2005－SLDM－122 (15)
2005／12／1

特定用途向け低ビット複合演算回路設計

大窪啓太† 朝利壮吾‡ 矢野智則‡ 神戸尚志‡

†近畿大学大学院 総合理工学研究科 エレクトロニクス系工学専攻
‡近畿大学 理工学部 電気電子工学科

〒577-8502 大阪府東大阪市小若江 3-4-1
E-mail: †0533340406p@kindai.ac.jp, ‡tkambe@ele.kindai.ac.jp

あらまし 組み込みシステムに用いる演算回路は、回路規模、処理時間、精度の間でトレードオフがあり、どの項目を優先すべきかは対象となるシステムに依存する。本稿では、処理時間を優先するシステムに対して、分母に平方根を持つ除算処理回路を減算シフト型アルゴリズムとテーブル近似を併用した手法で設計し、処理時間短縮の効果を示す。

キーワード 開平, 除算, 2乗計算, 減算シフト型アルゴリズム, テーブル近似

Application Specific Arithmetic Circuit Design

Keita OKUBO† Sougo ASARI‡ and Tomonori YANO‡ and Takashi KAMBE‡

† ‡ Kinki University, 3-4-1 kowakae, Higashi-Osaka City, Osaka, 577-8502 Japan
E-mail: †0533340406p@kindai.ac.jp, ‡tkambe@ele.kindai.ac.jp

Abstract In the design of the arithmetic circuit for the embedded system, it is necessary to consider optimization of circuit scale, processing time, and calculation accuracy simultaneously and each embedded system may have different criteria to be optimized. The processing time of division circuit with the square root in the denominator is important for the particle tracking system. In this paper, it is designed based on Digit-Recerrence algorithm and table reference and evaluated its performance.

Keyword Square Root, Division, Square Calculation, Digit-Recurrence Algorithm, Table Approximation

1. はじめに

演算回路の設計目標は組み込み対象となるシステムに応じて変化する。低速なCPUを用いた場合でも高度な処理を実現するために、対象となるシステム用途に特化した演算回路を設計することで、システムの性能を向上させることが必要となる。

今回組み込み対象とするシステムは粒子マスク相関法とカルマンフィルタとx^2検定を組み合わせたKC法からなる粒子追跡プログラムのリアルタイム処理を目的としてHW化を行なう。粒子マスク相関法では入力画像(1008pixel×1018pixel)を4倍し、1pixelごとに粒子の識別を行う。その粒子マスク相関法で行う相関値計算は、分母に平方根を含んだ除算処理を含み、その出力がしきい値(0.7)より大きいか小さいかを判別することで、粒子を識別する。相関値計算の式を(1)式に示す。(1)式中のパラメータは以下の意味を持つ。

n, m:水平, 鉛直相関領域サイズ i, j:水平, 鉛直位置
I:元画像の輝度値 $\overline{I}$:元画像の輝度値の平均
M:元画像の輝度値の平均 $\overline{M}$:マスク画像の平均値

$$r = \frac{\sum_{i,j=1,1}^{n\times m}\left(I_{i,j}-\overline{I}\right)\left(M_{i,j}-\overline{M}\right)}{\sqrt{\left[\sum_{i,j=1,1}^{n\times m}\left(I_{i,j}-\overline{I}\right)^2\right]\left[\sum_{i,j=1,1}^{n\times m}\left(M_{i,j}-\overline{M}\right)^2\right]}} \quad (1)$$

(1)式において、

$$\sqrt{\left[\sum_{i,j=1,1}^{n\times m}\left(M_{i,j}-\overline{M}\right)^2\right]}$$

はマスクの輝度の標準偏差(定数=0.030088)であり、標準偏差を0.7に乗じた値(0.121421)をしきい値とする。

入力画像に対して、粒子の識別処理は400万回必要となり、演算回路の処理サイクル数を削減することができれば、画像全体の処理サイクルを大幅に削減することになり、システム性能を大きく向上させることができる。

本稿では、頻繁する演算処理に対してサイクル数を削減し、かつ必要精度を保つための手法として、漸化式の繰り返し計算とテーブル近似を併用した手法を示し、その効果について述べる。

2. 演算アルゴリズム

回路の設計には、しきい値よりも大きいか小さいかを判別するため、解となる小数点以下のビットを漸化式の繰り返し計算により、上位から1ビットずつ算出していく減算シフト型アルゴリズム[1]を採用した。近似に用いる漸化式を以下に示す。

2.1. 漸化式

分母D、分子N、中間結果Snとしたとき、(2)式が得られる。

$$S_n = \frac{N}{\sqrt{D}} \qquad (2)$$

(2)式より、減算シフト型HWアルゴリズムの自動合成手法[1]を用いると、分母に平方根を持つ除算処理を行う漸化式が得られる。

$$\begin{aligned} X_{n+1} &= 2X_n - 2\cdot Y_n \cdot s_{n+1} - D\cdot 2^{-n-1} \\ Y_{n+1} &= Y_n + D\cdot s_{n+1}\cdot 2^{-n-1} \end{aligned} \qquad (3)$$

(3)式中のXnは残余、Ynは分母に中間結果を乗じた値、s_nは商のn桁目の値を表す。初期値$X_0=N^2, Y_0=0$としたとき、残余Xnの正負を判別することで、小数点以下の商を上位から1ビットずつ求める。ここで、初期値計算に、分子Nの2乗を求める必要があり、乗算回路が必要となる。今回この乗算回路を2乗計算の特性を生かした2乗計算専用回路を用いることで、処理の高速化及び回路規模の削減を図った。

2.2. 2乗計算

2乗計算では、部分積に以下のような特徴がある。図2.1で示すように、○で囲まれた部分積を線で結び、その線を境界とした上部と下部の部分積は等しくなる。上部もしくは下部のどちらか片方の部分積の和を2倍した値と○で囲んだ部分の和が部分積の総和と等しく、加え合わせを必要とする部分積の個数を削減することができる。

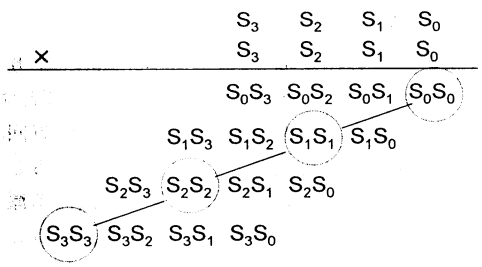

図2.1 二乗計算の部分積

Fig.2.1 Partial product of square calculation

Boothアルゴリズムを用いた乗算回路と、二乗計算専用回路を用いた場合と比較を行った。16ビットの乗算を、CSAのWallace木構造と桁上げ先見加算を用いて構成した。表2.1に論理合成の結果を示す。

表2.1 乗算回路の比較

Table.2.1 Comparison of multiplication circuits

	回路規模(gate)	処理時間(ns)
Booth乗算回路	4,830	10.12
2乗専用回路	1910	7.65

Boothを用いた乗算回路に比べ、部分積生成に複雑な処理を必要としないことから、2乗計算専用回路が規模及び処理時間において良好な結果が得られた。

2.3. 計算範囲の限定

今回採用したアルゴリズムは商の小数点以下のビットを上位から一桁ずつ求めていくものであり、$N < \sqrt{D}$である。この条件を満たすために入力値を以下の範囲に限定する。

$$0.5 \le N < 1$$
$$1 \le D < 4$$

範囲外の値はシフト操作により限定範囲内に収める。

限定範囲内に収めた値はアルゴリズムを用いた計算を行った後に値を復元する必要がある。値の復元もシフト操作を用いて行う。復元時に行うシフト数及び右シフトであるか、左シフトであるかの判定は、範囲限定時の分子のシフト数nS、分母のシフト数dSとし、右シフトを－、左シフトを＋で表したとき、復元時のシフト数rSは以下の式で求める。

$$rS = \frac{dS}{2} - nS \qquad (4)$$

以上に述べたように、シフト操作を多く使用することから、アンダーフローによる演算精度の低下が考えられるが、処理時間は削減される。

3. テーブル近似

漸化式を繰り返し計算することで、商の上位ビットから順に1ビットずつ求めているため、精度は繰り返し回数に依存する。繰り返し回数を削減し、なおかつ目標とする精度を達成できれば、処理時間を優先とするシステムにおいても高い精度が得られる。

本文では繰り返し回数を削減し、なおかつ精度を保つ手法として、テーブルを用いた手法を提案する。しかし、商の全ビットをテーブルに持つと、分母分子に変数を持つ除算では、演算に用いるビット数をnとしたとき、2^{2n}乗通りの値を記憶しなければならず、回路規模が非常に大きくなってしまう。そこで、テーブル

を用いて近似を行う部分を商の下位のビットに限定し、上位のビットは漸化式の繰り返しにより求める。

3.1. テーブル作成方法

(3)式に示した漸化式において、繰り返し回数 n→∞ とすると、$D\cdot 2^{-n-1}$ は 0 に収束する。これを利用すると、(5)式の様になる

$$X_{n+1} = 2X_n - 2\cdot Y_n \cdot s_{n+1} \qquad (5)$$

Yn は $D\cdot 2^{-n-1}=0$ であると考えると、繰り返し計算を行っても値が変化することはない。

(5)式が与えられたとき、Xn及びYnの値が分かれば、その後の繰り返し計算を行った際に Xn の符号の変化を予め計算しておいた値を格納したテーブルから読み出すことで近似を行う。但し、Xn,Yn の全てのビットをインデックス情報としたテーブルを用意するとサイズが大きくなる。Xn,Yn を少ないビット数で表すことで、テーブルサイズを抑えることができる。

範囲限定をしていることから Yn の値は 0.5～2 までの値となる。0.5～2 の区間を細かく分割し、分割したどの区間に Yn の値が存在するかを調べる。分割した各区間はインデックス情報を持っており、Yn の値からインデックス情報 Yindex を求める。Xn の値は Yn の 2 倍以上となると、(5)式を何度繰り返し計算しても Xn の符号に変化はない。このことから、Xn の値は 0～4 を分割した区間から選択することでインデックス情報 Xindex を求める。区間選択で求めた Xindex と Yindex の合計ビットがテーブルのインデックス数となる。Xn が負の数である場合は、Xn の 2 の補数を用いて区間選択を行い、テーブルから得られたビットを反転する。漸化式の計算とテーブル近似を併用した場合の回路構成を図 3.1 に示す。

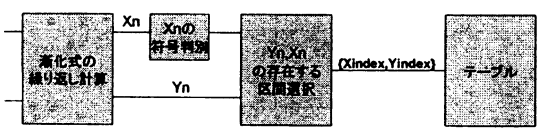

図 3.1　回路構成

Fig.3.1 Block diagram of the circuit

漸化式の繰り返し計算で Yn=1.865,Xn=0.425 が得られた場合の区間選択及びテーブル読み出しの例を示す。図 3.2 は 0.5～2 を 0.09375 間隔で 16 分割したもので、インデックス情報は 4 ビット必要となる。Yn=1.865 であり、1.8125～1.90625 の区間に存在することから、インデックス情報 Yindex=1110 を得る。

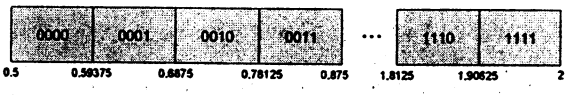

図 3.2　Yn の区間選択

Fig.3.2 Interval selection of Yn

図 3.3 は 0～4 を 32 分割したもので、Xindex は 5 ビット必要になる。Xn=0.425 であり、0.375～0.5 の区間に存在することから、インデックス情報 Xindex=00011 を得る。

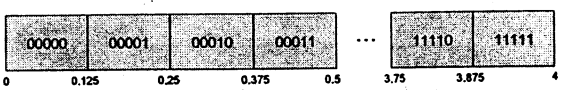

図 3.3　Xn の区間選択

Fig.3.3 Interval selection of Xn

テーブルには 000111110 {Xindex,Yindex} を入力し、Yindex から得られる区間の中間値 1.859375 と Xindex から得られる区間の中間値 0.4375 を用いて(5)式より計算した値が保持してあり、インデックス情報を元に値を読み出す。読み出した値は、漸化式の繰り返し計算で求めた商に連結する。漸化式で求める商のビット数とテーブルで求める商のビット数の振り分けは、対象となるシステム用途に応じて最適なものを選ぶ。

3.2. 計算精度の比較

テーブル近似を用いた場合の計算精度の比較を行った。その結果を以下に示す。今回ターゲットとなるシステムは整数 24 ビット、小数 20 ビットの固定小数点方式を用いており、演算回路の出力を小数点以下 20 ビットとする。但し、演算回路の目的は、しきい値よりも大きいか小さいかの判別である。2.3 節で述べた範囲限定を行っていることから、商は 0.25～1 の値となり、しきい値付近になる可能性があるものは、値の復元を行う際に、2 又は 3 ビットの右シフトを行う。このことから、小数点以下 18 ビットまで求める。実際に粒子追跡システムで使用しているデータを用いて、漸化式の繰り返し回数 14 回＋テーブル出力 4 ビット、繰り返し回数 12 回＋テーブル出力 6 ビット及び繰り返し回数 10 回＋テーブル出力 8 ビットの場合の精度検証を行った。以下に各テーブル出力ビットの比較を示す。

3.2.1. 4 ビット出力テーブル

4 ビット出力テーブルを用いた場合、漸化式の繰り返し計算は 14 回必要となる。インデックス情報の生成に用いる X 及び Y 区間の分割数は、3.1 で例を示した X=32 分割＋Y=16 分割の場合と、倍の分割数 X=64 分

割＋Y=32 分割での比較を行った。また、14 回の繰り返し計算を行い、テーブル近似を使用しなかった場合についても比較を行った。表 3.1 に比較結果を示す。比較に用いた計算データは、商がしきい値付近になる可能性のある値(計算後、値の復元で 2 及び 3 ビットの右シフトを行うもの)を 10 万個用いた。誤差値は全処理結果を SW の結果と比較した際の相対誤差の平均値である。また、誤差の個数はしきい値との大小の判別が SW と一致しているかを調べたものである。

表 3.1　4 ビットテーブルによる計算結果
Table.3.1 Calculation results using 4bit table

	誤差値	誤差の個数
X=32 分割＋Y=16 分割	1.495×10^{-5}	0
X=64 分割＋Y=32 分割	1.480×10^{-5}	0
テーブル未使用	1.210×10^{-4}	2

テーブルを使用した場合とそうでない場合を比較すると約 10 倍精度を向上させることができた。

3.2.2. 6 ビット出力テーブル

6 ビット出力テーブルを用いた場合、4 ビット出力テーブルの場合と同様の条件で処理精度の比較を行った。その結果を表 3.2 に示す。

表 3.2　6 ビットテーブルによる計算結果
Table.3.2 Calculation results using 6bit table

	誤差値	誤差の個数
X=32 分割＋Y=16 分割	1.856×10^{-5}	0
X=64 分割＋Y=32 分割	1.561×10^{-5}	0
テーブル未使用	3.249×10^{-4}	4

テーブルを使用した場合とそうでない場合を比較すると約 20 倍精度を向上させることができた。

3.2.3. 8 ビット出力テーブル

8 ビット出力テーブルを用いた場合、4 及び 6 ビット出力テーブルの場合と同様の条件で処理精度の比較を行った。その結果を表 3.3 に示す。

表 3.3　8 ビットテーブルによる計算結果
Table.3.3 Calculation results using 8bit table

	誤差値	誤差の個数
X=32 分割＋Y=16 分割	5.140×10^{-5}	0
X=64 分割＋Y=32 分割	2.812×10^{-5}	0
テーブル未使用	1.279×10^{-3}	19

テーブルを使用した場合とそうでない場合を比較すると、大きく精度を向上させることができたが、4 ビット及び 6 ビット出力テーブルを用いた場合と比較すると区間分割数への精度の依存度が高くなっている。更に高い精度を必要とする場合や、8 ビット以上の出力テーブルを使用する場合には、区間分割数を多くする必要があり、小規模実装は困難になってくる。

4. 演算回路の設計結果

全体の回路構成を図 4.1 に示す。演算処理を行う 4 つのモジュールと、それを制御する回路で構成した。初期値生成の 2 乗計算には専用回路を用いた。各モジュールは 1 サイクルで処理を行う。動作周波数は 100MHz で合成を行った。

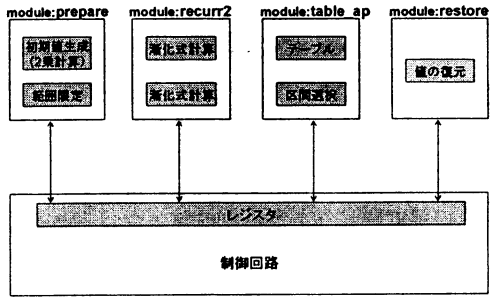

図 4.1　テーブルを用いた回路構成
Fig.4.1 The circuit diagram using table

インデックス情報の生成は X=64 分割＋Y=32 に分割した区間から選択し、漸化式の繰り返し 14 回＋4 ビット出力テーブルを用いた場合の回路規模を表 4.1 に示す。比較対象としては、テーブル近似を使用しない場合でかつ近似精度が同等となる漸化式の繰り返し回数を必要とする回路(漸化式の繰り返し回数 18)を選択した。

表 4.1　4 ビット出力テーブルを用いた回路性能
Table.4.1 Circuit performance(using 4bit table)

	回路規模 (gate)	処理時間 (ns)	誤差値
4 ビット出力テーブル使用	18,328	100	1.480×10^{-5}
漸化式の繰り返し 18 回(テーブルなし)	12,891	110	1.467×10^{-5}

テーブル近似を用いた場合、インデックス情報の生成部とテーブルが付加されることから回路規模が増加してしまう。特に、インデックス情報の生成には分割した多くの区間から選択する必要があり、セレクタによる回路規模の増加が大きい。そこで、回路規模の縮小を目的として、セレクタを用いないインデックス情報の生成手法を考える。

4.1. セレクタの削除

分割した区間の中から Xn 及び Yn がどの区間に存在するかの選択において、Xn は 0～4 の値を 64 分割して

いる。2^{-4}間隔で分割した区間の選択は整数部が、4以上であるかどうかの判別をした後に、4より小さければ、Xnの整数2ビット＋小数4ビットを直接インデックス情報として扱うことができる。以下に例を示す。

例）0～4を2^{-4}間隔で分割した場合

		小数点の位置
4	=	100 0000
3.9375	=	11 1111
3.875	=	11 1110
3.8125	=	11 1101

3.8125≦ Xn<3.875であるとき、Xnの整数2ビット、小数4ビットが111101となり、それ以下のビットは区間選択に影響しない。111110である場合は3.875≦ Xn<3.9375となる。2^{-n}間隔に分割したとき、小数点以下n桁目までを調べることで区間選択を行える。

例に示した整数2＋小数4ビットで調べる場合、4以上であるかどうかを判別する必要がある。範囲限定をしている関係から、整数部に必要なビット数は4ビットとなり、4以上であるかどうかの判別は容易に行うことができる。このように区間を2のべき乗間隔にすることで、大規模なセレクタを必要としない。

Ynの区間選択は、0.5～2の値を32分割しており、1.5/32=0.046875と区間間隔が2のべき乗とならないことから、セレクタを必要とする。ここで、Ynの区間選択を0～2に変更する。これにより、2/32=0.0625と区間間隔が2のべき乗となり、セレクタが不要になる。セレクタ削減後の回路構成は、テーブル近似部のパスに余裕ができたため、図4.2に示すようにテーブル近似と値の復元を1サイクルで行うように変更した。

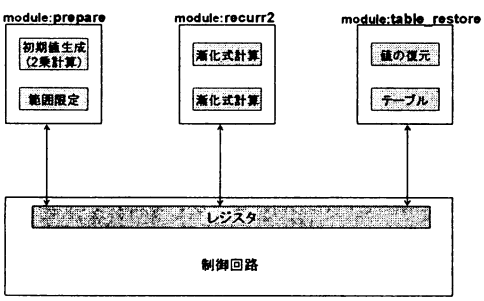

図4.2　セレクタ削減後の回路構成

Fig.4.2 Circuit diagram after selector is reduced

Ynの存在する区間を調べるセレクタを削除したことによる回路規模の変化を表4.2に示す。

表4.2　セレクタ削減の効果

Table.4.2 Effect of selector reduction

	回路規模	処理時間	誤差値
セレクタ使用	18,328	100	1.480×10^{-5}
セレクタ未使用	16,080	90	1.484×10^{-5}

Ynの区間を大きくしたことから、多少の精度低下が見られたが、規模削減効果が得られた。

4.2. 改良回路の設計結果

セレクタ削減の手法を適応し、漸化式の繰り返し12回＋6ビット出力テーブルを使用した回路と、漸化式の繰り返し10回＋8ビット出力テーブルを使用した回路の性能を表4.3と表4.4に示す。比較対象には、テーブルを使用しなかった場合に精度が同等となる漸化式の繰り返し回数の回路を用いた。6ビットテーブルを使用した場合は、使用しなかった場合の漸化式繰り返し17回と同等の精度となり、8ビットテーブルを使用した場合は16回と同等の精度となった。

表4.3　6ビット出力テーブルを用いた回路性能

Table.4.3 Circuit performance(using 6bit table)

	回路規模	処理時間	誤差値
6ビット出力テーブル使用	16,694	80	1.594×10^{-5}
漸化式の繰り返し17回(テーブルなし)	12,891	100	1.747×10^{-5}

表4.4　8ビット出力テーブルを用いた回路性能

Table.4.4 Circuit performance(using 8bit table)

	回路規模	処理時間	誤差値
8ビット出力テーブル使用	16,894	70	3.127×10^{-5}
漸化式の繰り返し16回(テーブルなし)	12,891	100	2.730×10^{-5}

テーブル近似を使用した場合、テーブルを使用しなかった場合と比較すると同等の精度を得るのに必要なサイクル数は2～3サイクル短縮することができた。

8ビット出力のテーブルを用いた場合、他と比較すると、精度の低下が大きくなった。(5)式に示した変更後の漸化式は多く繰り返し計算を行った場合にのみ適応するもので、10回の繰り返し計算しか行わない8ビット出力テーブルの場合、漸化式変更に伴う誤差が大きくなるため、何らかの対策が必要となる。

最後に、本研究の成果を示す。漸化式の初期値生成に2乗専用回路を用いたことによる規模削減効果と、テーブル近似を用いたサイクル数削減効果を表4.5に示す。

表 4.5 設計結果のまとめ

Fig.4.5 Summary of design results

構成	回路規模	処理時間	誤差値
Booth乗算 + テーブルなし	18,292	110	1.467×10^{-5}
2乗専用回路 + テーブルなし	12,891	110	1.467×10^{-5}
2乗専用回路 + テーブル使用	16,080	90	1.484×10^{-5}

本稿に示した結果はSynopsys社のDesign Compilerを使用し、ライブラリはHITACHI0.18μmを用いて合成を行った。

5. まとめ

分母に平方根を含んだ除算を同時処理するアルゴリズムに2乗専用回路を用いることで、回路規模を削減した。また、漸化式の繰り返し計算とテーブル近似を併用することで、テーブルを使用しなかった場合と比較すると回路規模は増加したものの、2〜3サイクル処理を短縮することができた。

文 献

[1] 熊澤文雄、高木直史："算術演算のための減算シフト型ハードウェアアルゴリズムの自動合成"、情報処理学会研究報告、pp.245-248,SLDM-117,2004

[2] 高木直史："算術演算のVLSIアルゴリズム"、コロナ社、東京、2005

[3] Israel Koren："Computer Arithmetic Algorithms"、A K Peters、2002

[4] Muller, J.M："Elementary Functions", Birkhäuser, 1997