Dirk Timmermann

VLSI CHIP ZUR EFFIZIENTEN BERECHNUNG ELEMENTARER FUNKTIONEN MIT EINEM SCHNELLEN, HARWAREORIENTIERTEN ALGORITHMUS

Kurzfassung

Es wird eine CMOS Realisierung einer Vektorarithmetikeinheit zur Verarbeitung von Zahlen im IEEE-754 Gleitkommaformat (24 Bit Mantisse, 8 Bit Exponent) vorgestellt. Es stehen u.a. Multiplikation, Division, Sinus, Kosinus, Hyperbelfunktionen, Quadratwurzeln, Logarithmen, Arcustangens, Vektorlänge, Vektorphase und Vektorrotation mit einer normalisierten Durchsatzrate von bis zu 220 MFLOPS zur Verfügung.

1. CORDIC Algorithmus

Anwendungsgebiete für den beschriebenen Chip sind zeitkritische Aufgaben z.B. in der Bildverarbeitung, Mustererkennung, Echtzeitsimulation, Robotik, Kinematik, Signalverarbeitung, Daten- und Bildkompression, Signal- und Spektralanalyse und Telekommunikation. In diesen Gebieten sind häufig die Algorithmen zur Lösung des jeweiligen Problems theoretisch bekannt, jedoch sehr rechenaufwendig. Die mathematische Basis des COordinate Rotation DIgital Computer (CORDIC) Algorithmus bilden iterative Vektorrotationen [1]. Der Iterationsteil der Schaltung implementiert die Iterationsgleichungen

$$\begin{aligned} x_{i+1} &= x_i - m\sigma_i\, 2^{-S(m,i)}\, y_i \\ y_{i+1} &= y_i + \sigma_i\, 2^{-S(m,i)}\, x_i \\ z_{i+1} &= z_i - \sigma_i \alpha_{m,i} \quad , \end{aligned}$$

wobei m das Koordinatensystem festlegt, σ_i die Drehrichtung der Rotation, $S(m,i)$ die Shiftfolge, $\alpha_{m,i}$ den Teildrehwinkel und i den Iterationsindex. Der Parameter m wird zu 1, 0, oder -1 gewählt, wodurch die entsprechenden Vektorrotationen als Drehungen auf einem Kreis, einer Geraden oder einer Hyperbel interpretiert werden können. Die Konvergenz des Verfahrens wird durch die feste Shiftfolge S(m,i) bestimmt, die ihrerseits den Teildrehwinkel für m = 1, 0 bzw. -1 gemäß $\alpha_{m,i} = \arctan(2^{-S(m,i)})$, $\alpha_{m,i} = 2^{-S(m,i)}$ bzw. $\alpha_{m,i} = \operatorname{artanh}(2^{-S(m,i)})$ festlegt.

	$z_n \rightarrow 0$ (Rotation)	$y_n \rightarrow 0$ (Vectoring)
m = -1 hyperbolisch	$x_n = k_{-1}\,(x_0 \cosh(z_0) + y_0 \sinh(z_0))$ $y_n = k_{-1}\,(x_0 \cosh(z_0) + y_0 \sinh(z_0))$	$x_n = k_{-1}\sqrt{x_0^2 - y_0^2}$ $z_n = z_0 + \operatorname{artanh}(y_0 / x_0)$
m = 0 linear	$x_n = x_0$ $y_n = x_0\, z_0 + y_0$	$x_n = x_0$ $z_n = z_0 + y_0 / x_0$
m = 1 zirkular	$x_n = k_1\,(x_0 \cos(z_0) - y_0 \sin(z_0))$ $y_n = k_1\,(y_0 \cos(z_0) + x_0 \sin(z_0))$	$x_n = k_1\sqrt{x_0^2 + y_0^2}$ $z_n = z_0 + \arctan(y_0 / x_0)$

Tabelle I: CORDIC - Funktionen ($k_{-1,0,1} \equiv$ Skalierungsfaktoren)

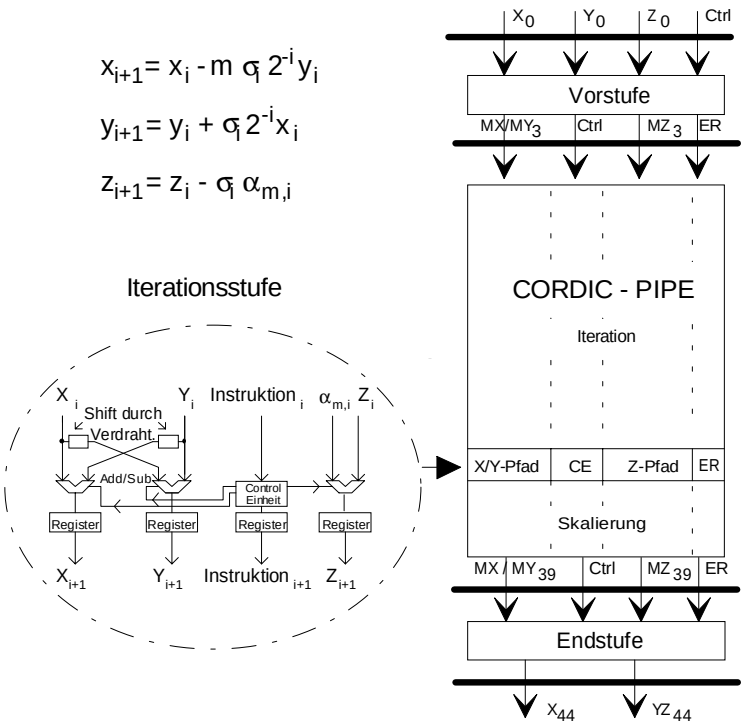

Bild 1: Blockdiagramm des IMSCOR24

Im Verlauf der Iteration wird entweder der Wert für *z* oder *y* gegen Null gezwungen, indem man die Drehrichtung entsprechend $\sigma_i = \text{sign}(z_i)$ oder $\sigma_i = -\text{sign}(x_i)\text{sign}(y_i)$ wählt, wobei $\text{sign}(\beta) = 1$ für $\beta \geq 0$ und $\text{sign}(\beta) = -1$ für $\beta < 0$. Durch Vorgabe des Iterationsziels und des Koordinatensystems kann eines der sechs Funktionspaare in Tabelle I berechnet werden. Die Startwerte der Iteration und damit die Funktionsargumente sind mit x_0, y_0, und z_0 bezeichnet und k_m repräsentiert den sogenannten Skalierungsfaktor, eine unvermeidbare, algorithmusbedingte Verlängerung oder Verkürzung des Vektors während der Iteration. Da dieser Faktor im allgemeinen ungleich Eins ist, muß er nachträglich durch Verdoppelung einiger Iterationen und multiplikative Aufspaltung in möglichst wenig Faktoren (1 ± 2^{-j}) kompensiert werden [2].

2. Funktionaler Überblick

Bild 1 zeigt das Blockschaltbild des Chips. Der Eingangsdatenstrom wird auf die drei Eingangsports *x*, *y* und *z* gegeben, und nach einer Pipelinelatenz von insgesamt 44 Taktzyklen werden die Daten an den Ausgangsports *x* und *yz* ausgegeben (die Bezeichnungen *M* und *E* in Bild 1 stehen für Mantisse und Exponent der Gleitkommazahl). Der Chip erlaubt die Berechnung aller in Tabelle I spezifizierten Funktionen. Beispielsweise wird eine komplette Koordinatentransformation im kartesischen Koordinatensystem mit der Instruktion ROT durchgeführt. Die wichtigsten unterstützten Instruktionen sind in Tabelle II zusammengefaßt.

Instruktion	**Ergebnis**	
	x	**yz**
DIVADD	x	z+y/x
DIVSUB	x	z-y/x
MULADD	x	y+z*x
MULSUB	x	y-z*x
HVECT	$(x^2-y^2)^{0.5}$	z+artanh(y/x)
HVECTM	$(x^2-y^2)^{0.5}$	z-artanh(y/x)
HROT	x*cosh z+y*sinh z	y*cosh z+x*sinh z
HROTM	x*cosh z-y*sinh z	y*cosh z-x*sinh z
VECT	$(x^2+y^2)^{0.5}$	z+arctan(y/x)
VECTM	$(x^2+y^2)^{0.5}$	z-arctan(y/x)
ROT	x*cos z - y*sin z	y*cos z + x*sin z
ROTM	x*cos z + y*sin z	y*cos z - x*sin z
FX....	wie oben, Festkomma	

Tabelle II: Funktionstabelle

Der Chip verwendet intern eine Festkomma-Pipeline, die festverdrahtete Additions-und-Schiebeoperationen enthält. Die Festkomma-Pipeline führt die CORDIC Iterationsgleichungen und die anschließende Skalierungsfaktorkompensation aus. Jede Iterationsstufe implementiert eine Iteration des Algorithmus. Der 29-stufigen Festkomma-Pipeline geht eine Vorstufe voraus, die die Eingangsdaten vom externen

Gleitkommaformat nach IEEE-754 (24 Bit Mantisse, 8 Bit Exponent) in das interne Festkommaformat (32 Bit, siehe Bild 2) umsetzt. Die Berechnung selbst erfolgt mit einer modifzierten Gleitkommaversion des CORDIC Algorithmus, die ein Verschieben der Mantissen und Exponentberechnungen innerhalb der inneren Festkommapipeline vermeidet. Prinzipiell wird dabei nach dem gleichen Schema wie bei Gleitkomma-Addierern vorgegangen: 1) Bestimmung eines gemeinsamen Exponenten (Referenzexponent) für die Argumente, 2) binäres Schieben der Mantissen in Abhängigkeit von der Differenz zwischen dem Exponent des jeweiligen Exponenten und des Referenzexponents, 3) Durchführung der Berechnung mit den geschobenen Mantissen, 4) Normalisierung der Mantissen und entsprechende Berechnung des zugehörigen Exponenten aus dem Referenzexponenten. Im Falle von CORDIC stellt sich dies jedoch komplizierter dar, weil hier prinzipiell mit drei Argumenten gearbeitet wird.
Der Datenfluß wird durch das 6 Bit breite Instruktionswort kontrolliert sowie durch das Iterationsziel und die Wahl des Koordinatensystems. Zusätzlich zum Gleitkommaformat wird auch ein 24 Bit Festkommaformat unterstützt.

3. Design

Die Realisierung der CORDIC Gleichungen bedingt drei Datenpfade. Aufgrund des größeren Ergebnisbereichs der berechneten Funktionen und der Add-and-shift Charakteristik des Algorithmus sind zusätzliche Überlauf und Schutzbits bereitzuhalten. Für das Gleitkomma IEEE-754 Format einfacher Genauigkeit sind zwei interne 32-Bit Pfade (1 Vorzeichen, 3 Überlauf-, 23 Nachkomma- und 5 Schutzbits) für *x* und *y* und ein 29 Bit Datenpfad (1 Vorzeichen, 2 Überlauf-, 23 Nachkomma- und 3 Schutzbits) für *z* erforderlich. Zusammen mit der 29-stufigen Festkomma-Pipeline und den 8 Skalierungsstufen und 10 Registerbits für den Referenzexponenten summiert sich dies zu einen 103 Bit breiten × 37 Stufen tiefen internen Datenpfad, so daß die Chipfläche von zentraler Bedeutung wird.
Zur Durchführung der CORDIC Gleichungen müssen die *x* und *y* Mantissen jeweils geschoben und überkreuz addiert/subtrahiert werden. Die direkte Umsetzung dieses Datenflusses in Silizium würde zu sehr irregulären Leitungskreuzungen und einer Mindestlänge der Verdrahtung horizontal über 32 Bit führen. Aus diesem Grund sind der *x* und *y* Pfad im Layout bitweise nebeneinander plaziert, wodurch sich die Mindestverdrahtungslänge auf horizontal 2 Bit reduziert (jeweils 1 Bit für *x* und *y*).

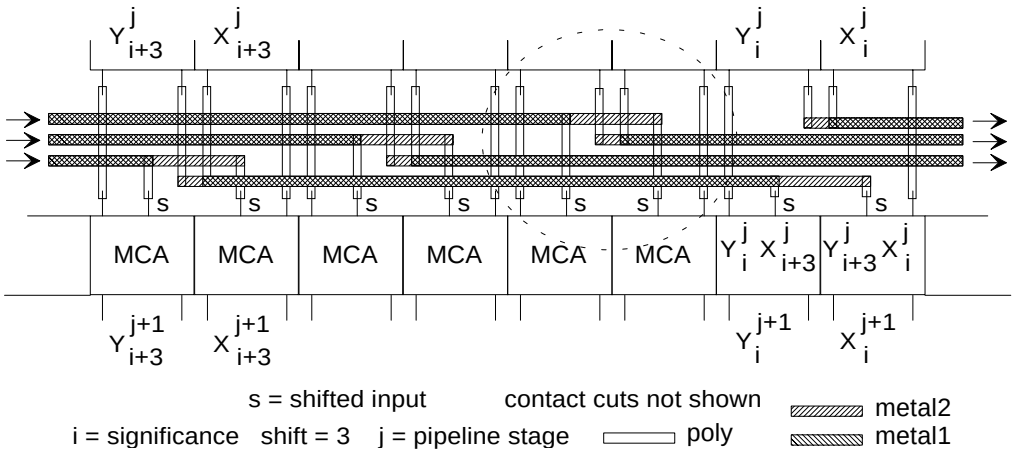

Bild 2: Shift-verdrahtung

Prozeß	2-metal CMOS
Design Regeln	1.6µm
Chipfläche	13.3 x 14.2 mm
Aktive Fläche	8.2 x 13.4 mm
Transistoren	210 000+
Gehäuse	280 Pin PGA
# Pipe. Stufen	(3+37+4)= 44
Max. Takt	10 MHz
Datenformat:	
extern	24b mant., 8b exp. (IEEE-754 single prec.) oder 24b Festkomma
intern	32b mant., 10b exp.
Instruktion	6bit

Tabelle III: Technische Daten

Für das Layout der internen Festkomma-Pipeline wurde ein flächenoptimierender Modulgenerator entwickelt. In Bild 2 ist das Layout- und Verdrahtungsschema des Generators zusammen mit dem bitweise geschachtelten *x* und *y* Pfad und der Shiftverdrahtung dargestellt (MCA steht für Manchester Carry Chain Addierer).

Die gesamte Shiftverdrahtung nimmt dadurch nur 18% der Chipfläche der Festkomma-Pipeline ein. Zur Bestimmung der Drehrichtung muß σ_i bekannt sein. Dazu werden in einer Kontrolleinheit zwischen dem *xy* und *z* Pfad die Vorzeichen der drei Pfade ausgewertet und entsprechend σ_i bestimmt. Die Skalierungsfaktorkompensation erfolgt durch Multiplikationen von *x* und *y* mit $(1+\text{sign}(a(m,i))2^{-|a(m,i)|})$, wobei die $a(m,i)$ festliegen. Dies resultiert in einer Vektorkontraktion (m = 1, $1/k_1$ = 0.784039965) oder -expansion (m = -1, $1/k_{-1}$ = 1.327798882). Dazu kann die gleiche Hardware wie bei der Iteration verwendet werden, nur die Shiftverdrahtung und die Kontrolleinheit ändert sich. Die wesentlichen technischen Daten des Chips (Bild 3) sind in Tabelle III zusammengefaßt. Es erreicht eine normalisierte Spitzenleistung von 220 MFLOPS.

⇐ **Bild 3:** Chipfoto

Literatur

[1] J.S. Walter, "A unified algorithm for elementary functions", *Proc. of Spring Joint Computer Conference*, Band. 38, S. 379-385, 1971

[2] D. König, J.F. Böhme, "Optimizing the CORDIC algorithm for processors with pipeline architecture", Proc. EUSIPCO-90, S. 1391-1394, Barcelona, Sept. 1990

[3] D.Timmermann, H. Hahn, B.J. Hosticka, "Low latency time CORDIC algorithms", IEEE Trans. on Computers, S. 1010-1015, August 1991

Prof. Dr.-Ing. Dirk Timmermann, Universität Rostock, Fachbereich Elektrotechnik, Institut für Technische Informatik, Richard-Wagner Str. 31, 18119 Rostock-Warnemünde, ☎ 0381 57-224