Algorithmen und Architekturen für MSB-First-Signalverarbeitungssysteme

Steffen Dolling, Andreas Wassatsch, Hagen Ploog, Dirk Timmermann

1. Einleitung

Signalverarbeitungs-Anwendungen mit einem großen Rechen- und Kommunikationsbedarf werden vielfach mit arithmetischen Strukturen realisiert, die die Eingabe der Operanden und die Ausgabe des Ergebnisses in serieller Form gestatten. Dabei werden sehr häufig Rechenelemente mit zeichenserieller, least-significant-digit-first Arithmetik (LSDF) [1] eingesetzt. Neben Vorteilen, wie z.B. leichte Pipelinebarkeit, ist diese gebräuchlichste Form der Serialisierung aber auch mit gravierenden Nachteilen verbunden. So ist eine Überlappung intern entscheidungsbasierter Funktionen (Division, Quadratwurzel, etc.) nicht möglich, da eine Berechnung im Empfänger erst starten kann, wenn alle Operandenbits empfangen wurden. Entsprechend steigt die Latenzzeit des Verfahrens ganz erheblich an.

Eine effiziente Alternative zu den seriellen Kommunikationsformen im LSDF-Format bilden Digit-On-Line (DOL)-Verfahren, die die Eingabe der Operanden und die Ausgabe des Ergebnisses im most-significant-digit-first-Format (MSDF) gestatten. Die folgenden Ausführungen befassen sich mit den Eigenschaften und den grundlegenden Merkmalen der Algorithmenentwicklung. Ebenso werden Architekturen für Festkomma- und Gleitkommarealisierungen sowie eine Hardwarestruktur zur Kaskadierung einzelner DOL-Module dargestellt.

2. Eigenschaften von Digit-On-Line-Algorithmen

Die Digit-On-Line-Arithmetik basiert auf einer redundanten Zahlendarstellung. Sie ist dadurch gekennzeichnet, daß jede Stelle einer redundanten Zahl mit der Basis r mehr als r verschiedene ganzzahlige Werte annehmen kann. Die Bearbeitung der Operanden und Ergebnisse erfolgt nach dem MSDF-Prinzip. Dabei benötigen die On-Line-Verfahren δ anliegende Digits des Eingangsoperanden, bevor die erste Ziffer des Ergebnisses generiert werden kann. Das On-Line-Delay δ ist dabei lediglich von der Operation und der Zahlenbasis r, aber nicht von der Operandenlänge n abhängig. Dabei gilt allgemein $\delta << n$. In Bild 1 ist das Zeitverhalten einer Funktion $c = f(x, y)$ dargestellt. Dabei wird von einer Festkommadarstellung der Operanden ausgegangen, die für einen Wert X folgender Notation genügt:

$$X_j = X_{j-1} + x_{j+\delta}\, r^{-(j+\delta)} \quad mit \quad X_0 = \sum_{i=1}^{\delta} x_i\, r^{-i} \tag{1}$$

Mit der Taktperiode t_D , die die Zeitspanne zur Berechnung des nächsten Resultatszeichens kennzeichnet, ergibt sich damit folgende Gesamtzeit zur Berechnung eines Ergebnisses mit der Operandenlänge n:

$$T_{Gesamt} = (n + \delta) \cdot t_D \tag{2}$$

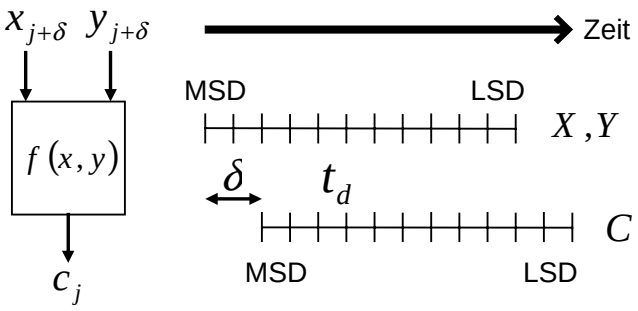

Bild 1: Zeitverhalten einer DOL-Funktion *f*

Entsprechend sind die DOL-Verfahren mit folgenden Vorteilen verbunden. Da nicht alle Zeichen der Eingangsoperanden gleichzeitig angelegt werden müssen, ist eine Kommunikation mit niedriger Bandbreite zwischen einzelnen DOL-Modulen möglich. Die Taktperiode t_D ist lediglich von der internen Realisierung der arithmetischen Operation, aber nicht von der Operandenlänge *n* abhängig. Da die Resultatzeichen schon während der eingangsseitigen Einspeisung der Operandendigits erzeugt werden, ist eine hohe Überlappung von aufeinanderfolgenden und möglicherweise voneinander abhängigen Operationen erreichbar.

3. Algorithmenentwicklung und Festkomma-Architekturen

DOL-Algorithmen setzen sich allgemein aus einer Rekursionsgleichung zur Bestimmung eines Restvektors W_j und einer Selektionsfunktion zur Berechnung des Ausgabedigits c_j zusammen:

$$W_j \leftarrow f\left(W_{j-1}, X_{j-1}, x_{j+\delta}, Y_{j-1}, y_{j+\delta}, c_{j-1}\right) \tag{3}$$

$$c_j \leftarrow S(W_j) \tag{4}$$

Für eine Implementierung müssen die genannten Gleichungen auf VLSI-gerechte Grundelemente abgebildet werden. Die Selektionsfunktion wird dabei durch ein Auswahlverfahren bestimmt, bei dem eine Auswertung von Betrag und Vorzeichen des Rekursionsvektors W_j erfolgt. Die zentralen Elemente der Rekursion sind Signed-Digit (SD) – Addierer [2], Shifter für Digitvektoren und Zellen, die die Multiplikation zweier Digits mit Hilfe eines Digit-Negators und eines Multiplexers realisieren. Bei verschiedenen DOL-Modulen werden auch Truncation-Elemente zur Korrektur von Pseudoüberläufen notwendig.

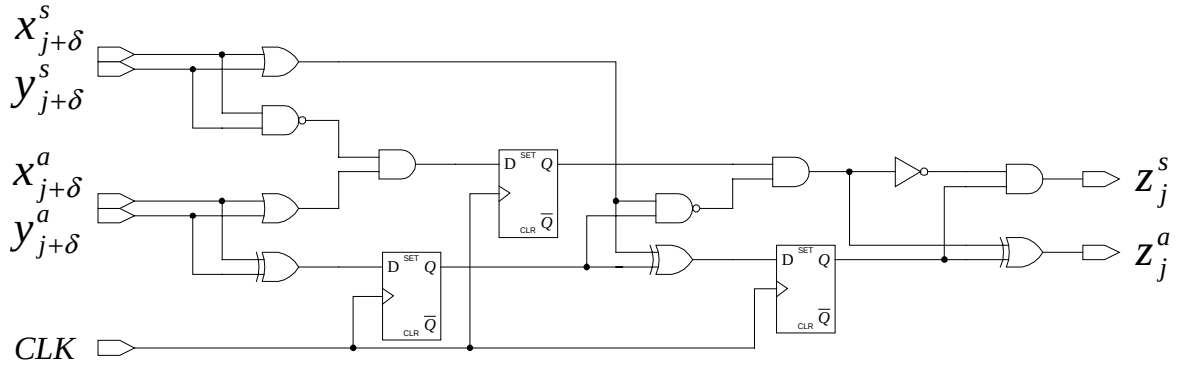

Bild 2: DOL-Addierer für zwei Operanden mit der Basis 2

Eine hinsichtlich der Fläche und der Geschwindigkeit effiziente Realisierung von DOL-Modulen ist entsprechend Gleichung (2) mit der Erfüllung folgender Zielstellungen verbunden. Zum einen muß bei der Algorithmenentwicklung ein minimales On-Line-Delay δ angestrebt

werden. Andererseits muß beim Architekturentwurf durch eine maximale Parallelisierung der internen DOL-Berechnungseinheiten eine minimale Taktperiode t_D erreicht werden.

Modul	δ
Addition mit zwei Operanden	2
Addition mit vier Operanden	3
Addition mit acht Operanden	4
Addition mit skalierten Summanden	3
Multiplikation	3
Potenz zweiten Grades	3
Multiplikation mit festem Faktor	3
Division	4
Quadratwurzel	2
Lineare Funktion	4

Tabelle 1: DOL-Module

Eine Möglichkeit für das Design der DOL-Einheiten basiert auf einer geschlossenen und systematischen Methodik der Algorithmenentwicklung [3]. Ausgehend von den Fehler- und Konvergenzkriterien werden hier die Iterationsgleichungen und die Initialisierungsbedingungen hergeleitet. In einem weiteren Schritt sind diese auf die oben genannten Grundelemente abzubilden. Eine weitere Möglichkeit ist die Ableitung elementarer DOL-Architekturen (Addition/Subtraktion, Inkrement, Multiplikation, lineare Funktion) aus den parallelen Algorithmen. So können z.B. durch Einfügen von sequentiellen Elementen in redundante Addiererstrukturen DOL-Operatoren für die Addition von zwei Operanden realisiert werden (siehe Bild 2). Mit diesem Verfahren sind kleinere DOL-Zellen ohne interne Rekursion bei gleichem On-Line-Delay δ realisierbar. Damit sind hier kompaktere Signalpfade mit kürzerem kritischen Pfad und entsprechend höhere Taktfrequenzen möglich.

Auf der Basis der genannten Entwicklungsmethoden wurden die in Tabelle 1 aufgeführten DOL-Module zur Realisierung elementarer Funktionen entwickelt. Diese als VHDL-Beschreibungen vorliegenden komplexen Anordnungen sind hinsichtlich der Operandenlänge parametrierbar und in einem entsprechenden VLSI-Zyklus an die geforderten Zielvorgaben anpaßbar.

4. Kaskadierung und Gleitkomma-Architekturen

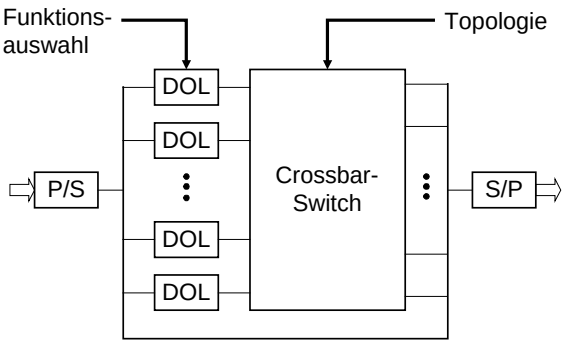

Bild 3: Rekonfigurierbare Architektur

Eine Kaskadierung von aufeinanderfolgenden und auch funktionell voneinander abhängigen Operationen ist durch das in Abbildung 3 dargestellte flexible und rekonfigurierbare Architekturkonzept möglich. Das zentrale Element dieser Anordnung ist ein Crossbar-Switch mit der Kapazität von 160 E/A-Anschlüssen. Um dieses sind acht FPGA's der Firma XILINX gruppiert. In diesen Elementen können ein oder mehrere verkettete DOL-Grundfunktionen implementiert werden. Die Funktionsauswahl erfolgt dabei über die Programmierbarkeit der XILINX-Bausteine, während die Topologie über den Crossbar-Switch eingestellt werden kann.

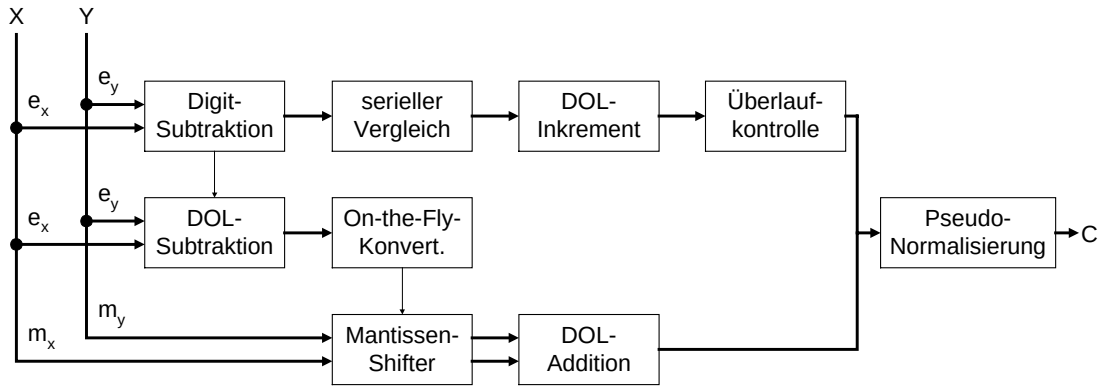

Bild 4: Gleitkomma-DOL-Addierer

Bei einer starken Kaskadierung führt eine Festkommadarstellung der DOL-Operatoren zu Ungenauigkeiten bzw. Fehlern der Ergebnisvektoren. Dieses Problem kann durch pseudo-normalisierte Eingangsoperanden mit einem Wertebereich von [¼, 1[oder]-1, -¼] beseitigt werden. Entsprechend ist eine Erweiterung der Algorithmen und Architekturen auf eine Gleitkommadarstellung notwendig. Dabei werden die Exponenten den jeweiligen Mantissen vorangestellt und ebenfalls nach dem MSDF-Prinzip übertragen. Bild 4 zeigt den Aufbau eines DOL-Addierers. Dabei wird durch eine Digit-Subtraktion und einen seriellen Vergleich der größere Exponent ermittelt und anschließend inkrementiert. Parallel dazu wird über eine DOL-Subtraktion und eine On-the-Fly-Konvertierung der Differenzbetrag der beiden Exponenten berechnet. Danach erfolgt die DOL-Addition der entsprechend verschobenen Mantissen. Das Delay δ der Anordnung beträgt zu diesem Zeitpunkt 2. Das δ der anschließenden Pseudo-Normalisierung ist abhängig vom Additionsergebnis und entsprechend variabel.

5. Zusammenfassung

Die dargestellten DOL-Module beanspruchen eine minimale Kommunikationsbandbreite und sind durch eine einfache innere Struktur der zugrundeliegenden Operatoren und eine hohe Modularität gekennzeichnet. Eine Parallelisierung zwischen verschiedenen Operationen und damit der Aufbau einer komplexen rekonfigurierbaren DOL-Struktur wird möglich.

Literatur

[1] S.G. Smith, P. Denyer, "Serial-data computation", Kluwer Academic Publishers, 1988
[2] N. Takagi, H. Yasuura, S. Yajima, „High-Speed VLSI Multiplication Algorithmen with a Redundant Binary Addition Tree“, IEEE Trans. on Computers., Vol. 9, S. 789-796, 1985
[3] M.D. Ercegovac, T. Lang, "On-line arithmetic: a design methodology and applications in digital signal processing", VLSI Signal Proc., III, IEEE Press 1988, S. 252-263, 1988

Verfasser

Dipl.-Ing. Steffen Dolling
Dipl.-Ing. Andreas Wassatsch
Dipl.-Ing. Hagen Ploog
Prof. Dr. Dirk Timmermann
Universität Rostock
Fachbereich Elektrotechnik und Informationstechnik
Institut für Angewandte Mikroelektronik und Datentechnik
R.-Wagner-Str. 31, D-18119 Rostock