論　文

複合算術演算の減算シフト型ハードウェアアルゴリズムの設計支援

熊澤　文雄[†a)]　　高木　直史[†]

Design Support of Digit-Recurrence Algorithms for Arithmetic Circuits

Fumio KUMAZAWA[†a)] and Naofumi TAKAGI[†]

あらまし　信号処理やグラフィックスなどで，より高い算術演算性能を実現するためには，頻繁に出現する複合算術演算のための専用回路を開発することが有効な手段の一つであると考えられる．複合算術演算の専用回路を搭載するかどうかを決定するためには，専用回路の基礎となる減算シフト型のハードウェアアルゴリズムを設計し，その回路の面積や遅延などを見積もる必要がある．本論文では，複合算術演算の減算シフト型ハードウェアアルゴリズムの設計において多大な労力を要する桁選択関数の設計について，最適な桁選択関数を求める体系化した方法を提案する．この手法の適用により，種々の実用的な桁選択関数を設計することができ，特に，平方根の逆数計算のアルゴリズムについて，従来提案されているものよりもよい桁選択関数を求めることができた．

キーワード　算術演算回路，ハードウェアアルゴリズム，アルゴリズム合成，VLSI

1. まえがき

信号処理やグラフィックスなど多くの用途において，高い算術演算性能が求められている．より高い演算性能を実現するためには，加算や乗算，除算などの基本演算の高速化とともに，通常は多数の基本演算の組合せにより計算される演算（本論文では複合算術演算と呼ぶ）を直接計算する専用回路を開発することが有効な手段の一つであると考えられる．このような演算の例として，グラフィックスにおいて，頻出するベクトルの正規化の過程で出現する，平方根の逆数計算（$X^{-\frac{1}{2}}$）などが挙げられる．用途において頻繁に出現し，計算に時間のかかる複合算術演算のための専用回路を開発し，プロセッサに搭載することにより，この演算の高速化とともに，他の基本演算器，特に乗算器をこの演算の計算から解放することができるため，プロセッサ全体の演算性能を大きく向上させることができると考えられる．

これまでに，平方根の逆数計算，立方根計算などの複合算術演算に対し，専用回路の基礎となる減算シフト型のハードウェアアルゴリズムが提案されている [1]〜[6]．これらのアルゴリズムは乗算器を用いず直接結果を求める計算法であり，剰余が 0 に収束するように，結果を上位桁から順に 1 桁ずつ計算する．

プロセッサ設計者が，複合算術演算のための減算シフト型の専用回路を採用するかどうかを選択するためには，その演算の出現頻度とともに，その回路の面積や遅延，クロックサイクル数などの見積りが必要である．回路の規模を見積もるためには，減算シフト型アルゴリズムの詳細設計を行い，それをどのように回路として構成するかを定める必要がある．この際に問題となるのが，アルゴリズムの設計に人手では非常に手間が掛かるということである．アルゴリズムを設計するためには，その演算のための，残余（スケーリングした剰余）に関する漸化式を導き，更に，具体的な桁選択関数（残余等から結果の 1 桁を定める関数）を定めなければならない．このうち，特に回路実現に適した桁選択関数を厳密に決定することに多大な労力を要する．本論文では，減算シフト型アルゴリズムの設計における桁選択関数の設計支援について考える．

[1]〜[6] では，桁選択関数の設計について，関数が満たすべき条件をパラメータ（関数の引数となる残余の上位ビット数等）を含む式として導出し，その条件からパラメータの値を定め，具体的な関数を決定している．関数が満たすべき条件の導出は，除算や開平に対し体系化された導出過程 [7], [8] をもとに，演算ごと，

† 名古屋大学大学院情報科学研究科情報システム学専攻，名古屋市
Department of Information Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, 464–8603 Japan
a) E-mail: kumazawa@takagi.i.is.nagoya-u.ac.jp

アルゴリズムの提案ごとに経験により行われており，演算に依存しない一般的な導出手順は知られていない．また，パラメータの値の組を経験や勘により決定しており，得られた関数の最適性は保証されていない．パラメータ間にはトレードオフが存在するが，一つのパラメータの値の組に対し具体的な関数を決定するために大きな手間がかかるため，いくつかのパラメータの値の組に対して試すことも困難な状況である．

本論文では，最適な桁選択関数（トレードオフ曲線上のパラメータの値の組）を求める体系化した方法を提案する．パラメータのとり得る範囲内のすべての値の組に対し，桁選択関数が設計可能かどうかを調べることにより，最適なパラメータの値の組を求める．一つの組に対し調べることに時間が掛かるので,「具体的なパラメータの値の組を与え，その組に対し関数が満たすべき条件を求め，条件を満たす場合に具体的な関数を求める」という部分をプログラム化し，種々の複合算術演算に適用できるように一般化する．特に，関数が満たすべき条件の導出について，これまでの個々の演算の設計法をもとに，厳密な条件を求めるプログラム化可能なアルゴリズムを構築する．

提案した手法の適用により，桁選択回路への入力ビット数が少ない種々の桁選択関数を求めることができるようになった．平方根の逆数計算及び立方根計算を具体例として提案手法を適用したところ，平方根の逆数計算については, [2] に比べ，桁選択回路への入力ビット数が 1 少ない関数が見つかり，立方根計算については，最適な関数として, [5] で提案された関数と同じものが得られた．提案手法により，減算シフト型アルゴリズムに基づく専用回路全体のハードウェア量や計算時間などを見積ることができ，専用回路を搭載するかどうかを短期間で決定できるようになるものと考えられる．

以降，**2.** では，減算シフト型ハードウェアアルゴリズムの設計フローと，従来の桁選択関数の決定法について述べる．**3.** では，最適な桁選択関数を求める体系化した方法を提案する．**4.** では，平方根の逆数計算と立方根計算を例として，**3.** で提案する手法を適用した結果を示す．

2. 減算シフト型アルゴリズムの設計

減算シフト型のアルゴリズムは，人が筆算で除算を行う方法のように，桁選択，加算・減算，シフトといった操作の反復により結果を上位から 1 桁ずつ求めていく計算法である．各反復において，剰余が 0 に収束するように結果の桁の値を定め，漸化式の計算により剰余を更新していく．減算シフト型のアルゴリズムを，反復 1 回分あるいは数回分を実行する順序回路として構成すると，乗算器を必要とせず，一次元配列の規則正しいビットスライス構造となり，比較的少ないハードウェア量で実現できる．

Ercegovac らは，除算，開平のそれぞれに対し，減算シフト型のアルゴリズムの設計法を体系化した [7]. [1]〜[6] で提案された複合算術演算の減算シフト型アルゴリズムの設計は，除算や開平について体系化された設計フローをもととしている．計算したい算術式と，正の小数である演算数が与えられたとき，まず，結果を r 進で小数点以下 n 桁まで求める一般的なアルゴリズムを設計し，次に，r の値を定め具体的なアルゴリズムを設計している．ここに，r は結果の基数であり，2 のべきである．

アルゴリズムの設計フローは，結果を小数点以下 j 桁目まで求めたときの中間結果を $Z[j]$ とし，結果の小数点以下 $j+1$ 桁目の値 q_{j+1} を桁集合 $\{-a, -a+1, \cdots, a\}$（$\frac{r}{2} \leq a < r$）から選択するものとして，以下のように表される．

- Step 1： 基数を r として，残余（スケーリングした剰余）$W[j]$ に関する漸化式を求める．
- Step 2： 具体的な r 及び a の値に対し，中間結果や残余の初期値を定める．
- Step 3： 具体的な r 及び a の値に対し，q_{j+1} を選択する関数を定める．

r 及び a の値の選択により，種々の減算シフト型アルゴリズムを設計できるが，特に，r を 2 または 4 とし，1 反復で 1 ビットまたは 2 ビットずつ結果を求めるアルゴリズムが実用的である．以降，各ステップについて説明する．

Step 1 では，計算したい演算に対し，剰余が 0 に収束するような式として $W[j]$ を定義する．この式の j に $j+1$ を代入し，中間結果 $Z[j]$ に関する漸化式

$$Z[j+1] = Z[j] + q_{j+1} r^{-j-1} \tag{1}$$

から残余に関する漸化式を導く．

例えば平方根の逆数計算のアルゴリズム [2] では，計算したい演算 $X^{-\frac{1}{2}}$ から残余を

$$W[j] = r^j (1 - X Z[j]^2)$$

と定義し，残余の漸化式を

$$W[j+1] = rW[j] - 2XZ[j]q_{j+1} - Xq_{j+1}^2 r^{-j-1}$$

と求めている．この場合，漸化式中に乗算 $XZ[j]$ が含まれるため，[2] では $XZ[j]$ を $P[j]$ と置き，$P[j]$ に関する漸化式

$$P[j+1] = P[j] + Xq_{j+1}r^{-j-1}$$

を求めている．

Step 2 では，まず，結果が満たすべき条件をもとに，残余が満たすべき条件 $\underline{B}[j] \le W[j] < \overline{B}[j]$ を r 及び a を含む式で求める．次に，具体的な r 及び a の値に対し，与えられた演算数の範囲を考慮して，この条件を満たすように初期値を定める．

例えば [2] では，$\rho = \frac{a}{r-1}$ として，

$$\underline{B}[j] = -2XZ[j]\rho + X\rho^2 r^{-j}$$
$$\overline{B}[j] = 2XZ[j]\rho + X\rho^2 r^{-j}$$

が得られ，具体的な r 及び a の値として $r=4$，$a=2$ としたとき，演算数の範囲 $\frac{1}{4} < X < 1$ を考慮して，$X < 2^{-1}$ のとき $\{Z[0]=2, W[0]=1-4X, P[0]=2X\}$，$X \ge 2^{-1}$ のとき $\{Z[0]=1, W[0]=1-X, P[0]=X\}$ と初期値を設定している．

Step 3 は，桁選択関数が満たすべき条件をパラメータ（関数の引数となる残余の上位ビット数等）を含む式として導出するまでの段階，及び，その条件を満たすようにパラメータの値を決め，具体的な桁選択関数を定める段階の二つに分けられる．まず，$W[j+1]$ が満たすべき条件から，q_{j+1} として k $(=-a, -a+1, \cdots, a)$ を選んでよい $rW[j]$ の範囲 $L_k[j] \le rW[j] < U_k[j]$ を r 及び a を含む式で求め，これらをもとに，パラメータを用いて桁選択関数を定義する．次に，具体的な r，a の値に対し，$L_k[j]$ 及び $U_k[j]$ から，関数の入力となる変数の表現方法（桁上げ保存形などの冗長表現を用いるかどうか）や演算数の範囲を考慮して，q_{j+1} の選択の境界となる値が満たすべき条件をパラメータを含む式として求める．この条件を満たし，桁選択回路が小面積かつ高速となるように，パラメータの値を決め，具体的な関数を定める．

例えば [2] では，

$$L_k[j] = 2P[j](k-\rho) + X(k-\rho)^2 r^{-j-1}$$
$$U_k[j] = 2P[j](k+\rho) + X(k+\rho)^2 r^{-j-1}$$

が得られ，これらをもとに，$rW[j]$ を小数点以下 t ビットで打ち切った値を $\widehat{rW}[j]$，$P[j]$ を小数点以下 d ビットで打ち切った値を $\widehat{P}[j]$ として，桁選択関数を

$$q_{j+1} = \begin{cases} -a & \text{if} \quad \widehat{rW}[j] < m_{-a+1}(\widehat{P}[j]) \\ k & \text{if} \quad m_k(\widehat{P}[j]) \le \widehat{rW}[j] < m_{k+1}(\widehat{P}[j]) \\ & \qquad (-a+1 \le k \le a-1) \\ a & \text{if} \quad m_a(\widehat{P}[j]) \le \widehat{rW}[j] \end{cases}$$

と定義している．ここに，$m_k(\widehat{P}[j])$ $(k = -a+1, \cdots, a)$ は桁の選択の境界となる値である．具体的な r 及び a の値として $r=4$，$a=2$ とし，$W[j]$ 及び $P[j]$ を桁上げ保存形で表す場合について，$j \ge 1$ と $j=0$ で別個に関数を定めている．$j \ge 1$ については，$L_k[j]$ 及び $U_k[j]$ から，$4W[j]$ 及び $P[j]$ の打切り誤差を考慮して各 k について選択の境界値 $m_k(\widehat{P}[j])$ が満たすべき条件を求め，それらから，パラメータの値を $d=6$，$t=4$ として具体的な $m_k(\widehat{P}[j])$ を

$$m_{-1}(\widehat{P}[j]) = -tr_4(3(\widehat{P}[j] + 2^{-6})) \quad (2)$$
$$m_0(\widehat{P}[j]) = -tr_4(\widehat{P}[j] + 2^{-6}) \quad (3)$$
$$m_1(\widehat{P}[j]) = tr_4(\widehat{P}[j] + 2^{-6}) \quad (4)$$
$$m_2(\widehat{P}[j]) = tr_4(3(\widehat{P}[j] + 2^{-6})) \quad (5)$$

と定めている．ここに，$tr_4()$ は小数点以下 4 ビット目までの打切りを示す．また，$j=0$ については，X の上位ビットから桁を定める関数を設計している．

一般に，境界値が満たすべき条件に r^{-j} を含む項がしばしば現れる．この場合，桁選択関数は j に依存し，小さな j について別個に扱わないと関数を設計できない場合がある．ここで，J を，$j \ge J$ で共通の桁選択関数となることを示す指標とすると，J より小さな j については別個に関数を定める必要がある．上の例では，$J=1$ である．

従来の個々の演算のアルゴリズムの提案では，求めた $L_k[j]$ 及び $U_k[j]$ をもとに桁選択関数をパラメータを用いて定義しているが，これらの式中に含まれる変数は演算に依存するため，関数の定義が演算ごとに異なっている．例えば [2] では，新たに定めた変数 $P[j]$ を関数への入力として用いている．そのため，具体的な r や a の値に対する，関数の選択境界値が満たすべき条件の導出については，演算ごと，アルゴリズムの提案ごとに経験により行われており，具体的に条件を導出する一般的な方法はこれまでに示されていない．また，条件を満たすようにパラメータの値を定め，具

体的な関数を決定する際，パラメータにはトレードオフがあるが，選択の境界値の決定に大きな手間が掛かるため，パラメータの値は経験や勘により定めており，得られた関数の最適性は保証されていない．例えば上の例では，t，d，J に関するトレードオフが存在するが，パラメータ t と d の一つの組に対して式 (2)，(3)，(4)，(5) で示したような選択の境界値を求めるために大きな手間が掛かる．以上のことから，複合算術演算の減算シフト型アルゴリズムの設計において，特に，桁選択関数の設計の支援が必要であると考えられる．

3. 最適な桁選択関数を求める体系的手法

本章では，複合算術演算の減算シフト型アルゴリズムの設計における桁選択関数の設計において，最適な関数（トレードオフ曲線上のパラメータの値の組）を求める体系化した方法を提案する．表 1 に本章で現れる主なパラメータなどの一覧を示す．

桁選択回路を実現する際，J より小さな j に対しては $j \geq J$ の場合と別に回路が必要となるため，J の値が小さな関数を設計することが望ましい．また，t と d は桁選択回路への入力ビット数を決めるパラメータであり，回路面積や遅延を考慮すると，t と d のそれぞれが小さな関数を設計することが望ましい．したがって，t，d，J のそれぞれの値が小さな関数が良い実現方法となるが，これらの値にはトレードオフの関係がある．本章では，t，d，J に関するトレードオフ曲線上の値の組に対する桁選択関数を，最適な関数と定義する．

計算したい演算は，演算数を X 及び Y とし，$f(X,Y) = X^{\varsigma}Y^{\eta}$ と表されるものとする．ここに，ζ と η は有理数である．また，演算数 X，Y は正の 2 進小数とし，範囲をそれぞれ $\underline{X} \leq X \leq \overline{X}$，$\underline{Y} \leq Y \leq \overline{Y}$ とする．除算，開平，平方根の逆数計算，立方根計算など，減算シフト型アルゴリズムが提案されている多くの演算がこの形で表される．$f(X,Y)$，$\underline{X}$，$\overline{X}$，$\underline{Y}$，$\overline{Y}$，及び，結果の精度（小数点以下のビット数）N が与えられ，

表 1　主なパラメータなどの一覧
Table 1　A list of major parameters.

r	基数
a	桁集合（$\{-a, -a+1, \cdots, a\}$）の要素の最大値
t	桁選択回路への入力となる $rW[j]$ の小数点以下のビット数
d	桁選択回路への入力となる X の小数点以下のビット数
J	$j \geq J$ で共通の桁選択関数となることを示す指標

$$-r^{-n} \leq f(X,Y) - Z[n] < r^{-n} \tag{6}$$

を満たす結果 $Z[n]$ を r 進で小数点以下 n 桁まで求める減算シフト型アルゴリズムを設計することを考える．ここに，$n = \lceil \frac{N}{\log_2 r} \rceil$ である．

$f(X,Y)$ に対するアルゴリズムの設計において，**2.** で示した設計フローの Step 1 及び Step 2 により，残余 $W[j]$ に関する漸化式や残余が満たすべき条件 $\underline{B}[j] \leq W[j] < \overline{B}[j]$ を r 及び a を含む式で求め，具体的な r 及び a の値に対するアルゴリズムの初期値を定めたものとする．これらの過程で求められる $W[j]$，$\underline{B}[j]$，$\overline{B}[j]$，中間結果 $Z[j]$ の初期値 $Z[i]$（j の初期値を i として，i 及び $Z[i]$ の値を定めたものとする），及び，r の値により定まる n の値を，桁選択関数の設計に用いる．$W[j]$ は $r^j(X^AY^BZ[j]^C - X^DY^EZ[j]^F)$ と表される．ここに，A，B，C，D，E，F は非負の整数である．また，$\underline{B}[j]$ は，求めた式を展開したとき，$\rho = \frac{a}{r-1}$ として

$$\pm A_1 X^{B_1} Y^{C_1} Z[j]^{D_1} \rho^{E_1} r^{-F_1 j} \pm \cdots \pm A_p X^{B_p} Y^{C_p} Z[j]^{D_p} \rho^{E_p} r^{-F_p j}$$

と表される（$\overline{B}[j]$ についても同様である）．ここに，p は $\underline{B}[j]$ の項数，A_l，B_l，C_l，D_l，E_l，F_l（$l = 1, \cdots, p$）は非負の整数である．定める中間結果の初期値 $Z[i]$ は r^{-i} の倍数とし，[2] の基数 4 のアルゴリズムのように演算数の範囲により初期値が異なる場合を考慮して，

$$Z[i] = \begin{cases} Z_1 & \text{if} \quad \underline{X} \leq X \leq \overline{X}' \\ Z_2 & \text{if} \quad \underline{X}' \leq X \leq \overline{X} \end{cases}$$

と表す（初期値が X に依存しない場合は $Z_1 = Z_2$，$\underline{X}' = \underline{X}$，$\overline{X}' = \overline{X}$ とする）．

具体的な r，a の値に対し設計する桁選択関数を，残余 $W[j]$ を桁上げ保存形で表し，

$$q_{j+1} = \begin{cases} -a & \text{if} \quad \widehat{rW}[j] < m_{-a+1}(j, \widehat{X}) \\ k & \text{if} \quad m_k(j, \widehat{X}) \leq \widehat{rW}[j] < m_{k+1}(j, \widehat{X}) \\ & \qquad (-a+1 \leq k \leq a-1) \\ a & \text{if} \quad m_a(j, \widehat{X}) \leq \widehat{rW}[j] \end{cases}$$

と定義する．ここに，$\widehat{rW}[j]$ は $rW[j]$ を小数点以下 t ビットで打ち切った値，$\widehat{X}$ は X を小数点以下 d ビットで打ち切った値である．また，$m_{-a+1}(j, \widehat{X}), \cdots, m_a(j, \widehat{X})$ は桁の選択の境界となる値であり，すべて 2^{-t} の倍数とする．

以下，提案法全体の枠組みについて述べる．パラ

メータ t，d のとり得るすべての値の組に対し，関数が設計可能かどうか，すなわち，とり得るすべての $\{j, \widehat{X}\}$ の値の組に対する $m_{-a+1}(j,\widehat{X}),\cdots,m_a(j,\widehat{X})$ の値が定められるかどうかを調べ，t，d，J に関するトレードオフ曲線上のパラメータの値の組を求める．（調べるべき t と d の値の範囲は，$W[j]$ が満たすべき条件，演算数 X のビット数，X の範囲から定まる．）そのために，

- 演算の定義として与えられる $f(X,Y)$ 及び $\underline{X}$，$\overline{X}$，$\underline{Y}$，$\overline{Y}$ の値
- r 及び a を含む式で表された $W[j]$，$\underline{B}[j]$，$\overline{B}[j]$
- 具体的な r 及び a の値
- r 及び a の値に対し求めた Z_1，Z_2，$\underline{X}'$，$\overline{X}'$，i，n の値
- 関数を設計するためのパラメータ t，d の値

を入力として与え，出力として

- 関数が定められるかどうか
- 関数が定められる場合，とり得るすべての $\{j, \widehat{X}\}$ の値の組に対する $m_{-a+1}(j,\widehat{X}),\cdots,m_a(j,\widehat{X})$ の値，及び，J の値

を求める部分をプログラム化し，種々の複合算術演算に適用できるように一般化する．**3.1** で，この部分の詳細を述べ，**3.2** で，どのようにパラメータ t，d の値を変化させてトレードオフ曲線上のパラメータの値の組を求めるかについて述べる．

3.1 与えた t 及び d の値の組に対する関数の設計

プログラム化する部分は，以下の三つのステップからなる．

（1） $W[j]$，$\underline{B}[j]$，$\overline{B}[j]$ を与え，q_{j+1} として k $(=-a,-a+1,\cdots,a)$ を選んでよい $rW[j]$ の範囲 $L_k[j] \le rW[j] < U_k[j]$ を r 及び a を含む式で求める．

（2）（1）の出力と，$f(X,Y)$，及び，$\underline{X}$，$\overline{X}$，$\underline{Y}$，$\overline{Y}$，r，a，Z_1，Z_2，$\underline{X}'$，$\overline{X}'$，i，n，t，d の値を与え，とり得るすべての $\{j, \widehat{X}\}$ の値の組に対し，境界値 $m_k(j,\widehat{X})$ $(k=-a,\cdots,a)$ が満たすべき条件 $\underline{m}_{k,j,\widehat{X}} \le m_k(j,\widehat{X}) \le \overline{m}_{k,j,\widehat{X}}$ を求める．ここに，$\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ は 2^{-t} の倍数とする．また，得られた条件から，関数が定められるかどうかを判定する．

（3） 関数が定められる場合，（2）の出力と，a，i，n，t の値を与え，とり得るすべての $\{j, \widehat{X}\}$ の値の組に対する境界値 $m_{-a+1}(j,\widehat{X}),\cdots,m_a(j,\widehat{X})$ を定め，J の値を求める．

以下，各ステップについて詳細を述べる．

3.1.1 $L_k[j]$ 及び $U_k[j]$ の導出

$L_k[j]$ 及び $U_k[j]$ は，$W[j+1]$ が満たすべき条件

$$\underline{B}[j+1] \le W[j+1] < \overline{B}[j+1] \tag{7}$$

から求められる．q_{j+1} として k を選ぶとき，中間結果の漸化式 (1) は，$Z[j+1] = Z[j] + kr^{-j-1}$ と表される．$W[j+1]$，$\underline{B}[j+1]$，$\overline{B}[j+1]$ は一般に $Z[j+1]$ を含む式として表されるので，すべての $Z[j+1]$ を $Z[j]+kr^{-j-1}$ と置き換え，$rW[j]$ が満たすべき条件を求めればよい．

$W_k[j]$，$\underline{B}_k[j]$，$\overline{B}_k[j]$ をそれぞれ，式中に存在するすべての $Z[j+1]$ を $Z[j]+kr^{-j-1}$ で置き換えた後の $W[j+1]$，$\underline{B}[j+1]$，$\overline{B}[j+1]$ とすると，式 (7) より，

$$\begin{aligned}\underline{B}_k[j] - W_k[j] + rW[j] &\le rW[j] \\ &< \overline{B}_k[j] - W_k[j] + rW[j]\end{aligned}$$

が得られる．したがって，

$$\begin{aligned}L_k[j] &= \underline{B}_k[j] - W_k[j] + rW[j] \\ U_k[j] &= \overline{B}_k[j] - W_k[j] + rW[j]\end{aligned}$$

となる．

3.1.2 選択境界値が満たすべき条件の導出

まず，j と $\widehat{X}$ の値を固定したときの $\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ の導出について述べ，次に，とり得るすべての $\{j, \widehat{X}\}$ の値の組に対しこれらを求めるアルゴリズムを示す．

（1） ある $\{j, \widehat{X}\}$ の値の組に対する $\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ の導出

除算や開平の設計法 [7] をもとに，ある $\{j, \widehat{X}\}$ の値の組に対する境界値 $m_k(j,\widehat{X})$ $(k=-a+1,\cdots,a)$ が満たすべき条件を求めると，

$$\max_{\widehat{X}}(L_k[j]) \le m_k(j,\widehat{X}) \le \min_{\widehat{X}}(U_{k-1}[j]) - 2^{-t}$$

となり，更に，$m_k(j,\widehat{X})$ が 2^{-t} の倍数であることを考慮すると，

$$\underline{m}_{k,j,\widehat{X}} = 2^{-t}\lceil 2^t \max_{\widehat{X}}(L_k[j])\rceil \tag{8}$$

$$\overline{m}_{k,j,\widehat{X}} = 2^{-t}\lfloor 2^t(\min_{\widehat{X}}(U_{k-1}[j]) - 2^{-t})\rfloor \tag{9}$$

が得られる．ここに，$\max_{\widehat{X}}(L_k[j])$，$\min_{\widehat{X}}(U_{k-1}[j])$ はそれぞれ，X の見積り値が $\widehat{X}$ であるときの $L_k[j]$ の最大値，$U_{k-1}[j]$ の最小値である．$L_k[j]$ 及び $U_{k-1}[j]$ は

一般に X，Y，$Z[j]$ を含む式として表される．したがって，見積り値が $\widehat{X}$ となる演算数 X の範囲 $\underline{x}_{\widehat{X}} \le X \le \overline{x}_{\widehat{X}}$，及び，その X 及び Y（$\underline{Y} \le Y \le \overline{Y}$）に対する中間結果 $Z[j]$ のとり得る値の範囲 $\underline{z}_{j,\widehat{X}} \le Z[j] \le \overline{z}_{j,\widehat{X}}$ を求め，これらの範囲に対する $L_k[j]$，$U_{k-1}[j]$ を計算することにより $\max_{\widehat{X}}(L_k[j])$，$\min_{\widehat{X}}(U_{k-1}[j])$ が得られる．（$\underline{x}_{\widehat{X}}$，$\overline{x}_{\widehat{X}}$，$\underline{z}_{j,\widehat{X}}$，$\overline{z}_{j,\widehat{X}}$ については後で述べる．）

上で示した $\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ の導出の過程において，ある $Z[j]$ の値に対し q_{j+1} として選ばれる値が限定される場合を別個に考えることにより，より小さな $\underline{m}_{k,j,\widehat{X}}$，及び，より大きな $\overline{m}_{k,j,\widehat{X}}$ を求めることができる．これまでの個々の演算のアルゴリズム設計では，このような場合を経験と勘により発見していたが，提案法では，各 j ごとに q_{j+1} として選ばれる可能性のある値を解析することで，厳密な $\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ を求める．以下にその詳細を示す．

ここで，$j = i$ の場合のように，ある j について，$Z[j]$ のとり得る値が一つに定まる場合について考える．このとき，桁 q_{j+1} として選ばれる可能性のある値の集合を $\{\alpha, \cdots, \beta\}$（$-a \le \alpha, \beta \le a$）とすると，桁を正しく選択するためには $\alpha+1 \le k \le \beta$ である k に対する $m_k(j, \widehat{X})$ の値が必要である．また，$-a+1 \le k \le \alpha$ である k については，$m_k(j, \widehat{X})$ を $\overline{m}_{\alpha,j,\widehat{X}}$ 以下の任意の 2^{-t} の倍数とすることができ，$\beta+1 \le k \le a$ である k については，$m_k(j, \widehat{X})$ を $\underline{m}_{\beta+1,j,\widehat{X}}$ 以上の任意の 2^{-t} の倍数とすることができる．したがって，$\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ は以下のように表される．

$$\underline{m}_{k,j,\widehat{X}} = \begin{cases} -\infty & \text{for} \quad -a+1 \le k \le \alpha \\ 2^{-t}\lceil 2^t \max_{\widehat{X}}(L_k[j])\rceil & \text{for} \quad \alpha+1 \le k \le \beta \\ 2^{-t}\lceil 2^t \max_{\widehat{X}}(L_{\beta+1}[j])\rceil & \text{for} \quad \beta+1 \le k \le a \end{cases} \tag{10}$$

$$\overline{m}_{k,j,\widehat{X}} = \begin{cases} 2^{-t}\lfloor 2^t(\min_{\widehat{X}}(U_{\alpha-1}[j]) - 2^{-t})\rfloor & \text{for } -a+1 \le k \le \alpha \\ 2^{-t}\lfloor 2^t(\min_{\widehat{X}}(U_{k-1}[j]) - 2^{-t})\rfloor & \text{for } \alpha+1 \le k \le \beta \\ +\infty & \text{for } \beta+1 \le k \le a \end{cases} \tag{11}$$

α 及び β の値の決定方法を示す．式 (6) を満たす結果を得るためには，$f(X,Y)$ と $Z[j]$ の差が

$$-\rho r^{-j} \le f(X,Y) - Z[j] < \rho r^{-j}$$

を満たさなければならないため，桁 q_{j+1} は，$Z[j+1]$ が

$$f(X,Y) - \rho r^{-j-1} < Z[j+1] \le f(X,Y) + \rho r^{-j-1}$$

を満たすように選ばれる必要がある．したがって，中間結果に関する漸化式 (1) より，$-a$ から a までの k に対し，

$$\min_{\widehat{X}}(f(X,Y)) - \rho r^{-j-1} < Z[j] + kr^{-j-1} \le \max_{\widehat{X}}(f(X,Y)) + \rho r^{-j-1} \tag{12}$$

を満たすならば，q_{j+1} として k が選ばれる可能性がある．式 (12) を満たす最小の k が α の値となり，最大の k が β の値となる．

以上をもとに，ある j に対する $Z[j]$ の範囲が $\underline{z}_{j,\widehat{X}} \le Z[j] \le \overline{z}_{j,\widehat{X}}$ と表される場合における，$\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ の導出方法を示す．$Z[j]$ が r^{-j} の倍数であり，漸化式 (1) の計算により $Z[j+1]$ が $\underline{z}_{j+1,\widehat{X}}$ 以上 $\overline{z}_{j+1,\widehat{X}}$ 以下の r^{-j-1} の倍数となることを考慮すると，特に $Z[j]$ の値が $\underline{z}_{j,\widehat{X}}$ または $\overline{z}_{j,\widehat{X}}$ の場合において，桁 q_{j+1} としてとり得る値が限定される．したがって，$Z[j]$ の範囲を

$$\underline{z}_{j,\widehat{X}} + r^{-j} \le Z[j] \le \overline{z}_{j,\widehat{X}} - r^{-j}$$

としたときの $\underline{m}_{k,j,\widehat{X}}$ の値 $\dot{\underline{m}}_{k,j,\widehat{X}}$，$\overline{m}_{k,j,\widehat{X}}$ の値 $\dot{\overline{m}}_{k,j,\widehat{X}}$ をそれぞれ式 (8)，(9) により求め，$Z[j]$ を $\underline{z}_{j,\widehat{X}}$ としたときの値 $\check{\underline{m}}_{k,j,\widehat{X}}$，$\check{\overline{m}}_{k,j,\widehat{X}}$，及び，$Z[j]$ を $\overline{z}_{j,\widehat{X}}$ としたときの値 $\hat{\underline{m}}_{k,j,\widehat{X}}$，$\hat{\overline{m}}_{k,j,\widehat{X}}$ を式 (10)，(11) により求めると，最終的な $\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ は，

$$\underline{m}_{k,j,\widehat{X}} = \max(\dot{\underline{m}}_{k,j,\widehat{X}}, \check{\underline{m}}_{k,j,\widehat{X}}, \hat{\underline{m}}_{k,j,\widehat{X}}) \tag{13}$$

$$\overline{m}_{k,j,\widehat{X}} = \min(\dot{\overline{m}}_{k,j,\widehat{X}}, \check{\overline{m}}_{k,j,\widehat{X}}, \hat{\overline{m}}_{k,j,\widehat{X}}) \tag{14}$$

となる．

（2） すべての $\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ を求めるアルゴリズム

まず，初期値を Z_1 とする X の範囲 $\underline{X} \le X \le \overline{X}'$ について，$\widehat{X}$ としてとり得る値の集合を求める．これは，$2^{-d}\lfloor 2^d \underline{X}\rfloor$ 以上 $2^{-d}\lfloor 2^d \overline{X}'\rfloor$ 以下の 2^{-d} の倍数となる．次に，各 $\widehat{X}$ に対し，その値が見積り値となる X の範囲 $\underline{x}_{\widehat{X}} \le X \le \overline{x}_{\widehat{X}}$ を求める．$\underline{x}_{\widehat{X}}$，$\overline{x}_{\widehat{X}}$ はそれぞれ，

$$\underline{x}_{\widehat{X}} = \max(\widehat{X}, \underline{X})$$

$$\overline{x}_{\widehat{X}} = \min(\widehat{X} + 2^{-d}, \overline{X}')$$

となる．

これらの値を用い，すべてのとり得る $\widehat{X}$ について，j を i から $n-1$ まで変化させ，$\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ を順に求める．ここで，$\underline{z}_{j+1,\widehat{X}}$ 及び $\overline{z}_{j+1,\widehat{X}}$ は，$Z[j]$ が $\underline{z}_{j,\widehat{X}}$ であるときに q_{j+1} として選ばれる可能性のある桁の最小値を $\check{\alpha}$，$\overline{z}_{j,\widehat{X}}$ であるときに q_{j+1} として選ばれる可能性のある桁の最大値を $\hat{\beta}$ とすると，以下のように表される．

$$\underline{z}_{j+1,\widehat{X}} = \underline{z}_{j,\widehat{X}} + \check{\alpha} r^{-j-1} \quad (15)$$

$$\overline{z}_{j+1,\widehat{X}} = \overline{z}_{j,\widehat{X}} + \hat{\beta} r^{-j-1} \quad (16)$$

したがって，$\underline{z}_{i,\widehat{X}} = \overline{z}_{i,\widehat{X}} = Z_1$ とし，i から $n-1$ までの j に対し式 (13)，(14)，(15)，(16) の計算を行うことにより，すべての j に対する $\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ が求められる．

更に，Z_1 と Z_2 の値が異なる場合は，初期値を Z_2 とする X の範囲 $\underline{X}' \le X \le \overline{X}$ について，上と同様の手順を実行する．

以上により求められたすべての条件について，$\underline{m}_{k,j,\widehat{X}} \le \overline{m}_{k,j,\widehat{X}}$ を満たせば，桁選択関数が定められると判定できる．

3.1.3 具体的な境界値の決定法

すべての k，j，$\widehat{X}$ に対して $\underline{m}_{k,j,\widehat{X}} \le \overline{m}_{k,j,\widehat{X}}$ を満たす $\underline{m}_{k,j,\widehat{X}}$ 及び $\overline{m}_{k,j,\widehat{X}}$ が得られた場合において，具体的な選択境界値 $m_k(j,\widehat{X})$ 及び J の値をどのように決定するかについて述べる．

境界の値が j に対し異なると，j に依存する桁選択関数となり，回路実現の際により多くのハードウェア量を必要とする．したがって，回路実現に適した桁選択関数を得るためには，異なる j に対しできるだけ共通の値となるように $m_k(j,\widehat{X})$ を決定することが重要である．本手法では，各 k 及び $\widehat{X}$ について，以下のようにして i から $n-1$ までの j に対する $m_k(j,\widehat{X})$ を決定する．

まず，$\beta = n-1$ とし，以下の式を満たす最小の α ($\le \beta$) を見つける．(この α が J の値となる．)

$$\max(\underline{m}_{k,\alpha,\widehat{X}}, \underline{m}_{k,\alpha+1,\widehat{X}}, \cdots, \underline{m}_{k,\beta,\widehat{X}}) \le \min(\overline{m}_{k,\alpha,\widehat{X}}, \overline{m}_{k,\alpha+1,\widehat{X}}, \cdots, \overline{m}_{k,\beta,\widehat{X}}) \quad (17)$$

次に，式 (17) の左辺を G，右辺を H とし，α から β までの j について，$[G,H]$ の範囲の 2^{-t} の倍数の中から，できるだけ小さな整数 c により 2^{-c} の倍数として表すことができる数を $m_k(j,\widehat{X})$ の値とする（この値は必ず一つに定まる）．例えば，$t=3$，$G=\frac{9}{8}$，$H=\frac{13}{8}$ である場合は，$\frac{9}{8}$ 以上 $\frac{13}{8}$ 以下の五つの 2^{-3} の倍数のうち，2^{-1} の倍数として表現できる $\frac{12}{8}$ を $m_k(j,\widehat{X})$ として選ぶ．

i から $\alpha-1$ までの j についても，$\beta=\alpha-1$ として以上の手順を繰り返し行うことにより，各 k 及び $\widehat{X}$ に対するすべての $m_k(j,\widehat{X})$ が求められる．

3.2 トレードオフ曲線上のパラメータの値の組の導出方法

t，d，J に関するトレードオフ曲線上のパラメータの値の組を求める方法を示す．

まず，t と d の間のトレードオフについて考える．t，d を，とり得る範囲内の最大値として与え，プログラムの実行により関数が設計できるかどうかを調べる．このとき関数が定められるならば，d を固定し，t を一つずつ小さくして繰り返しプログラムを実行することにより，大きな d に対し関数が設計できる最小の t の値 $t_{\min}$ を求める．更に，t を $t_{\min}$ と固定して，d を小さくして繰り返しプログラムを実行することにより，最小の t に対する d の値 $d_{\max}$ を求める．同様に，パラメータの値を固定する順序を逆にすることで，最小の d の値 $d_{\min}$，及び，その d に対する t の値 $t_{\max}$ を求める．t と d の間のトレードオフ曲線上の値の組は，$t_{\min} \le t \le t_{\max}$，$d_{\min} \le d \le d_{\max}$ の範囲で t と d を変化させ，繰り返しプログラムを実行することにより得られる．

次に，J の値を含めたトレードオフについて考える．ここで，t と d をとり得る範囲内の最大値として与えたときの関数から得られる J の値を $J_{\min}$ とする．上で得られた t と d のトレードオフ曲線上の値の組のそれぞれについて，得られた関数の J の値を調べる．この値が $J_{\min}$ より大きい場合について，d を固定し，t を一つずつ大きくして繰り返しプログラムを実行する．また，同様に，t を固定し，d を一つずつ大きくして繰り返しプログラムを実行する．これらの過程で得られた関数について J の値を調べ，J が小さくなったときの t，d の値の組がトレードオフ曲線上の組となる．

4. 提案手法の適用

3. で述べた手法においてプログラム化する部分を，C 言語によりツールとして構築した．本章では，平方根の逆数計算と立方根計算を例として，提案手法を適用した結果を示す．また，以前提案された桁選択関数と比較し，提案手法の効果を確認する．

表 2　平方根の逆数計算のための選択境界値 $m_k(j, \widehat{X})$（$d = 6, t = 4$）
Table 2　Threshold values $m_k(j, \widehat{X})$ for reciprocal square-root computation. ($d = 6, t = 4$)

$\widehat{X} \times 64$	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$m_{-1}(\widehat{X}) \times 16$	-24	-26	-26	-28	-28	-28	-30	-31	-32	*$-36, -32$	-32	-32	-32	-32	-32	-36
$m_0(\widehat{X}) \times 16$	-8	-8	-8	-8	-8	-8	-8	-8	-8	-8	-12	-12	-12	-12	-12	-12
$m_1(\widehat{X}) \times 16$	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
$m_2(\widehat{X}) \times 16$	24	24	24	26	28	28	28	28	28	28	30	32	32	32	32	32
$\widehat{X} \times 64$	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$m_{-1}(\widehat{X}) \times 16$	-36	-36	-36	-36	-36	-40	-40	-40	-40	-40	-40	-40	-40	-40	-40	-40
$m_0(\widehat{X}) \times 16$	-12	-12	-12	-12	-12	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16
$m_1(\widehat{X}) \times 16$	8	8	10	10	12	12	12	12	12	12	12	12	12	12	12	12
$m_2(\widehat{X}) \times 16$	*28, 32	32	32	34	34	36	36	40	40	40	40	40	40	40	40	40
$\widehat{X} \times 64$	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$m_{-1}(\widehat{X}) \times 16$	-44	-40	-40	-40	-44	-48	-48	-48	-48	-48	-48	-48	-48	-48	-48	-48
$m_0(\widehat{X}) \times 16$	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16	-16
$m_1(\widehat{X}) \times 16$	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
$m_2(\widehat{X}) \times 16$	40	40	40	42	44	44	44	44	48	48	44	48	48	48	48	48

*：$j = 0$, $j \geq 1$

4.1　平方根の逆数計算

[2] と同様に，浮動小数点数の平方根の逆数計算の仮数部の計算，すなわち，$2^{-2} < X < 1$ である X に対し，$f(X) = X^{-\frac{1}{2}}$ を求める計算について考える．結果の小数点以下のビット数 N は 23（単精度）とする．$\underline{X}$，$\overline{X}$ はそれぞれ，$\underline{X} = 2^{-2} + 2^{-25}$，$\overline{X} = 1 - 2^{-24}$ である．

2. で述べたように，この計算のための減算シフト型アルゴリズムにおける残余 $W[j]$，残余の範囲の下限 $\underline{B}[j]$ 及び上限 $\overline{B}[j]$ は，それぞれ

$$
\begin{aligned}
W[j] &= r^j(1 - XZ[j]^2) \\
\underline{B}[j] &= -2XZ[j]\rho + X\rho^2 r^{-j} \\
\overline{B}[j] &= 2XZ[j]\rho + X\rho^2 r^{-j}
\end{aligned}
$$

と表される．ここで，基数を 2，桁集合を $\{-1, 0, 1\}$ としたとき（中間結果の初期値は $Z[0] = 1$），及び，基数を 4，桁集合を $\{-2, -1, 0, 1, 2\}$ としたとき（中間結果の初期値は $X \geq 2^{-1}$ ならば $Z[0] = 1$，それ以外ならば $Z[0] = 2$）の桁選択関数を提案手法の適用により求めた．以下に詳細を示す．

まず，基数が 2 の場合について述べる．ツールへの入力として，上で述べた $f(X)$，$\underline{X}$，$\overline{X}$，$W[j]$，$\underline{B}[j]$，$\overline{B}[j]$，及び，$r = 2$，$a = 1$，$Z_1 = 1$，$Z_2 = 1$，$\underline{X}' = 2^{-2} + 2^{-25}$，$\overline{X}' = 1 - 2^{-24}$，$i = 0$，$n = 23$ を与え，パラメータ t，d を変化させてツールを実行することにより，桁選択関数が設計できる最小の t，d の値，及び，そのときの境界値を求めた．その結果，$t = 1$ としたときに，$j \geq 0$ で共通の境界値（すなわち $J = 0$）として $m_0 = -2^{-1}$，$m_1 = 0$ が得られた．（d については任意となり，X に依存しない関数が得られた．）[2] で示されている桁選択関数は $t = 2$ であるため，桁選択回路への入力ビット数が 1 少ない実現方法が存在することが分かった．

次に，基数が 4 の場合について述べる．ツールへの入力として，$f(X)$，$\underline{X}$，$\overline{X}$，$W[j]$，$\underline{B}[j]$，$\overline{B}[j]$，及び，$r = 4$，$a = 2$，$Z_1 = 2$，$Z_2 = 1$，$\underline{X}' = 2^{-1}$，$\overline{X}' = 2^{-1} - 2^{-25}$，$i = 0$，$n = 12$ を与え，パラメータ t，d を変化させてツールを実行することにより，桁選択関数が設計できる最小の t，d の値，及び，そのときの境界値を求めた．その結果，以下に示す，d，t，J に関するトレードオフ曲線上の値の組が得られた．

- $d = 5$，$t = 5$，$J = 2$
- $d = 5$，$t = 9$，$J = 1$
- $d = 6$，$t = 4$，$J = 1$
- $d = 6$，$t = 5$，$J = 0$
- $d = 7$，$t = 3$，$J = 2$
- $d = 7$，$t = 4$，$J = 0$
- $d = 9$，$t = 3$，$J = 1$

実際に基数を 4 とする回路を設計する際には，これらのパラメータの値に対する実現方法を比較することで，用途に応じてより適した実現方法を選択できる．このうち，$d = 6$，$t = 4$ としたときに得られた選択境界値を表 2 に示す．例えば，$X = \frac{51}{128}$ のとき，$\widehat{X} = \frac{25}{64}$ となり，$m_{-1}(\widehat{X})$，$m_0(\widehat{X})$，$m_1(\widehat{X})$，$m_2(\widehat{X})$ はそれぞれ，$j = 0$ の場合は $-\frac{36}{16}$，$-\frac{8}{16}$，$\frac{8}{16}$，$\frac{28}{16}$ とすればよく，$j \geq 1$ の場合は $-\frac{32}{16}$，$-\frac{8}{16}$，$\frac{8}{16}$，$\frac{28}{16}$ とすればよ

い．この表から，$m_0(\widehat{X})$ 及び $m_1(\widehat{X})$ はすべての j に対し共通の値となることが分かる．また，$m_{-1}(\widehat{X})$ 及び $m_2(\widehat{X})$ は $\widehat{X}$ の値がそれぞれ $\frac{25}{64}$，$\frac{32}{64}$ の場合のみ，$j=0$ とそれ以外で異なる境界値が必要となることが分かる．

このパラメータの値による関数と，[2] で示されている関数の，桁選択回路への入力ビット数の総和を比較する．上で示した関数は，$\frac{1}{4}<X<1$ より，X の上位 6 ビットから境界値を決定する．[2] では，d，t，J の値は同じであるが，X と $Z[j]$ の積 $P[j]$ の上位 7 ビットから境界値を決定している．したがって，提案手法の適用により，入力ビット数が 1 少ない実現方法が存在することが分かった．

4.2 立方根計算

[5] と同様に，浮動小数点数の立方根計算の仮数部の計算，すなわち，$2^{-3}\le X<1$ である X に対し，$f(X)=X^{\frac{1}{3}}$ を求める計算について考える．結果の小数点以下のビット数 N は 24（単精度）とする．（結果の範囲が平方根の逆数計算と異なるため，N の値も異なる．）$\underline{X}$，$\overline{X}$ はそれぞれ，$\underline{X}=2^{-3}$，$\overline{X}=1-2^{-24}$ であり，$W[j]$，$\underline{B}[j]$，$\overline{B}[j]$ は，それぞれ

$$
\begin{aligned}
W[j] &= r^j(X-Z[j]^3)\\
\underline{B}[j] &= -3Z[j]^2\rho+3Z[j]\rho^2r^{-j}-\rho^3r^{-2j}\\
\overline{B}[j] &= 3Z[j]^2\rho+3Z[j]\rho^2r^{-j}+\rho^3r^{-2j}
\end{aligned}
$$

と表される．

基数を 2，桁集合を $\{-1,0,1\}$ としたとき（中間結果の初期値は $Z[1]=2^{-1}$）の桁選択関数について，平方根の逆数計算の場合と同様にして，関数が設計できる最小の t，d の値，及び，そのときの境界値を求めた．その結果，$t=1$ としたときに，$j\ge 1$ で共通の境界値（すなわち $J=1$）として $m_0=-2^{-1}$，$m_1=0$ が得られた（d については任意となり，X に依存しない関数が得られた）．[5] で示されている桁選択関数は $t=1$ であり，境界値も上で示した関数と同一である．したがって，提案手法の適用により，最適な関数として以前提案された関数と同じものが得られた．

5. むすび

複合算術演算の減算シフト型アルゴリズムの設計において大きな手間が掛かる桁選択関数の設計について，最適な桁選択関数を求める方法を提案した．この手法の適用により，桁選択回路への入力ビット数の総和に関するトレードオフ曲線上の桁選択関数を設計でき，特に，平方根の逆数計算のアルゴリズムについて，従来提案したものよりもよい桁選択関数を求めることができた．提案手法により，減算シフト型アルゴリズムに基づく複合算術演算の専用回路全体のハードウェア量や計算時間などを見積ることができ，プロセッサ設計者が専用回路を搭載するかどうかを短期間で決定できるようになるものと考えられる．

謝辞 本研究を進めるにあたり，多大な助言を頂いた名古屋大学の高木一義准教授に心より感謝致します．

文献

[1] N. Takagi, “A hardware algorithm for computing reciprocal square root,” Proc. 15th IEEE Symposium on Computer Arithmetic, pp.94–100, June 2001.

[2] N. Takagi, D. Matsuoka, and K. Takagi, “Digit-recurrence algorithm for computing reciprocal square-root,” IEICE Trans. Fundamentals, vol.E86-A, no.1, pp.221–228, Jan. 2003.

[3] T. Lang and E. Antelo, “Correctly rounded reciprocal square-root by digit recurrence and radix-4 implementation,” Proc. 15th IEEE Symposium on Computer Arithmetic, pp.83–93, June 2001.

[4] T. Lang and E. Antelo, “Radix-4 reciprocal square-root and its combination with division and square root,” IEEE Trans. Comput., vol.52, no.9, pp.1100–1114, Sept. 2003.

[5] N. Takagi, “A digit-recurrence algorithm for cube rooting,” IEICE Trans. Fundamentals, vol.E84-A, no.5, pp.1309–1314, May 2001.

[6] 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.

[7] M.D. Ercegovac and T. Lang, Division and Square Root — Digit-Recurrence Algorithms and Implementations, Kluwer Academic, 1994.

[8] M.D. Ercegovac and T. Lang, Digital Arithmetic, Morgan Kaufmann, 2004.

（平成 20 年 3 月 31 日受付，5 月 28 日再受付）

熊澤 文雄 （正員）

平 13 名大・工・電気電子情報工卒．平 15 同大大学院修士課程了．平 19 同大学院博士課程満期退学．現在同大大学院在学中．算術演算回路，ハードウェアアルゴリズムに関する研究に従事．

高木　直史　（正員）

昭 56 京大・工・情報工卒．昭 58 同大大学院修士課程了．昭 63 同大工博．昭 59 同大・工・情報工助手．平 3 同助教授．平 6 名大・工・情報工助教授．平 10 同教授．算術演算回路，ハードウェアアルゴリズム，論理設計などの研究に従事．平 7 日本 IBM 科学賞，情報処理学会坂井記念特別賞，平 17 文部科学大臣表彰など受賞．