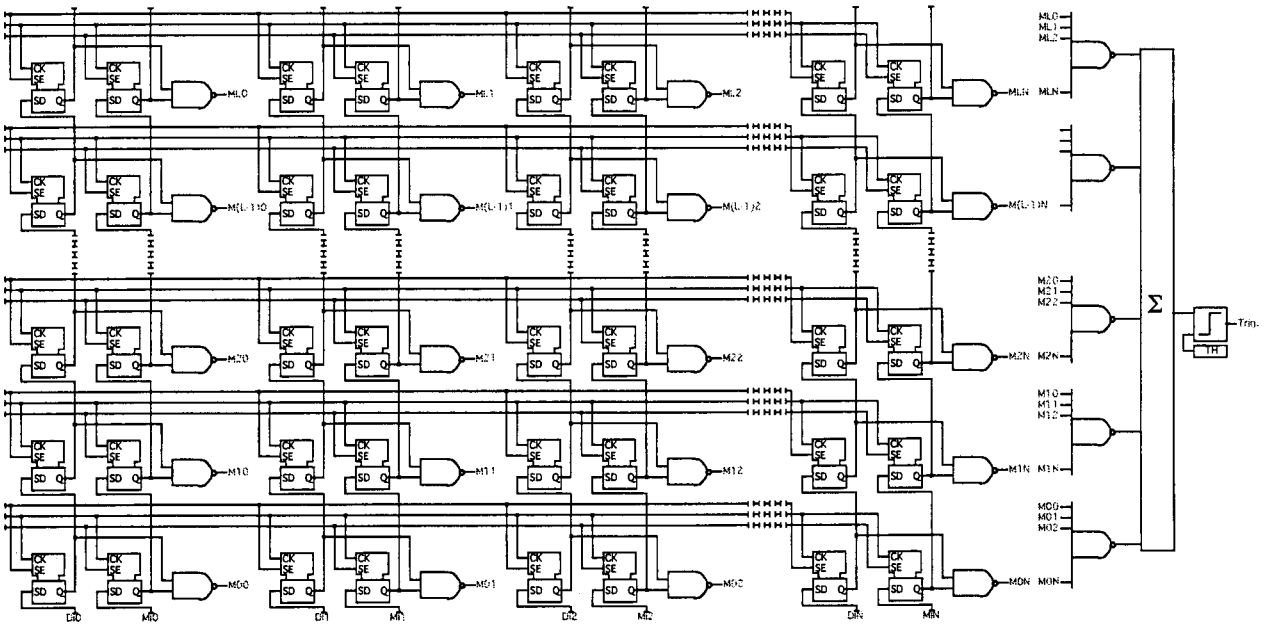


Figure 7: The implementation of the systolic array. Dn is data line n from an n/2^n encoder that decodes the database data on the flow. Mn is line n of the complex motive input.

2.5. Difference from a Correlator

The systolic array differs from a correlator in that it allows a flexible (multiple) definition of the search pattern. A correlator allows the definition of a search pattern of the form:

AIBICIDIEI...

Where each letter indicates the requirement to find this letter in the designated position.

The systolic array allows a search pattern of the form:

A+B+C...IX+Y+Z...I...

Here, for each motif's position, a definition of any combination of alternative characters (or values) can be given and the comparison with all the allowed characters for each position is performed simultaneously.

3. The ASIC Implementation

3.1. Technology

The first implementation of the VeSPA ASIC employs the AMS C35B4 0.35μ CMOS technology. Up to 32 ASICs will be populated on a mezzanine board, which will be mounted on an off-the-shelf PCI board (as described in the following sections).

3.2. Array Size

An ASIC implementation of a VeSPA is currently under development. It is designed specifically for bioinformatics applications, and therefore with character size to 5 bits, allowing a set of up to 32 characters (or values). This reduces the electronics in the main array by a factor of 8 compared to an 8-bit version. The motif length (the main array's length) is 32.

3.3. Interface

In order to reduce the pin count, the search motif (thirty-two 32-bit words) and threshold loading is de-multiplexed via 8-input pins. The overall slowdown of the search tasks is negligible due to the fact that only a single motif loading is performed for each scanning through the complete database.

The ASIC has three control inputs and a single trigger output that signals when a hit occurs. The trigger output is active when the number of matched positions between the data stream and the search motif is above the threshold. Table II contains a listing of the ASICs inputs and outputs.

Table II
Interface I/O

Pin	IO	Signal	Function
CLK	I	Clock	Shifting in data or parameters depending on the control inputs.
/PSE	I	Parameter Shift Enable	Active during parameter load phase. Parameters are shifted sequentially into the main array.
/DSE	I	Data Shift Enable	Active during normal operation. New data word is shifted into the main array each clock cycle.
Trig	O	Trigger Output	Indicates hits as the database flows through the array.
D	I	Data Input	5-bit database character input.
P	I	Parameter	8-bit parameter and threshold input.

4. Applying the VeSPA

Figure 8 illustrates a block diagram of a search engine based on the VeSPA. It consists of a memory to store the complete database, one or more VeSPA ASICs, a memory to store the results (hit addresses) and a host interface.

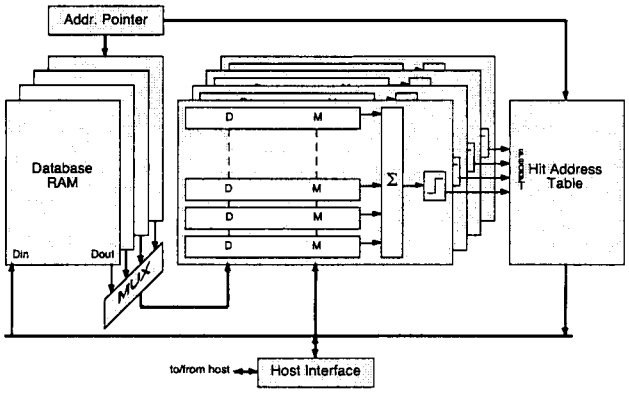


Figure 8: A complete motif search system based on the VeSPA.

The hit address table contains the addresses triggered as well as the trigger values. This enables the host computer to determine which of the motifs matched. The host computer first loads the search motif into the search engine. The search is then executed by shifting the database into the VeSPA. An address pointer keeps track of the location. The content of this address pointer is then transferred into the hit address table in the case of a match. The hit addresses are then read by the host-computer.

4.1. Memory Interfacing

In order to utilize the speed of the systolic array, data should be fed into it at a speed of one character per clock cycle. Considering that the clock cycle time can be reduced to the sub-nanosecond region, memory and bus bandwidth may become the bottleneck. A possible solution for the bandwidth problem is to use a banked structure for the database memory. This can be done either externally on the printed circuit board, or by integrating several input buses and a multiplexing mechanism in the ASIC.

4.2. The Alpha-Data ADP-XPI Board

The ADP-XPI board (Alpha Data, Edinburgh, UK) [15] is a flexible co-processor and digital video input board. Its block diagram is shown in figure 9. It consists of an FPGA (Xilinx 2VP70), a PCI interface, four banks of DDR SDRAM and static DDR SSRAM. A set of mictor connectors enables the interfacing of FPGA signals to a mezzanine board containing VeSPA ASICs. Up to 16 GByte of SDRAM can be used (4 GByte per bank). The board design enables data transfers at speeds of up to 160 MHz from the SDRAM to the mezzanine. The board also contains programmable user-clock generators. The hit-address table can be implemented in the SSRAM, which can enable the implementation of a FIFO with up to 256 Mega hit addresses.

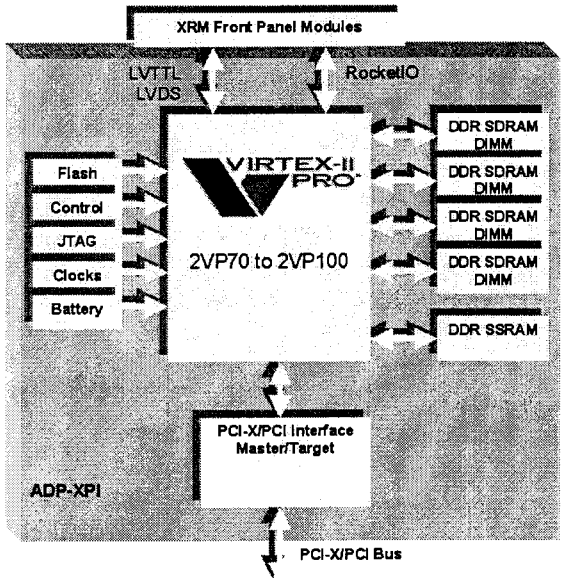
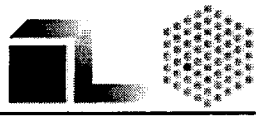


Figure 9: Block diagram of the ADP-XPI board.



Using the ADP-XPI board enables us to test a product shortly after ASIC samples are available. It suffers however from two limitations:

1. Restriction of the database scanning speed at slightly below the ASIC's limitation incurred by the on-board memory bandwidth.

2. The maximum database size to be scanned by a single board is limited by the RAM available to 16 GB.

4.3. The Mezzanine Board

In order to further increase throughput and price-to-performance ratio, the plug-in mezzanine board interfacing the main board to the ASIC is designed to carry up to 32 ASICs (figure 10). The data and parameter buses are fed in parallel to all the chips. Each chip is loaded with an individual search motif (using the individual MSE signal). During search operation, data flows in parallel through all the chips. The trigger signals are monitored and registered on the SRAM on the ADP-XPI board and subsequently transferred to the host.

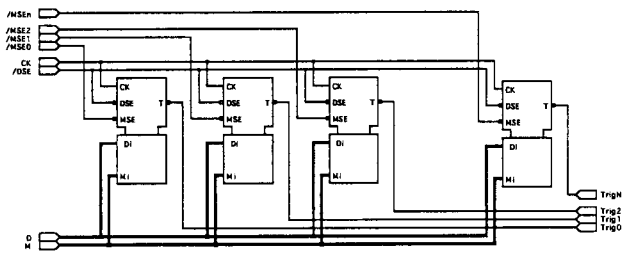


Figure 10: Block diagram of the mezzanine board.

Hit location readout uses part of the PCI bus bandwidth, this is however not a problem since, considering the application for large databases, we are only interested in motifs that are not too common. Therefore the impact of results transfer on the bus bandwidth usable for database transfer may be neglected.

5. Future Developments

5.1. Dedicated Board

The design of a dedicated main board would enable utilizing the full speed of the systolic array. Such a board will contain local memory both for database storage and hit results and interfaces between the database, the host computer, and the systolic array (or arrays). The interface to the database memory also enables the multiplexing of multiple SDRAM modules in order to achieve the required throughput.

A single host computer may be serving multiple search machine boards. The host downloads once the database into the local (on-board) database memory. Afterwards, motif descriptions will be loaded into the systolic arrays (limited by the number of arrays installed on board) and a search is initiated. Hit locations may then be transferred back to the host during the progress of the search.

Access to the database memory is only necessary for initial download, verification and update. The communication with the host is restricted to motif description and search results.

5.2. Array Expansion

The array can be expanded in both dimensions:

1. Character set.

2. Motif's length.

In order to extend the utility of the search machine, a second generation ASIC is planned with a main array element designed for 8-bit data (or 256-character set). This will expand the application possibilities of the machine to full text searches and to applications requiring higher value resolution.

Considering, for example, standard cell implementation using AMS 0.35 μ CMOS technology (with 273 μm^2 flip-flop area), a single character element is expected to require about 0.1 mm^2. This makes it feasible to implement an array of 256 or even 512 character elements on a single chip (requiring about 25 and 50 mm^2 respectively).

5.3. Multiple Array ASIC

Because of the small dimensions of the systolic array, it is feasible to produce a chip containing an array of such arrays. Using 0.13μ CMOS technology will enable the implementation of an array of 256 such systolic arrays along with readout circuitry for the hit vectors on a single integrated circuit. One method of implementing a multiple array ASIC is by feeding all the systolic arrays through a data bus as shown in figure 10. The data bus routing to a very large number of arrays adds a significant overhead to the IC surface. Moreover, the large fan-out introduces a capacitive load to the input signal, which would either restrict clock frequency or require a tree structure with additional pipeline registers.

5.3.1 Cascading VeSPA Arrays

The cascade expansion scheme shown in figure 11 solves these problems. Cascading arrays results in short connections, similar to the inter-array routing. The disadvantage of the scheme is the non-synchronized trigger signals. This can be managed by additional logic (*e.g.* shift registers to realign the triggers) or by the host computer.

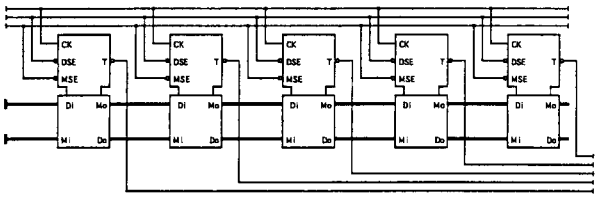


Figure 11: Cascade expansion of the VeSPA systolic array.

5.3.2 Complex PE

Another method to increase the ASIC's performance is by the use of more complex PEs. The PE can be expanded to handle more than one motif as shown in figure 12.

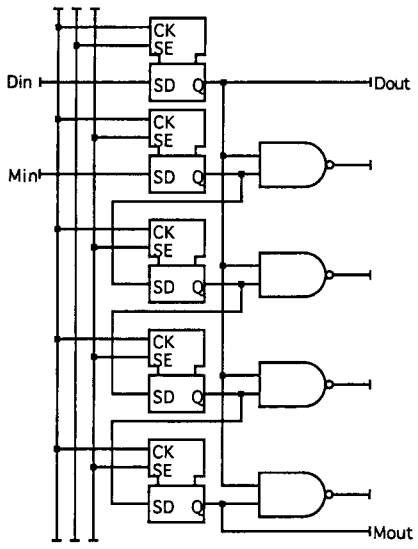


Figure 12: Complex PE to handle simultaneous comparison with four motifs.

The signal routing in arrays using multiple motif PEs is becoming more complex with the number of motifs the PE can handle. The congestion occurs between the individual PEs and the adder. Therefore, for optimal usage of space on the IC, both array cascading and PE expansion should be used. The choice of PE complexity depends on the word's width (character set), the motif's length, and the available routing resources (metal layers) in the technology used as described in [16].

6. Performance

The first VeSPA ASIC was designed using 0.35μ CMOS. The technical data is described in Table III. The performance is expected to improve at a very fast rate as we migrate to 0.13μ CMOS technology and implement a multiple array.

Table III

Technical Specification of the VeSPA ASIC

Process	AMS 0.35μ 4M CMOS
Dimensions	2.4 x 1.5 mm
Frequency	200 MHz
Array size	32 Columns / 32 Rows
Pipeline stages	8
Package	SOJ 28
Prototype	Sep. 2005

6.1. Standard Cell Implementation

To demonstrate the possible performance of search machines based on the VeSPA architecture and designed using standard cells, we will consider four benchmarks:

1. Single ASIC of the first generation.
2. Alpha-Data ADP-XPI Board with 32 ASICs.
3. Second generation ASIC with 256 Arrays (0.13μ).
4. Crate with 32 dedicated boards, each containing 32 ASICs of the second generation.

The four benchmarks, two single IC and two single board devices, are summarized in Table IV.

Table IV

Benchmarks

	CMOS	Board	ICs	Clock
1	0.35	ADP-XPI	1	200 MHz
2	0.35	ADP-XPI	32	200 MHz
3	0.13	Dedicated	1	1 GHz
4	0.13	Dedicated	32	1 GHz

The resulting performance values of the four benchmarks are summarized in Table V.

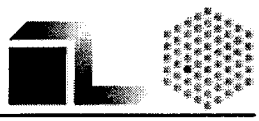


Table V

Performance

	IC	PE	Clock	OPS
1	1	1024	200 MHz	$200 \cdot 10^9$
2	32	32 K	200 MHz	$6.4 \cdot 10^{12}$
3	1	256 K	1 GHz	$260 \cdot 10^{12}$
4	32	8 M	1 GHz	$8.4 \cdot 10^{15}$

6.2. Custom Design

The numbers in table V are based on available current standard cell CMOS technology. If a custom design is considered, the available 90nm CMOS technology would enable a further increase of performance by a factor of four both in number of PEs and in clock speed. A crate with 32 boards, populated with 32 ICs each, implemented in 90nm CMOS would therefore yield performance of up to 4 Exa (10^{18}) operations per second.

7. Conclusions

The highly parallel architecture of the VeSPA systolic array ASIC makes it ideal for the implementation of extremely fast search machines, capable of complex motif searches in large databases. It is a promising tool for all fields requiring very high throughput of data and flexible motif definition like comparative genetics or high throughput screening.

Using available CMOS technology, systems based on this architecture are capable of achieving performance in the order of magnitude of 10^{18} operations per second.

8. References

[1] Lemoine E, Quinqueton J, Sallantin J, "High speed pattern matching in genetic data base with reconfigurable hardware," in: Proc. Int. Conf. Intell. Syst. Mol. Biol. 2, 269-275, AAAI Press, 1994.

[2] The International Human Genome Mapping Consortium, "A physical map of the human genome," *Nature* 409, pp. 934-941, 2001.

[3] G. Stoesser, W. Baker, A. van den Broek, E. Camon, M. Garcia-Pastor, C. Kanz, T. Kulikova, R. Leinonen, Q. Lin, V. Lombard, R. Lopez, N. Redaschi, P. Stoehr, M. A. Tuli, K. Tzouvara, R. Vaughan, "The EMBL Nucleotide Sequence Database," *Nucleic Acids Res.*, vol. 30 (1), pp. 21-26, 2002.

[4] C. O'Donovan, M.J. Martin, A. Gattiker, E. Gasteiger, A. Bairoch and R. Apweiler, "High-quality protein knowledge resource: SWISS-PROT and TrEMBL," *Brief. Bioinform.* 3, pp. 275-284, 2002.

[5] EBI: http://www.ebi.ac.uk/genomes/index.html, 2005.

[6] I. Rigoutsos and A. Floratos, "Combinatorial pattern discovery in biological sequences: the TEIRESIAS algorithm," *Bioinformatics* 14, pp. 55-67, 1998.

[7] A. J. Enright and C. A. Ouzounis, "GeneRAGE: a robust algorithm for sequence clustering and domain detection," *Bioinformatics* , Vol. 16, pp. 451-457, 2000.

[8] I. Rigoutsos, A. Floratos, C. Ouzounis, Y. Gao and L. Parida, "Dictionary building via unsupervised hierarchial motif discovery in the sequence space of natural proteins," *PROTEINS* 37, pp. 264-277, 1999.

[9] T. Rogens and E. Seeberg, "Six-fold speed-up of Smith-Waterman Sequence Database Searches Using Parallel Processing on Common Microprocessors," Bioinformatics Vol. 16 (8) pp. 699-706, 2000.

[10] G. J. Barton, "SCANPS Version 2.3.9 User guide," University of Dundee, UK, 2002.

[11] S. S. Sturrock and J. F. Collins, "MPsrch V1.3 User Guide," Biocomputing Research Unit, University of Edinbourgh, UK, 1993.

[12] A. Epstein "Schaltung zur Verarbeitung von Daten, Kennwort: Fast and flexible processor integrated circuit for motif searches," Deutsche Patent DPA 10106340.7, February 16th, 2001.

[13] P. James, "Subtle, Perceptive Drug Discovery: A contradiction?," Current Drug Discovery, July 2002, pp. 17-19, 2002.

[14] A. Epstein, G. U. Paul, B. Vettermann, C. Boulin and F. Klefenz, "The Parallel Hough-Transform Systolic Array ASIC," *IEEE Trans. on Nuclear Science*, vol. 49, 2002.

[15] Alpha-Data: http://www.alpha-data.com/adp-xpi.html 2005.

[16] A.E. Dogbe, G. U. Paul and A. Epstein, "Kontinuierlicher Vergleichsprozessor für große Datenmengen," Diplomarbeit (in progress), FH-Mannheim, 2005.

Implementation of a Radar Environment Simulator using Matlab/Simulink and Xilinx Systemgenerator

Matthias Neuber*, Ralf Gessler*, Thomas Mahr**

*Hochschule Heilbronn, Standort Künzelsau, Daimlerstr. 35, 74653 Künzelsau
E-Mail: gessler@fh-heilbronn.de

**EADS Deutschland GmbH

A radar environment simulator (RES) is essential for development of radar processing algorithms [1]. We do not only need a simulation in MATLAB or Java but also a realisation on FPGA which simulates the physical interface from AD-converter to the radar processor. Therefore, we describe the system in an abstract way independent from concrete implementation, namely by use of UML – the Unified Modelling Language [2]. Based on an UML specification of the system, RES is implemented in MATLAB and Simulink.

The system Radar Environment Simulator interacts with the neighbouring systems on a high level. The neighbouring systems are environment, radar and radar signal processor. The environment includes targets, noise and clutter. The radar controls the sensor characteristics (instrumented range, azimuth resolution, timings, etc.). The radar signal processor receives the radar signal from the AD-converter.

This paper introduces a novel development flow for system level design using MATLAB/Simulink and Systemgenerator [3]. The Systemgenerator from Xilinx closes the gap between the system level design and its actual hardware implementation [4].

1. Radar Signal Processing

For developing a radar system we need a real-time simulator for environment modelling.

1.1. Radar

A naval and ground radar consists of an antenna, transmitter/receiver, signal processor and tracker. The radar signal processor itself receives radar echo signals measured by an antenna, filters the signals by separating target signals from clutter signals, classifies targets and sends the filtered target signals to a target tracking processor(see figure 1).

The algorithms used in the radar signal processor are: spectrum analysis algorithms (used for pulse compression and moving target detection), constant false alarm rate algorithms (CFAR), target classification algorithms and others [1].

Customers of naval-, ground- and airborne-radar systems require low costs, high performance, real-time behavior, low latencies, accurate signaling, high communication bandwidth, life-cycle support of about 20 years, high reliability, good maintainability, convenient control and test facilities flexibility, short time to market, and upgrade potential.

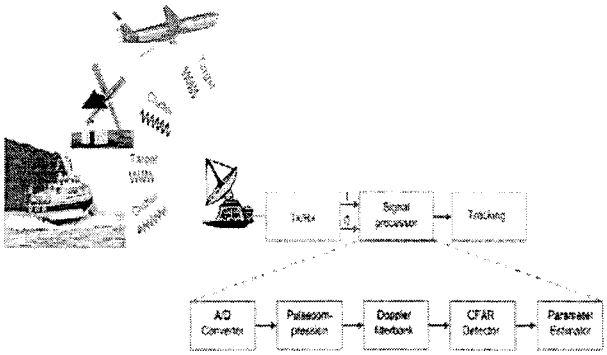


Figure 1: radar data processing flow

1.2. Radar Environment Simulator

In order to develop new algorithms for the signal processor, it's necessary to model the environment with its targets, noise and clutters. A hardware is needed which can simulate the environment in real-time. Such a system is called Radar Environment Simulator RES (see figure 2).

This claim fulfils a connection on programmable logic device - a FPGA (see chapter 3.).

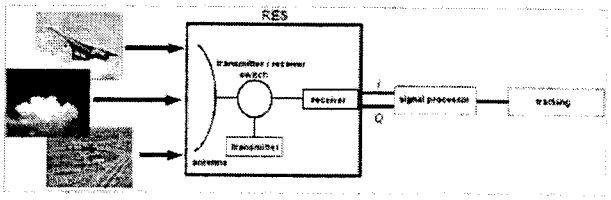


Figure 2: Radar Environment Simulator (RES)

2. Designflow

MATLAB and Simulink (graphical simulation) are most popular tools for algorithm and system simulation purposes. The Systemgenerator from Xilinx closes the gap between the system level design and its actual hardware implementation. For system-level-specification we use UML.

2.1. System Level Design with Systemgenerator

On system level, we use MATLAB/Simulink to model the radar signal processing algorithms (see figure 3). Several scenarios and configurations can be simulated. The novel design flow with Simulink / System Generator makes simulations on implementation level and automatic FPGA-mapping possible. Basic signal processing algorithms like FIR filters or FFT can be implemented and simulated rapidly. The design is implemented and simulated with Simulink / Systemgenerator using the Xilinx blockset (IPs). The Systemgenerator produces VHDL code for synthesis and a testbench for VHDL simulation.

2.2. Xilinx ISE

The generated VHDL-Code and testbench from Systemgenerator (see chapter 2.1.) can be read by ISE.

The ISE is an integrated development tool for programmable logic devices from Xilinx.

This tool makes it possible to create a design from different sources:

- HDL (VHDL, Verilog HDL, ABEL),
- „Schematic design"-files,
- EDIF-files,
- NGC/NGO-files,
- State Machines, and
- IP Cores.

After modelling, ISE automatically fulfils the process from synthesizing to generating a so called bit stream for the FPGA or CPLD in three steps. After all these steps, the device can be simulated to find possible errors. Now the developing steps from synthesis to bit stream in more detail:

The synthesis step checks the syntax for errors, before the VHDL-Code gets converted to logic cells (AND, OR, etc.). For the synthesis process, ISE allows to use three different tools:

- XST from Xilinx,
- Leonardo Spectrum from Mentor Graphics Inc. and
- Synplify und Synplify Pro from Synplicity Inc.

After converting the system to logic cells, the device must now be implemented and prepared for programming the FPGA or CPLD (Implementation). These stages can be differed:

- Translate,
- Map and
- Place&Route.

The translate stage merges the user constraints for the device with the device. Afterwards the device gets fitted to the specific FPGA or CPLD (map stage). Afterwards the components on the FPGA or CPLD get placed and routed (Place&Route). Finally ISE produces a so called Bit-Stream who programms the FPGA or CPLD (FPGA programming).

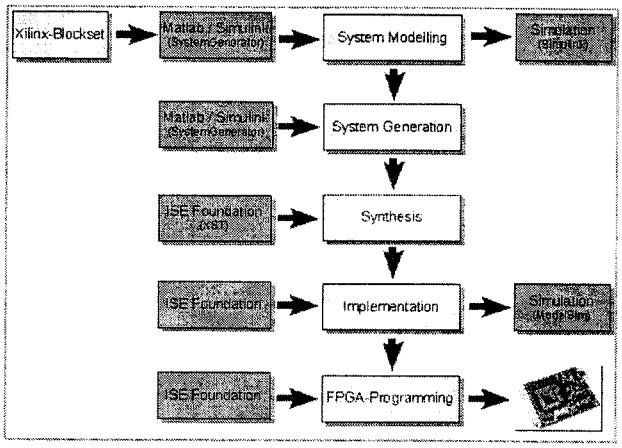


Figure 3: Design flow

3. Implementation

The implementation steps of RES are: UML specification, Matlab verification, Simulink implementation and the simulation of different levels and tools.

3.1. UML Specification

Figure 4 shows three interfaces to the radar environment simulator (RES). There are the two inputs environment controller and radar controller.

The user defines the possible objects (position, dimension, etc.) with the environment controller. The sensor characteristics (instrumented range, azimuth resolution, timings, etc.) are defined by the user in the radar controller.

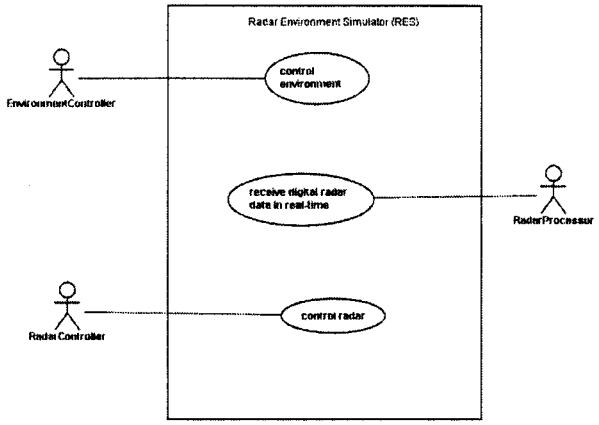


Figure 4: High level use case diagram of radar environment simulator and interaction with neighbouring systems

With those two actors, the RES generates the defined environment, and passes a complex signal of the environment to the radar processor, who itself filters the signal in order to separate the targets from the noise and clutters.

3.2. Simulink - Model

As in chapter 2.1. already mentioned , Xilinx allows to develop a design with Systemgenerator. This program consists of an own library, the Xilinx blockset, which is implemented in the standard Simulink library. With the Xilinx blockset, five different types of sources can be implemented in the design (see figure 5).

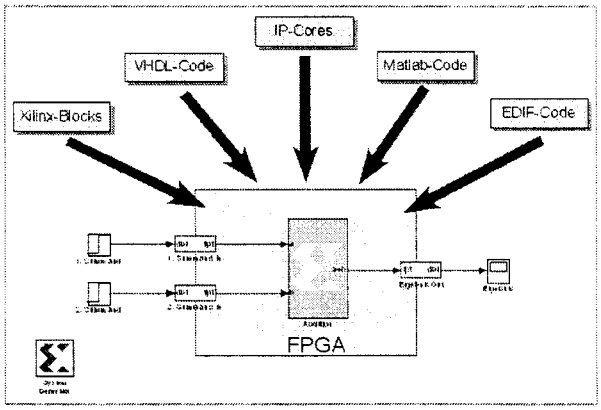


Figure 5: Design components

Xilinx-Blocks (IP-Cores) are pre-defined and optimized functional blocks like FIR-filters. Some of these blocks can also be parameterized (bit-length of the input and output, etc) by the user with the Core Generator, who is part of ISE. The „Core Generator" generates a VHDL-Code in which the IP-Core is implemented. This code can finally be implemented in the design over a „BlackBox".

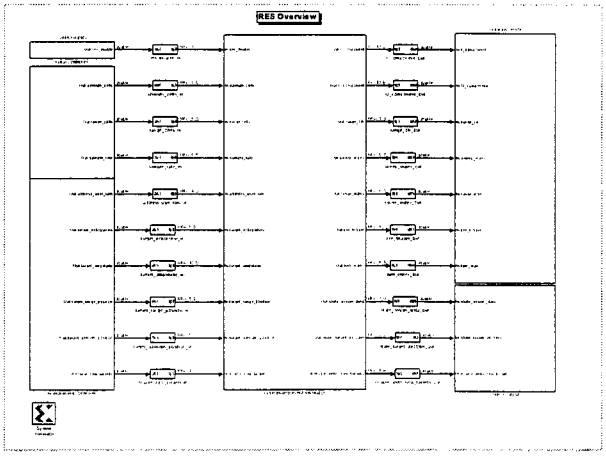


Figure 6: Simulink realization of Radar Environment Simulator

The designer has the ability to put his own VHDL- or EDIF-Code into his design with the so called „Black Box" from the Xilinx Library. Matlab functions can be implemented in Simulink with the „Mcode" Block. With the Xilinx-Library, the user has now the possibility to describe a design in high-level. One possible application can be the developing of a radar environment simulator (see also chapter 1. radar signal processing).

The figure 6 shows the RES, created in Simulink. You can see the yellow boxes who represent the ports of the FPGA. The RES itself (in the middle of figure 6) consists of IP-Blocks, like Adder, a generator who produces white Gaussian noise figures, state machines and „BlackBoxes" in which a VHDL Code is implemented.

4. Simulation

There are different levels of simulations.

4.1. Matlab/Simulink

The design, generated with the Systemgenerator, can be simulated in Simulink as if the design was build up with the standard Simulink library. The only difference is, that a „gateway out" has to be put between the design and the scope from the Simulink library.

Within Simulink an implemented VHDL -Code, over the „Black Box"-Block, can also be simulated. Therefore ModelSim, a VHDL simulator, can be used (see figure 7).

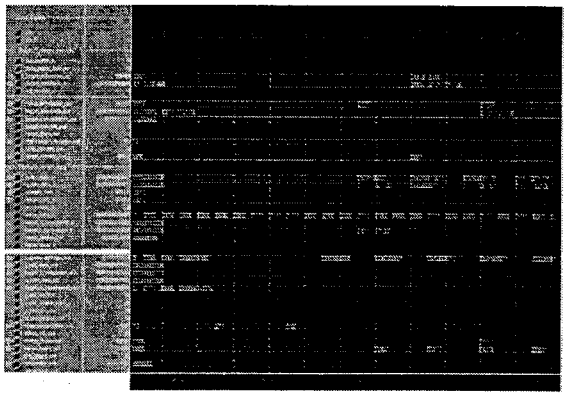


Figure 7: Simulation

4.2. ModelSim

ModelSim is a program in which a VHDL-Code can be simulated. ModelSim can either be started when simulating a design in Matlab/Simulink or when Synthesizing and Implementing the design in ISE (see figure 8). ISE offers four different types of simulating a design: Behavioural Simulation, Post Translate Simulation, Post Map Simulation, Post Place and Route Simulation.

Before synthesizing, the actual design can be simulated. This Behavioral simulation is typically used to check the syntax of the code. In order to find errors in the design, after synthesizing, this simulation can be started (post translate). At Post Map simulation, the user gets a first impression of the design of how it would behave on the FPGA or CPLD. This simulation pays attention to the delay-times of each separate block of the design. The Post Place and Route simulation represents nearly the same behaviour of the design on how it would work on the FPGA or CPLD. In difference to the Post Map Simulation, the signal delay-times between the separate blocks are considered. (see figure 8).

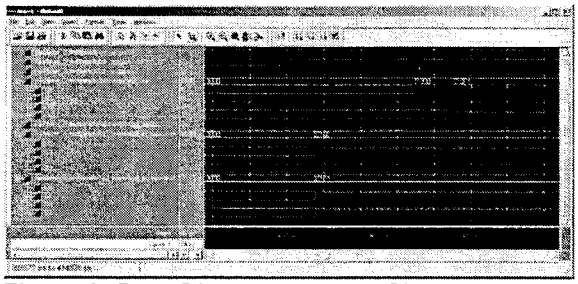


Figure 8: Post-Place and Route-Simulation

5. Results

Table 1 shows the implementation results of the RES. Slices are logic elements of a Xilinx FPGA. IOB's are the interfaces between the FPGA and his environment (see [5] Xilinx datasheet device: XC2VP4-ff672).

*component within the FPGA**	*resources*	*in %*
Number of Slices	1681 out of 3008	55
Number of Slice Flip Flops	1838 out of 6016	30
Number of 4 input LUTs	2540 out of 6016	42
Number of bonded IOBs	101 out of 348	29
Number of BRAMs	18 out of 28	64
Number of MULT18X18s	5 out of 28	17
Number of GCLKs	1 out of 16	6

*Xilinx device: XC2VP4-ff672

Table 1: Resources

In practice, to use the design flow as it was described here, is a very comfortable way to create a design in High-Level. You can concentrate on developing your design. It's not necessary to have depth knowledge of a hardware layout because there are already pre-defined and optimized blocks within the Xilinx Library.

Especially useful are the „BlackBoxes" blocks from the Xilinx Library. Programmers who have until now written their design in hardware description languages can simply implement their design without generating the same design from the beginning.

Another positive aspect of this design flow is the simple simulation of the design in every step – from the abstract-level to the hardware-level. For hardware-simulation in ISE, System Generator also allows to generate a testbench automatically.

It's not always useful to develop a design graphically. To use the „BlackBox"-Block from the Xilinx-Library to implement a VHDL-Code and therefore have to know some more depth knowledge of the hardware can sometimes be much more helpful. In fact, some graphically solved modules of a design can get a huge size and use more resources on an FPGA or CPLD.
Another aspect which must be considered when using this design flow is, that there can only be FPGA's and CPLD's from Xilinx be programmed.

6. Summary and Outlook

The presented design flow from UML the FPGAs via Simulink and Xilinx ISE offers a good approach for rapid prototyping. The advantages for a system engineer are: system level design with less hardware experiences, vendor IPs with good performance, playing with word lengths, fast simulations and VHDL Code generation. Additionally, UML provides a platform independent logical view to the system by use of an open and standardized notification.

The implement design is vendor dependent. Changing the vendor means redesigning (redrawing) the complete design. Synplicity offers a solution for this problem with the two programs Synplify DSP and Synplify Pro [6].
Our next steps are using the Xilinx Embedded Development Kit (EDK) for software support. This kit includes the MicroBlaze processor core and peripheral IP that supports PowerPC and MicroBlaze. The GNU-based development tools consists of C compiler, assembler, linker and debugger. Such a processor can be used as user interface for parameter configuration.

7. References

[1] A.Ludloff, *Praxiswissen Radar und Radar-signalverarbeitung*, Vieweg, 1998.

[2] Unified Modelling Language: www.uml.org.

[3] R.Gessler, *Design of a FPGA Radar Signal Processor*, MathWorks DSP Conference 2003, Stuttgart, Mai 2003.

[4] R.Gessler, T.Mahr, M.Wörz, *Modern Hardware-Software Co-design for Radar Signal Processing*, ISSSE 2004, Linz, August 2004.

[5] Xilinx Homepage: www.xilinx.com.

[6] Synplicity Homepage: www.synplicity.com.