LC-2　浮動小数点3Dユークリッドノルム計算回路の設計と評価

Design and Evaluation of Floating-Point 3D Euclidean Norm Computing Circuit

熊澤 文雄† 高木 直史† 高木 一義†
Fumio Kumazawa Naofumi Takagi Kazuyoshi Takagi

1 はじめに

三次元ベクトルのユークリッドノルム計算は，コンピュータグラフィックスなどで頻出するベクトルの正規化の過程で用いられる演算である．我々はこの計算の専用回路の実現を目指し，VLSIでの実現に向いたハードウェアアルゴリズムを提案し[1]，それに基づく仮数部計算回路の設計を行ってきた[2]．

本論文では，以前提案した基数2及び基数4のアルゴリズムの改良を行う．また，桁合せと仮数部計算を統合し，回路全体のハードウェア量を削減する手法を提案する．これらに基づく浮動小数点3Dユークリッドノルム計算回路を設計し，評価を行う．

2 ハードウェアアルゴリズム

IEEE754標準の基本フォーマットで表される三つの浮動小数点数 F_1，F_2，F_3 に対し，$F_P=\sqrt{F_1^2+F_2^2+F_3^2}$ を仮数部の誤差が1 ulp以内となるように求める．F_i の指数部を E_i，仮数部を M_i とし，$F_i=2^{E_i}\cdot M_i$ とする．

F_1，F_2，F_3 の中で，指数が最大であるものを F_X とし，残りの二つを F_Y，F_Z とする．$X=M_X\cdot 2^{-1}$，$Y=M_Y\cdot 2^{E_Y-E_X-1}$，$Z=M_Z\cdot 2^{E_Z-E_X-1}$ とし，指数部が 2^{E_X+1} となるように仮数部の桁合せを行うと，$F_P=\sqrt{F_X^2+F_Y^2+F_Z^2}=\sqrt{X^2+Y^2+Z^2}\cdot 2^{E_X+1}$ となる．

以前提案したハードウェアアルゴリズム[1]は仮数部の計算を行うもので，$\frac{1}{2}\le X<1$，$0\le Y<1$，$0\le Z<1$ である r 進小数 X，Y，Z（r は基数，2の冪）に対し，$P=\sqrt{X^2+Y^2+Z^2}$ を計算する．P は，初期値を上位から一桁ずつ順に補正していくことにより求める．アルゴリズムは以下のように表される．

[ユークリッドノルム計算アルゴリズム]

Step 1: $P[0]^+:=X+2^{-1}+1$; $P[0]^-:=X+2^{-1}-1$;
$W[0]:=Y\cdot Y_h+Z\cdot Z_h-X-2^{-2}$;
$(Y_h=[.y_1y_2\cdots y_h], Z_h=[.z_1z_2\cdots z_h])$

Step 2: **for** $j:=0$ **to** $n-1$ **do**
$q_{j+1}:=$ **select**$(r\hat{W}[j], \hat{P}[j])$;
$W[j+1]:=rW[j]+Yy_{j+h+1}r^{-h}+Zz_{j+h+1}r^{-h}-2P[j]q_{j+1}-q_{j+1}^2r^{-j-1}$;
$P[j+1]^+$ と $P[j+1]^-$ を拡張on-the-fly変換の規則にしたがって計算する;
end for

Step 3: 結果 $P[n]$ を $P[n]^+$ と $P[n]^-$ から得る;

select() は桁選択関数，$r\hat{W}[j]$ は $rW[j]$ の小数点以下 t ビットまでの値，$\hat{P}[j]$ は $P[j]$ の小数点以下 d ビットまでの値である．h は非負の整数であり，補正値 q_{j+1} の選択が可能となるように定める．計算を高速化するため，残余 $W[j]$ を桁上げ保存形で表現し，中間結果 $P[j]$ を二つの非冗長表現 $P[j]^+=P[j]+r^{-j}$，$P[j]^-=P[j]-r^{-j}$ で保持する．残余 $W[j+1]$ の漸化式では，開平計算と二乗和計算（$+Y^2+Z^2$）を重畳している．Step1は $j=-h$ から -1，Step3は $j=n$ としてStep2の反復を行うことにより実現できる．

2.1 基数2の改良アルゴリズム

基数 r が2，補正値 q_{j+1} の桁集合が $\{-1,0,1\}$ の場合について考える．以前提案したアルゴリズム[1]では，$t=2$，$h=3$ としていたが，より詳細な考察により，$t=1$，$h=3$ とできることが判明した．これにより，補正値 q_{j+1} の選択に必要な $2W[j]$ のビット数が1少なくなる．基数2の改良アルゴリズムは以下のようになる．

[基数2のアルゴリズム]

Step 1: $P[0]^+:=[1.1x_2\cdots x_n]$; $P[0]^-:=[(-1).1x_2\cdots x_n]$;
$W[0]:=Y\cdot Y_3+Z\cdot Z_3$;
$(Y_3=[.y_1y_2y_3], Z_3=[.z_1z_2z_3])$

Step 2: **for** $j:=0$ **to** $n-1$ **do**

$$q_{j+1}:=\begin{cases}-1 & if \quad 2\hat{W}[j]\le -\frac{3}{2}\\ 0 & if \quad -1\le 2\hat{W}[j]\le -\frac{1}{2}\\ 1 & if \quad 0\le 2\hat{W}[j]\end{cases};$$

$(2\hat{W}[j]$ は $2W[j]$ の小数点以下1ビットまで)
$W[j+1]:=2W[j]+Yy_{j+4}2^{-3}+Zz_{j+4}2^{-3}-2q_{j+1}(P[j]+2^{-j-2}q_{j+1})$;
$P[j+1]^+$ と $P[j+1]^-$ を拡張on-the-fly変換の規則にしたがって計算する;
end for

Step 3: 結果 $P[n]$ を $P[n]^+$ と $P[n]^-$ から得る;

2.2 基数4の改良アルゴリズム

基数 r が4，補正値 q_{j+1} の桁集合が $\{-2,-1,0,1,2\}$ の場合について考える．以前提案したアルゴリズム[1]では，$d=5$，$t=5$，$h=5$ としていたが，より詳細な考察により，$d=5$，$t=7$，$h=3$ などとできることが判明した．これにより，h を小さくできるので，Step1の計算に必要な反復回数が少なくなる．基数4の改良アルゴリズムは以下のようになる．

[基数4のアルゴリズム]

Step 1: $P[0]^+:=[2.(x_1-2)x_2\cdots x_n]$;
$P[0]^-:=[0.(x_1-2)x_2\cdots x_n]$;
$W[0]:=Y\cdot Y_3+Z\cdot Z_3-X-2^{-2}$;

Step 2: **for** $j:=0$ **to** $n-1$ **do**
if $j=0$ **then** $e:=2^{-1}$; **else** $e:=0$;
$q_{j+1}:=$

$$\begin{cases}-2 & if \quad 4\hat{W}[j]<-3\hat{P}[j]+e-2^{-4}\\ -1 & if \quad -3\hat{P}[j]+e-2^{-4}\le 4\hat{W}[j]<-\hat{P}[j]\\ 0 & if \quad -\hat{P}[j]\le 4\hat{W}[j]<\hat{P}[j]\\ 1 & if \quad \hat{P}[j]\le 4\hat{W}[j]<3\hat{P}[j]+e+2^{-5}\\ 2 & if \quad 3\hat{P}[j]+e+2^{-5}\le 4\hat{W}[j]\end{cases};$$

$(\hat{P}[j]$ は $P[j]$ の小数点以下5ビットまで)
$(4\hat{W}[j]$ は $4W[j]$ の小数点以下7ビットまで)
$W[j+1]:=4W[j]+Yy_{j+4}4^{-3}+Zz_{j+4}4^{-3}-2q_{j+1}(P[j]+\frac{1}{2}q_{j+1}4^{-j-1})$;
$P[j+1]^+$ と $P[j+1]^-$ を拡張on-the-fly変換の規則にしたがって計算する;
end for

Step 3: 結果 $P[n]$ を $P[n]^+$ と $P[n]^-$ から得る;

† 名古屋大学大学院工学研究科, Department of Information Engineering, Nagoya University

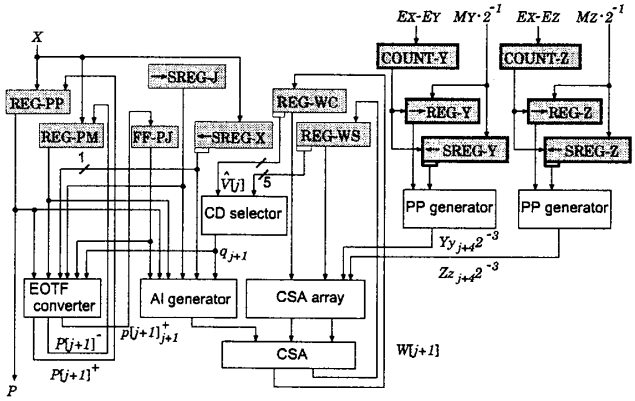

図 1: 基数 2 の仮数部計算回路の構成

3 桁合せと仮数部計算の統合

浮動小数点数に対する 3D ユークリッドノルム計算では，仮数部計算の前に，指数部を比較して仮数部の桁合せを行う処理が必要となる．この桁合せを行う回路と仮数部計算回路を別個に実現する場合，F_Y の仮数部 M_Y と F_Z の仮数部 M_Z の桁合せを行うために，最大 n 桁右シフトするバレルシフタが二つ必要となる．そこで，桁合せ処理を仮数部計算に埋め込むことを考える．以下では，仮数部計算回路が基数 2 のアルゴリズムに基づく場合について述べる．

桁合せ後の仮数 $Y = M_Y \cdot 2^{E_Y - E_X - 1}$ について考える．

$$Y = [.\underbrace{0\cdots0}_{E_X - E_Y}\ 1 y_{E_X - E_Y + 2} \cdots y_{E_X - E_Y + n}] \qquad (1)$$

である．仮数部計算回路において Y は，Y^2 の部分積 $Yy_{j+4}2^{-3}$ を計算するために用いられる．Step 1 と Step 2 で，反復回数 j の値は -3 $(= -h)$ から $n-1$ まで変化するので，1 反復ごとに，$Yy_1 2^{-3}$ から $Yy_{n+3}2^{-3}$ までの部分積が計算される．式 (1) より，最初の $E_X - E_Y$ 回の反復の間，部分積の値は 0 であり，Y の値が必要になるのは $E_X - E_Y + 1$ 回目からである．したがって，M_Y の桁合せと Y^2 の部分積の計算を次のような手法で行うことができる．

1. シフト前の仮数 $M_Y \cdot 2^{-1}$ を仮数部計算回路に入力する．
2. 最初の $E_X - E_Y$ 回の反復の間は，仮数を 1 ビット右シフトし，部分積の値を 0 とする．
3. 仮数の値は Y となるので，以降は仮数をシフトさせず，部分積の値を $Yy_{j+4}2^{-3}$ とする．

M_Z の桁合わせと Z^2 の部分積の計算も同様にして行うことができる．

4 回路の構成

回路全体は，前処理回路，仮数部計算回路，正規化回路で構成する．仮数部計算回路については，基数 2，基数 4，基数 2 の二段重ねの三種の構成を示す．

前処理回路では，F_1，F_2，F_3 の指数部を比較し，F_X，F_Y，F_Z を決定する．仮数 X，$M_Y \cdot 2^{-1}$，$M_Z \cdot 2^{-1}$ 及びシフト量 $E_X - E_Y$，$E_X - E_Z$ を仮数部計算回路へ出力する．桁合せは行わない．

基数 2 のアルゴリズムに基づく仮数部計算回路の構成を図 1 に示す．太線で示されたモジュールは，**3** で示した手法を用いるために [2] で示した構成を変更した箇所である．基数 4 の仮数部計算回路の構成もほぼ同様となる．

計算に必要なサイクル数を少なくするには，高基数で回路を実現するほかに，小さい基数の反復を 1 サイクルに数回計算する方法がある．そこで，1 サイクルで [基数 2 のアルゴリズム] の Step 2 の for ループの反復の 2 回分の計算を行う順序回路としての実現を考える．基数 2 による実現の組合せ回路部を単純に二段にしたのでは，サイクル時間，ハードウェア量とも 2 倍になってしまう．そこで，補正値 q_{j+2} の選択を補正値 q_{j+1} の選択と重畳させて行うことで高速化する．

正規化回路では，仮数を 1 以上 2 未満に正規化する．

表 1: 回路の面積及び計算時間

基数	統合なし	統合あり		
	2	2	4	2 の二段
面積 [mm^2]	0.4350	0.3503	0.4494	0.4517
遅延 [ns]	5.55	5.32	8.32	7.32
サイクル数	30	30	18	17
計算時間 [ns]	166.50	159.60	149.76	124.44

表 2: 開平器との比較

	3D ユークリッドノルム計算回路	開平器
面積 [mm^2]	0.3503	0.1832
遅延 [ns]	5.32	3.57
サイクル数	30	25
計算時間 [ns]	159.60	89.25

5 回路の評価

IEEE754 標準の単精度浮動小数点基本フォーマットを入出力とする回路を設計した．セルライブラリにはローム社 0.35μm プロセス用の東大 VDEC 版 EXD 社ライブラリを用いた．設計した回路の面積及び計算時間を表 1 に示す．

3D ユークリッドノルム計算回路は，3 回の平方算，2 回の加算，1 回の開平算を一度に行う回路である．残余の漸化式からわかるように，開平算に二乗和計算を取り込んだ回路となっている．そこで，基数 2 の開平計算アルゴリズム [3] に基づく浮動小数点開平器を設計し，比較を行った（表 2）．

6 まとめ

以前提案した基数 2 及び基数 4 のユークリッドノルム計算アルゴリズムの改良を行い，それらに基づく浮動小数点 3D ユークリッドノルム計算回路の設計及び評価を行った．回路の評価の結果，桁合せ処理を仮数部計算に埋め込むことにより，面積を約 20%削減することができた．また，平方算 3 回，加算 2 回，開平算 1 回に相当する 3D ユークリッドノルム計算を行う回路が，開平器に比べ，約 1.5 倍の遅延，約 2 倍の面積，約 2 倍の計算時間で実現できることが明らかになった．

参考文献

[1] N. Takagi and S. Kuwahara, "A VLSI Algorithm for Computing the Euclidean Norm of a 3D Vector," IEEE Trans. Comput., vol.49, no.10, pp.1074–1082, Oct. 2000.

[2] 武内大輔，高木一義，高木直史，"三次元ベクトルのユークリッド・ノルム計算回路の設計，" 2000 信学ソ大（基礎・境界），A-3-14, pp.81, Oct. 2000.

[3] M. D. Ercegovac and T. Lang, "Division and Square Root – Digit-Recurrence Algorithms and Implementations," Kluwer Academic Publishers, 1994.