Design flow zur Entwicklung Geschwindigkeits- und Leistungsoptimierter Schaltungen

Frank Sill, Frank Grassert, Dirk Timmermann

Universität Rostock
Institut für Angewandte Mikroelektronik und Datentechnik
Richard-Wagner-Str. 31, 18119 Rostock-Warnemünde
{frank.sill; frank.grassert; dirk.timmermann}@etechnik.uni-rostock.de

Zusammenfassung

Um den kontinuierlich steigenden Anforderungen an anwenderspezifischen Schaltkreisen gerecht zu werden, ist es möglich, auf Transistorebene neue Schaltungstechniken einzuführen. An der Universität Rostock wurde die Schaltungstechnik *Asynchronous Chain True Single Phase Clock* (AC-TSPC) entwickelt. Hierbei handelt es sich um dynamische Gatter innerhalb selbstgetakteter Ketten, welche in ein Einphasentaktsystem eingebunden sind. Da die Synthese dynamischer Logiken und asynchroner Schaltungen mit kommerziellen Programmen nur ungenügend unterstützt wird, wurden ein Design flow und die dazugehörigen Programme entwickelt. Dieser Design flow ermöglicht die Synthese einer Schaltung auf AC-TSPC-Logik. Wir zeigen die Grundlagen der AC-TSPC Schaltungstechnik und werden den entwickelten Design flow vorstellen.

1. Einführung

Zur Verarbeitung von Datenströmen hat sich die Verwendung von Pipelines als ein sehr effektives Mittel erwiesen. Eine Pipeline[1] besteht aus einzelnen Pipelinestufen. Der Ausgang einer Pipelinestufe ist dabei jeweils mit dem Eingang einer folgenden Stufe verbunden. In Jedem Takt verarbeitet eine Stufe ihre Eingangsmuster und legt die Ergebnisse an ihren Ausgang. Das Muster wird mit jedem neuen Taktimpuls von den jeweils nächsten Stufen verarbeitet.
An der Universität Rostock wurde das Programm DYNAMIC[1] entwickelt. Dieses ermöglicht die Synthese und Optimierung von Pipeline-Strukturen.
Zur Umsetzung der Pipelines werden dynamische Schaltungstechniken eingesetzt, da mit ihnen die schnellsten Gatter realisiert werden können. Die logische Funktion wird dabei nur durch schnelle N-Kanal-Transistoren aufgebaut.
Da jedes Gatter mit dem Taktsignal verbunden wird, ist die kapazitive Belastung sehr hoch. Das führt zu einem höheren Leistungsverbrauch, einer aufwendigen Entwicklung eines Taktbaumes und zu hohen zeitlichen Verschiebungen des Taktsignals. An der Universität Rostock wurde daher die Schaltungstechnik AC-TSPC (*Asynchronous Chain – True Single Phase Clock*) entworfen [2]. Hierbei erfolgt eine asynchrone Selbsttaktung der Gatter. Dabei wird für jedes Gatter ein Zustandssignal generiert, welches den aktuellen Zustand des Gatters angibt. Dieses Zustandssignal wird mit den Steuereingängen der vorhergehenden Gatter verknüpft. Ein Gatter berechnet erst dann neue Werte, wenn das nachfolgende Gatter seine Berechnungen beendet hat. Um den Synchronismus innerhalb der Schaltung und zum Taktsignal zu erhalten, sind die Gatter zu Ketten zusammengefasst, in denen das letzte Gatter jeder Kette mit dem Taktsignal verbunden wird. AC-TSPC ermöglicht den Aufbau latchfreier Strukturen, welche einen höheren

[1] Pipeline (engl.) = Verarbeitungskette

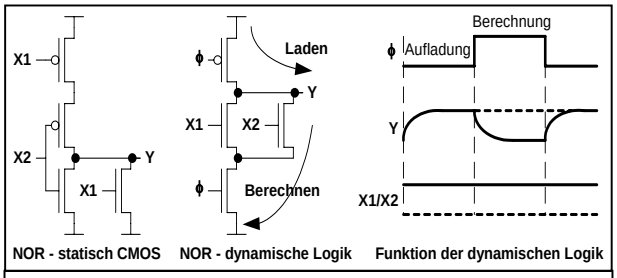

Abbildung 1 **NOR Gatter in Standard-CMOS und dynamischer Logik; Funktionsweise der dynamischen Logik**

Durchsatz zulassen. AC-TSPC Schaltungen besitzen somit ein niedriges *Power-Delay*-Produkt, welches den Energieverbrauch einer Schaltung angibt.

Ein Problem dieser Schaltungstechnik besteht darin, dass sie von heutigen Programmen zum Chipdesign nicht unterstützt wird. Daher war es notwendig, einen eigenen Design flow zu entwickeln, und die notwenigen Programme zu schreiben.

In Abschnitt 2 werden die Grundlagen selbstgetakteter Strukturen erläutert. Der entwickelte Design flow wird in Abschnitt 3 beschrieben. In Abschnitt 4 stellen wir einige Ergebnisse vor und in Abschnitt 5 erfolgt eine Zusammenfassung.

2. Selbstgetaktete Strukturen

2.1 dynamische Schaltungen

Die Abbildung 1 zeigt den Aufbau einer dynamischen Schaltung im Vergleich zur Standard-CMOS-Schaltungstechnik (SCMOS). Für die Realisierung der logischen Funktion wird im Gegensatz zur statischen CMOS Technik, bei der die Funktion durch N-Transistoren und durch komplementär verschaltete P-Transistoren aufgebaut wird, nur ein Netzwerk aus N-Transistoren benötigt, wodurch sich ein Geschwindigkeitsvorteil ergibt. Allerdings arbeiten dynamische Logiken grundsätzlich in zwei Phasen und erfordern somit ein Taktsignal. In der ersten Phase (Ladephase) wird der Ausgang auf einen vordefinierten Wert geladen. In der zweiten Phase (Berechnungsphase) erfolgt die Berechnung, wobei der Ausgangsknoten je nach Eingangszustand durch das Netzwerk in den alternativen Zustand umgeladen wird. Hauptnachteil dieser Struktur ist, dass der Ausgang nicht direkt als Eingang folgender dynamischer Stufen genutzt werden kann.

2.2 AC-TSPC Schaltungstechnik

Dynamische Schaltungstechniken mit komplementären Ausgängen, so genannte Dual-Rail-Techniken, können als asynchrone Logiken aufgebaut werden. Durch einen Wechsel zwischen der Ladephase, in der beide Ausgänge auf denselben, vordefinierten Zustand geladen werden, und der Berechnungsphase, in der die Ausgänge nach Beendigung der Berechnung komplementär zueinander sind, können s*elf-timed*-Signale erzeugt werden. Ist die Berechnung einer Stufe abgeschlossen, so kann ein erzeugtes *Ready*-Signal die vorhergehende Stufe in die Ladephase versetzen, da die Eingänge nicht mehr benötigt werden. Es ist ebenfalls möglich, eine nachfolgende Stufe in die Berechnungsphase zu setzen, wenn die Eingangsdaten bereits oder in naher Zukunft anliegen. Allerdings kann

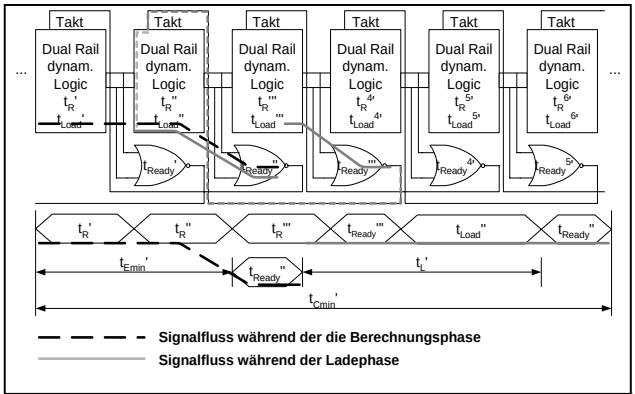

Abbildung 2 **Asynchrone Verschaltung dynamischer Logiken und Zeitdiagramm für den Ablauf einer Berechnungs- und Ladephase für die erste Logikstufe**

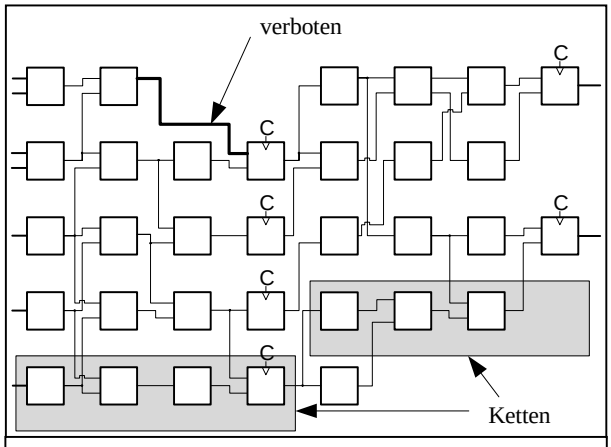

Abbildung 3 **Pipeline Struktur, welche aus Ketten der Länge 4 besteht. Die Ready-Signale sind nicht mit eingezeichnet. Die Eingangssignale, die mit dem globalen Takt verknüpft sind, sind mit einem C gekennzeichnet.**

bei dieser Variante die Berechnungszeit verzögert werden, wenn die Taktsignale nicht rechtzeitig eintreffen. Der allgemeine Vorteil derartiger asynchroner Verknüpfungen ist, dass eine minimale Berechnungszeit dieser Kette erreicht werden kann, wie z.B. von Williams und Horowitz in [3] gezeigt wurde.

Abbildung 2 zeigt eine solche Verschaltung und ein Zeitdiagramm, von dem die Funktionsbedingungen abgeleitet werden können. Es ist die minimal erforderliche Zykluszeit bestehend aus Berechnungs- und der Ladephase für die erste Stufe dargestellt. Der Nachteil solcher asynchroner Ketten ist, dass für ein leichteres Design eine Eintaktung in ein bestehendes synchrones System erfolgen muss. Eine solche Integration wurde von uns in [2] vorgestellt. Dabei wird jeweils die letzte Stufe einer Kette von asynchron verknüpften Stufen mit dem globalen Takt betrieben. In Abbildung 3 ist die resultierende Kettenstruktur abgebildet. Die Gatter besitzen zur vereinfachten Darstellung nur einen Ausgang. Es handelt sich jedoch um Dual-Rail-Gatter mit zusätzlicher Generierung eines Zustandssignals.

2.3 Probleme bei der Generierung der self-timed Signale

Sind mehrere Gatter miteinander verbunden, ist die Erzeugung der Steuersignale nicht mehr trivial. Um eine korrekte Funktionsweise der Gatter zu garantieren, müssen die Steuersignale innerhalb einer festgelegten Zeitspanne eintreffen [2]. Sind mehrere Gatter untereinander verbunden, müssen für alle einzelnen Gatter die Bedingungen für die Steuersignale beachtet werden. In Abbildung 4 sind die zwei Hauptfälle abgebildet. Im ersten Fall (Abbildung 4, links) wird ein Zustandssignal verwendet um zwei oder mehrere Gatter zu takten. Dies beschränkt das Zeitverhalten der getakteten Gatter, denn diese müssen in jedem Zustand annähernd parallel arbeiten. Im zweiten Fall sind mehrere Gatter mit dem Ausgangssignal eines Gatters verbunden (Abbildung 3, rechts). In diesem Fall existieren mehrere Zustandssignale, welche mit dem Takteingang verbunden werden können. Diese Signale müssen mit einer zusätzlichen Logik kombiniert werden, z.B. mit einem OR-Gatter. Diese Logik muss sicherstellen, dass das Gatter, welches das Ausgangssignal generiert, nicht in die Ladephase wechselt, bevor alle folgenden Gatter ihre Berechnung beendet und ihr Zustandssignal generiert haben. Dies ist erforderlich, da garantiert werden muss, dass die Eingangssignale lange genug während der Berech-

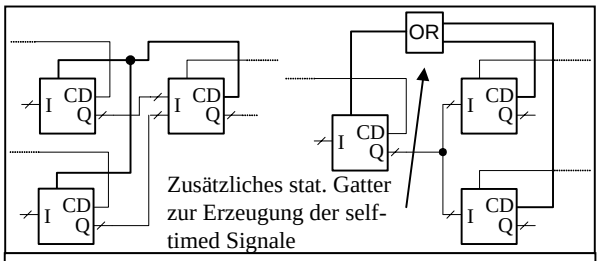

Abbildung 4 **Probleme bei der Generierung der self-timed-Signale bei mehrfach verknüpften Gattern; die Blöcke representieren ein einzelnes dynamisches Dual-Rail-Gatter aus Abbildung 2.**

nung anliegen, um eine korrekte Berechnung zu ermöglichen. Weiterhin ist eine Berechnung nur möglich, wenn die Gatter ihre Ladephase vollständig beendet haben. Das bedeutet, ein Gatter darf nicht zu früh in die Berechnungs- bzw. Ladephase versetzt werden.

Innerhalb eines großen Designs können Fälle auftreten, in denen es nicht möglich ist, die Einhaltung dieser Bedingungen für alle Gatter zu garantieren. Das macht es erforderlich, Änderungen an verschiedenen Parametern, wie z.B. der Länge der Ketten, vorzunehmen.

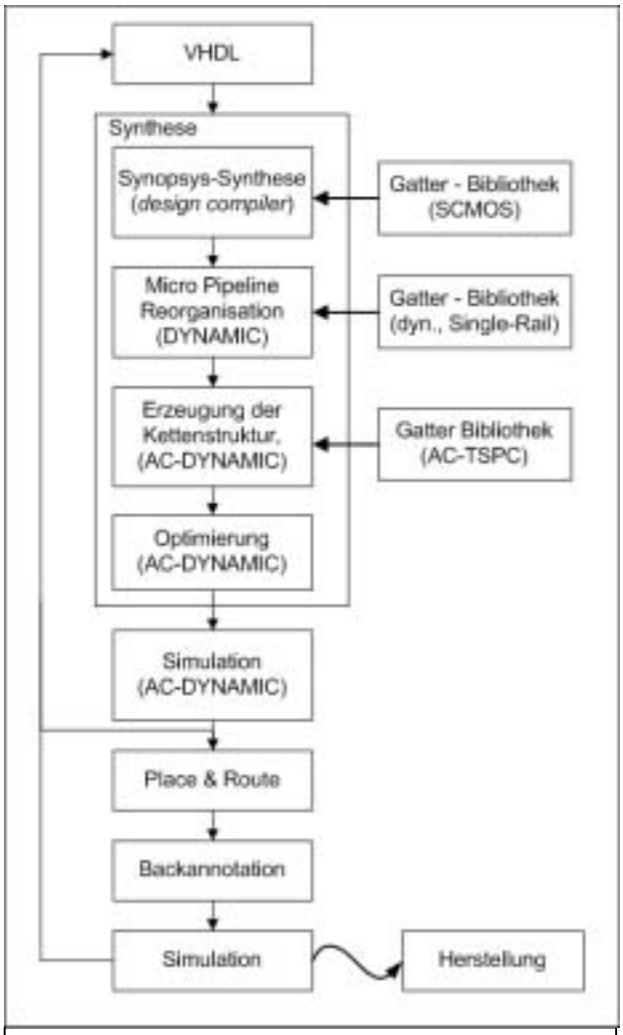

Abbildung 5 **Design flow zur Erzeugung von AC-TSPC Schaltungen**

3. Design flow

Der entwickelte Design flow ist in Abbildung 5 dargestellt. Er gliedert sich in die Synthese, Simulation und den Backend-Bereich. Dabei ist die Synthese unterteilt in die Synthese auf eine SCMOS-Gatter-Bibliothek, die Generierung der Pipeline, die Generierung der AC-TSPC-Struktur und die Optimierung der AC-TSPC-Struktur.

3.1 Entwicklung der Bibliothek

Es wurden alle notwendigen Gatter und Funktionsblöcke in einer neuen Synopsys [4] Synthese Bibliothek zusammengefasst. Um die Bibliothek, welche die Struktur und die logische Funktion der Gatter enthält und aus statischen Gattern besteht, zu erzeugen, wurde der *library compiler* von Synopsys benutzt.

Für jedes statische Gatter wurde ein funktional äquivalentes dynamisches Single-Rail-Gatter entworfen und in eine weitere Synthese Bibliothek hinzugefügt.

In einem späteren Schritt werden Dual-Rail-Gatter mit zusätzlicher Generierung eines Zustandssignals benötigt. Für diese Gatter, musste ebenfalls das Layout und eine Spice-Beschreibung erstellt werden. Weiterhin wurden für die Gatter alle markanten Zeitwerte aufgenommen. Dazu gehören die minimalen und maximalen Berechnungszeiten und Aufladungszeiten sowie die Generierungszeiten des Zustandsignals.

3.2 Synopsys-Synthese

Der Design flow beginnt mit der Schaltungsbeschreibung, welche in VHDL erfolgt. Die Schaltung wird mit dem *design compiler* von Synopsys auf die erstellte SCMOS-Gatter Bibliothek kompiliert. Als Ergebnis liegt eine Netzliste aus kombinatorischen Gattern vor, die keine Taktsignale enthält.

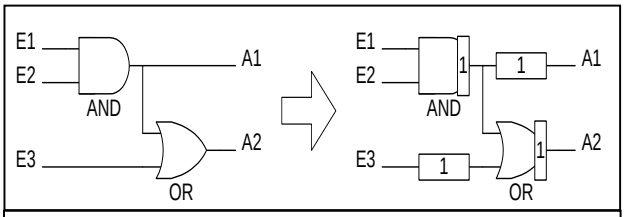

Abbildung 6 **Beispiel für die Umwandlung kombinatorischer Logik in eine Pipeline**

3.3 *Micro Pipeline Reorganisation* (MPR)

Die erhaltene Netzliste wird in das Programm DYNAMIC [1] eingelesen. Es erfolgt die *Micro Pipeline Reorganisation* (MPR), bei der die Schaltung als Pipeline umorganisiert wird. Die kombinatorischen Gatter werden durch äquivalente dynamische Single-Rail-Gatter ersetzt, die über eine Registerfunktionalität verfügen. Dadurch kann jedes Gatter als Pipelinestufe behandelt werden. Um das korrekte zeitliche Verhalten der Schaltung abzusichern, werden in einzelnen Leitungen Register hinzugefügt (siehe Abbildung 6). Dies ist notwendig, da jedes Signal pro Takt eine Registerstufe durchläuft. Es ist zu beachten, dass Schleifenkonstrukte nicht automatisch synthetisiert werden können und per Hand editiert werden müssen. Als Ergebnis liegt die Netzliste einer als Pipeline aufgebauten Schaltung vor, die aus dynamischen Single-Rail-Gattern besteht und ein Taktsignal enthält.

3.4 Erzeugung der Kettenstruktur

Die Netzliste wird in das Programm AC-DYNAMIC [5] eingelesen. Die Single-Rail-Gatter werden durch funktional äquivalente Dual-Rail-Gatter ersetzt, welche aus der erstellten Bibliothek stammen. Jedes Gatter verfügt über eine Zustandssignalgenerierung. Es werden die Takteingänge der Gatter mit den Zustandssignalen der nachfolgenden Gatter verbunden. Im nächsten Schritt werden in regelmäßigen Abständen die Takteingänge der Gatter nicht mit einem Zustandssignal sondern mit einem globalen Taktsignal verbunden (siehe Abbildung 3). Daraus ergibt sich die Länge der Ketten. Für alle Gatter werden zusätzlich die Zeitparameter eingelesen. Es ist möglich, diese Parameter automatisiert oder per Hand einzugeben. Weiterhin müssen zusätzliche Logiken zur Generierung der *self-timed* Signale eingebunden werden, die die Einhaltung der Zeitparameter der Gatter garantieren (siehe Abschnitt 2.3).

3.5 Optimierung

Es existieren Schaltungsparameter, die variiert werden können. Dazu gehören die Länge der Ketten, die Symmetrie des Taktsignals, die Frequenz des Taktsignals, die Wahl der zusätzlichen Logik zur Generierung der *self-timed* Signale sowie die Zeitparameter einzelner Gatter. Während der Optimierung wird geprüft, welche Parameter eine optimale Lösung für die entsprechende Schaltung darstellen [5]. Die Kriterien für eine optimale Lösung sind eine hohe Taktfrequenz, ein hoher Durchsatz und eine möglichst geringe Zahl der Logiken zur Generierung der *self-timed* Signale. Die Optimierung beginnt mit der Berechnung der maximalen Taktfrequenz für jede mögliche Kettenlänge. Als Grundlage für diese Berechnung dienen die zeitlichen Anforderungen an jedes Gatter. Während dieser Berechungen werden die zusätzlichen Logiken zur Generierung der *self-timed*-Signale überprüft und gegebenenfalls ersetzt. Im nächsten Schritt wird für jede Kettenlänge geprüft, welche dieser Logiken entfallen können, bzw. zusammengefasst werden können. In einem weiteren Schritt wird geprüft, inwieweit Ketten, die nur aus Buffern bestehen, zu einem einzelnen Buffer zusammengesetzt werden können. Weiterhin wird ermittelt, wie weit eine Verzögerung einzelner Zustandssignale die Taktfrequenz erhöht.

3.6 Simulation

Das Programm AC-DYNAMIC simuliert die

	SCMOS	TSPC	AC-TSPC
Latency (ns)	15.6	31.2	7.8
Power (mW/GHz)	277	600	1361
C-load (min. Trans.)	~380	~2205	~370
Pow.*Del. (10^{-9} Ws)	3.322	9.610	5.460

Tabelle 1 **Vergleich der Ergebnisse für einen 4x4Multiplizierer in verschiedenen Schaltungstechniken.**

Informationsverarbeitung innerhalb der Kettenstruktur. Es wird getestet, ob die zeitlichen Bedingungen für die Steuersignale jedes Gatter eingehalten werden, bzw. wo Fehler auftreten. Für jedes Gatter können die einzelnen Zeitpunkte und Parameter ausgegeben werden. Zusätzlich kann für verschiedene Taktfrequenzen die Funktionsfähigkeit der Schaltung ermittelt werden. Ist die Simulation erfolgreich beendet worden, liegt als Ergebnis eine Netzliste einer AC-TSPC Schaltung vor.

3.7 Backend

Die erhaltene Netzliste wird in den Backend-Prozess übernommen. Hierzu gehören das *Floorplaning*, das Platzieren der Gatter und Routing (*Place&Route*) sowie die Backannotation.

4. Ergebnisse

In Tabelle 1 sind einige markante Werte für die Realisierung eines 4x4-Multiplizieres in verschiedenen Schaltungstechniken dargestellt. Die Ergebnisse stammen aus den *HSpice*-Simulationen. Es ist zu beachten, dass für die Standard CMOS (SCMOS) Implementierung Gatter aus einer optimierten Bibliothek verwendet wurden.

5. Zusammenfassung

Dynamische Schaltungstechniken lassen die Verwendung hoher Taktfrequenzen zu. Selbstgetaktete Schaltungstechniken ermöglichen den Aufbau latchfreier Strukturen, die den Durchsatz erhöhen. *Asynchronous Chain True Single Phase Clock* (AC-TSPC) – Logik verbindet latchfreie Berechnungen mit dynamischer Logik und gestatten somit einen niedrigen Energieverbrauch. Wir haben einen Design flow für AC-TSPC-Logik vorgestellt, der vollständig unterstützt wird. Die dafür entworfene Software ermöglicht eine genaue Analyse des Schaltverhaltens und eine Optimierungen der Schaltung nach verschiedenen Kriterien.

Literaturangabe

[1] A. Wassatsch, *DYNAMIC – A Java Based Toolset For Integrating Dynamic Logic Circuits Into A Standard VLSI Design Flow*, International Cadence User Group Conference ICU, San Jose, September 2000, SIG IC - ic6.

[2] F. Grassert und D. Timmermann, *Dynamic Single Phase Logic with Selftimed Stages for Power Reduction in Pipeline Circuit Designs*, ISCAS, 2001

[3] T. E. Williams und M. A. Horowitz, *A Zero-Overhead Self-Timed 160-ns 54-b CMOS Divider*, IEEE Journal of Solid-State Circuits, Vol. 26, No. 11, November 1991

[4] Synopsys, Inc., 700 East Middlefield Road, CA 94043-4022 United States of America, *Synopsys Synthesis and Simulation Tools*, 2002 edition

[5] F. Sill, *Optimierungen an selbstgetakteten Einphasentaktschaltungen*, Diplomarbeit, Universität Rostock, Oktober 2002