社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－ARC－165 (7)
2005／12／1

ディジットシリアル浮動小数点演算器を用いた高精度FFT回路の設計検討

鈴木圭介† 浅生宗隆† 窪田昌史††
谷川一哉†† 弘中哲夫††

本稿では，演算器ベースの再構成デバイスを目指し，従来のパラレル浮動小数点演算器よりも集積度を高くすることのできるディジットシリアル浮動小数点演算器を提案する．ディジットシリアル浮動小数点演算器の初期評価として，最もきびしい条件のディジット幅1としたディジットシリアル浮動小数点演算器を使ったFFT回路とパラレル浮動小数点演算器を使ったFFT回路を面積当たりのFlopsで比較した．1ビット幅ではパラレル浮動小数点演算器の方がよい性能であったが，ディジット幅を2以上にした場合にはディジットシリアル浮動小数点演算器の性能がよくなる可能性がある．

Consideration of high precision FFT logic with digit-serial floating point unit

Keisuke Suzuki,† Munetaka Asao,† Atsushi Kubota,††
Kazuya Tanigawa†† and Tetsuo Hironaka††

In this paper, we propose a digit-serial floating point unit(FPU) to realize a reconfigurable device based on function units. This paper presents a digit-serial FPU can be implemented in a smaller area than a traditional parallel FPU. We compare a Flops per area of a FFT logic with one bit-serial FPUs to a FFT logic with the parallel FPUs. Though the parallel FPUs is better than one bit-serial FPUs, the digit-serial FPUs might be better than the parallel FPUs.

1. はじめに

近年，半導体の微細加工技術の進展に伴い，FPGAなどの再構成デバイスの利用範囲が広がっている.[1] 特に，ここ数年では，浮動小数点を用いた科学技術計算分野での応用が盛んである．その応用として，PCと比較して数倍から数十倍の性能を出す例も報告されている.[2][3] 例えば佐々木らの研究では倍精度浮動小数点演算による3次元FFTをFPGA上に実装し，100MHzで動作させ500MFLOPSの性能を得ている.[4] このFPGAを4チップ搭載したボードは8ノードのPCクラスタで同じ処理した場合と比較して，およそ1.5倍の性能に達する．このような数値演算分野での応用では，複数の演算器をパイプライン動作させて性能を高めている．したがって再構成デバイスによる数値計算において高い性能を得るための重要な要因の1つとして，集積する演算器数が挙げられる．実際，先の佐々木らの研究では実装できる演算器数の制限によって性能が1/2となっており，演算器数の増強による性能向上を期待している．またHemmertらの研究でのFPGAによるFFTの実装においても，今後さらに性能を向上させるためには多くの演算器を搭載して並列処理するパイプラインを増やす必要があるとしている.[5]

しかし，現在の再構成デバイスでは高精度の浮動小数点演算器を多数実装するのは難しい．これは再構成デバイスにおいて高精度の浮動小数点演算器を多数実装する際に，ロジックブロックの不足や配線資源の不足，動作周波数の低下といった問題を招くためである．特に浮動小数点演算器は演算精度に対し指数関数的にハードウェア量が増加する．これによりロジックブロックが不足するため，実装できる演算器数が制限される．配線資源の不足については，演算精度に比例して演算器間のバス配線が多くなるため配線制約が厳しくなり，実装できる演算器数が制限される．また演算精度に応じてバス幅が大きくなるので，論理ゲート遅延や配線遅延が増加して動作周波数が低下する．

そこで本稿では，演算器ベースの再構成デバイスを目指し，ベースとなる演算器の1つとしてディジットシリアル浮動小数点演算器を提案する．これはディジットシリアル浮動小数点演算器によって演算器の集積度を上げ，演算性能を高めるねらいである．ディジットシリアル浮動小数点演算器はパラレル浮動小数点演算

† 広島市立大学大学院 情報科学研究科
Graduate School of Information Sciences, Hiroshima City University
†† 広島市立大学 情報科学部
Hiroshima City University

器に比べて小面積で実装できる可能性があり，単位面積あたりに実装できる演算器数の制限を緩和できる．また，演算精度によらず演算器間の配線量を抑えられ，少ない配線資源で演算器を集積できると考えられる．同様に入力数の少ない論理と短い配線によって遅延を削減でき，高い動作周波数で動作できると予想される．また，演算精度を拡張しても回路面積の増加を線形に抑えられ，高精度の数値演算に対しても演算器の集積度を維持できる．

本稿の目的は，ディジットシリアル浮動小数点演算器の提案とパラレル浮動小数点演算器に対する優位性の検討である．パラレル浮動小数点演算器とディジットシリアル浮動小数点演算器でアプリケーションを実装し，両者の性能を比較する．実装するアプリケーションとしては，大気シミュレーションや流体解析の計算の中でも負荷の高い処理とされ，多くの DSP や数値計算にとっても利用価値の高い FFT を選んだ．性能の比較では，実装した FFT 演算ユニットの面積と動作周波数を用いる．ディジットシリアル浮動小数点演算器とパラレル浮動小数点演算器とでは実行サイクル数が違うため各々の Flops を性能とし，面積あたりの Flops を指標とした．その際，演算精度の拡張によるディジットシリアル浮動小数点演算器とパラレル浮動小数点演算器の特徴を示すため，1 ワード 32，64，128 ビットの浮動小数演算器で構成し，比較検討する．

本稿の節構成を以下に示す．2 節ではディジットシリアル浮動小数点演算器の詳細を示す．3 節ではディジットシリアル浮動小数点演算器，パラレル浮動小数点演算器で構成した FFT 演算ユニットをそれぞれ示す．4 節でこれら FFT 演算ユニットを使って演算精度ごとに面積当たりの Flops を比較し，5 節でまとめる．

今回，本稿では初期評価のため 1bit 幅のディジットシリアル浮動小数点のみ設計を行っているが，本来我々の研究はデータ幅 1bit だけでなく 2bit 以上のディジットシリアル浮動小数点演算器を対象にしている．

2. ディジットシリアル浮動小数点演算器の構成

まず，簡単にディジットシリアル演算器を紹介する．また，ディジットシリアル浮動小数点演算器に使われている主な構成ユニットについて述べる．その後，ディジットシリアル浮動小数点加算器，乗算器について述べる．

2.1 ディジットシリアル演算器

整数演算器単体で比較するとディジットシリアル演算器はパラレル演算器よりも高い単位面積あたりのスループットを発揮する結果が得られている[8]．また，漸化式などの反復演算において，誤差の蓄積を避けるために高精度演算が必要になるが，そのような高精度演算を行うにはディジットシリアル演算器は有効である．

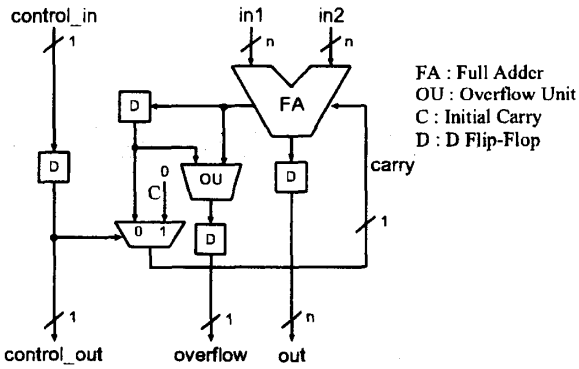

図 1 ディジットシリアル加算器

n ビットのディジットシリアル演算器は，1 ワード W ビットの入力を n ビットごとに処理し，W/n クロックかけて結果を出力する．ディジットシリアル加算器の回路図を図 1 に示し，図を例にとり説明する．コントロール信号はパルスで入力され，その入力タイミングは 2 回存在する．1 回目はデータの LSB が入力される 1 サイクル前，そして 2 回目はデータの MSB が入力された時である．このコントロール信号でキャリーをクリアしている．図 1 のように加算器ではキャリーをフィードバックすることで複数サイクルかけて実行する．

1 節でも述べたが，パラレル演算器では倍の精度にすることで回路面積が倍以上に増加する．それに対しディジットシリアル演算器は倍の精度を演算するのに倍のサイクル数をかけるだけで演算ができる．同じ面積のチップ上にそれらを集積した場合ディジットシリアル演算器の方が性能がよいことがわかる．

単純には整数演算器と同様の効果がえられないかもしれないが，浮動小数点演算器を構成している主なユニットは整数演算器であることを考えるとディジットシリアル演算器で構成する浮動小数点演算器は高精度化に向いていると考えられる．

では，そのディジットシリアル浮動小数点演算器を構成する主なユニットについて述べる．主な構成ユニットは加算器，シフタ，バッファ，乗算器である．加算器については前述したので残りのユニットについて簡単に説明する．

- シフタ
  シフタの構成図を図 2 に示す．シフタはディレイを挟みタイミングをずらすことでシフトさせる．
- バッファ
  バッファの構成図を図 3 に示す．バッファはデータの処理タイミングを合わせるために使われる．例えば，浮動小数点演算で指数部と仮数部では，指数部の方が処理にかかるサイクル数が少ないため一時的にバッファリングし，仮数部の処理のサイクル数分待つことでタイミングをあわせる．
- 整数乗算器

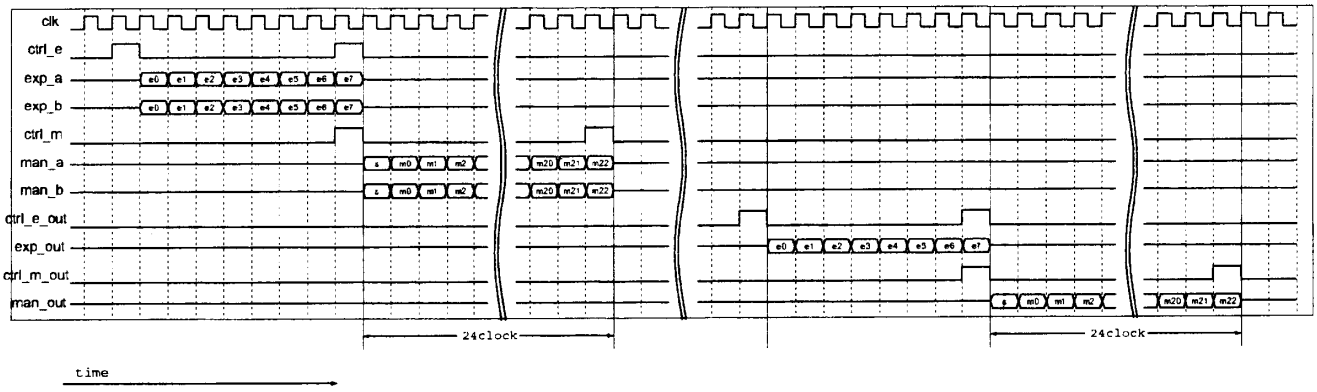

図 4 32 ビット浮動小数点演算のデータ入出力

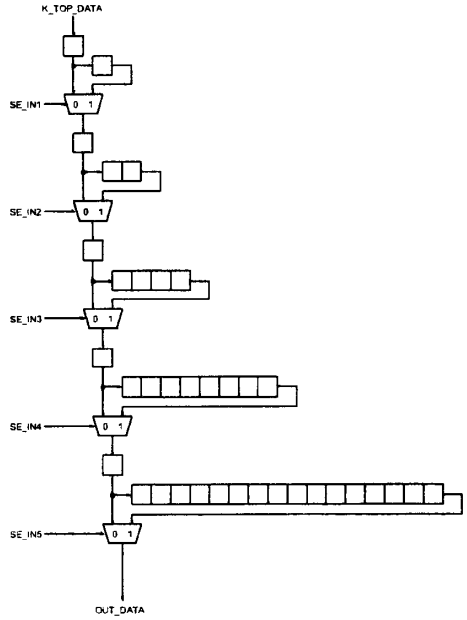

図 2 シフタ

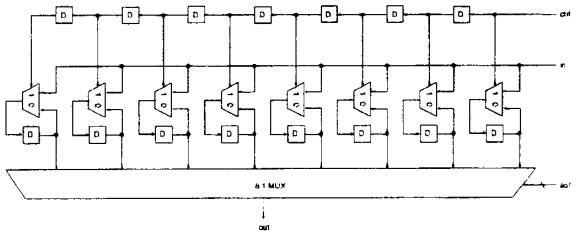

図 3 バッファ

lyon[6] の 2 の補数乗算器を符号無し乗算器にしたものを利用している．データはシリアル-シリアル入力，シリアル出力である．以下の漸化式をハードウェアで構成している．

$S_{(0)} = 0,$

$S_{(i+1)} = (S_{(i)} + X \cdot y_i)2^{-1}$

$(i = 0, 1, \cdots N-2)$

被乗数：$X = (x_{N-1}, x_{N-2}, ..., x_0)$

乗数：$Y = (y_{N-1}, y_{N-2}, ..., y_0)$

次にこれらを使って実際に設計したディジットシリアル浮動小数点演算器について述べる．

2.2 ディジットシリアル浮動小数点演算器の入出力データ形式

本研究で設計したディジットシリアル浮動小数点演算器について述べる．データフォーマットは IEEE754 形式にしている．

まずディジットシリアル演算器のデータ入出力について述べる．単精度の場合の入出力タイミングを図 4 に示す．図は左から時間がながれている様子を示している．つまり指数部 (exp_a,exp_b) の LSB から入力されている．図 4 のようにデータは指数部 (exp_a,exp_b) と仮数部 (man_a,man_b) は別々の入力線に入力される．ただし符号ビット (s) は仮数部の LSB へ付加される．データは LSB から順に入力され，演算後 LSB から出力される．そして，データの入力時にはコントロール信号も同時に入力される．コントロール信号はパルスで入力され，そのタイミングは 2 回ある．1 回目はデータの LSB が入力される 1 サイクル前，そして 2 回目はデータの MSB が入力された時である．1 回目が LSB 入力の 1 タイミング前である理由は，入力先の回路がデータの入力されるタイミングを事前に判断できることである．そして，1 ビット幅のデータを入力することができるためである．例えば，LSB の 1 サイクル前ではなく LSB 入力と同じサイクルで 1 ビット幅のデータを入力しようとすると，MSB のコントロール信号が重なってしまう．

次にディジットシリアル浮動小数点加算器・乗算器の構成について述べる．各々の構成図を図 5，6 に示す．図のデータ線のように，データの流れは指数部と仮数部があり，別々に処理される．

浮動小数点加算器は主に指数部比較，指数部正規化加算，仮数部交換，桁あわせ，加算，正規化，丸め，

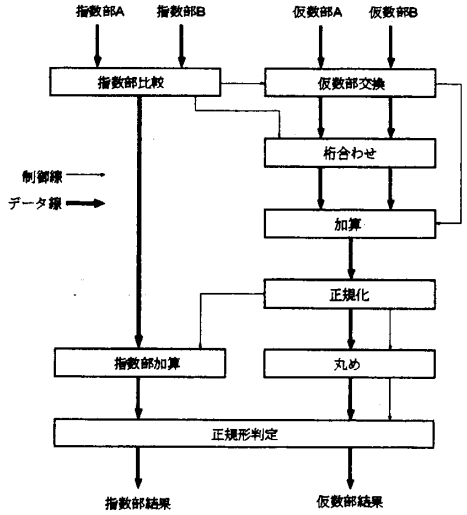

図 5 ディジットシリアル浮動小数点加算器のデータパス

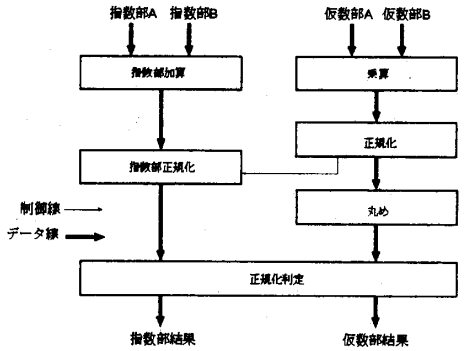

図 6 ディジットシリアル浮動小数点乗算器のデータパス

正規形判定のステージで構成される．そして，浮動小数点乗算器は指数部加算，指数部正規化加算，乗算，正規化，丸め，正規形判定のステージで構成される．

2.3 ディジットシリアル浮動小数点加算器

図 5 のディジットシリアル浮動小数点加算器を構成するステージの詳細を以下に示す。

- 指数部比較ステージ
  入力された指数部同士を比較し，値の大きい指数部を選択する．選択された指数部は次の指数部加算ステージへ出力される．また，比較結果によって仮数部のデータフローを交換するため，仮数部交換ステージへ制御信号を出力する．また，比較によって減算された結果はシフト量として桁合わせステージへ出力される．
- 指数部加算ステージ
  正規化ステージから仮数部がシフトされた結果を受け取り，そのシフト量を指数部へ加算する．
- 仮数部交換ステージ
  指数部比較ステージで判定された大小関係に従ってデータフローを交換する．交換したデータを桁合わせステージへ出力する．
- 桁合わせステージ
  指数部比較ステージから指数部の差を受け取り，その分シフトさせる．シフトさせたデータを加算ステージへ出力する．
- 加算ステージ
  桁合わせステージからデータを受け取り，加算する．加算した結果を正規化ステージへ出力する．
- 正規化ステージ
  加算ステージからデータを受け取り，いったんデータをバッファへ入力する．データをシフトさせる必要がある場合は，シフトさせながらデータを丸めステージへ出力する．また，丸め発生の有無を丸めステージへ出力する．シフト量を指数部に加算させるため，指数部正規化ステージへシフト量を出力する．
- 丸めステージ
  正規化ステージから正規化された値を受け取る．また，加算し丸めた値を正規形判定ステージへ出力する．
- 正規形判定ステージ
  正規形であるかどうかの判定し正規形へ直して指数部，仮数部を出力する．

2.4 ディジットシリアル浮動小数点乗算器

図 6 のディジットシリアル浮動小数点乗算器を構成するステージの詳細を以下に示す．

- 指数部加算ステージ
  入力された指数部同士を加算し指数部加算ステージへ出力する．
- 指数部正規化ステージ
  指数部比較ステージから指数部の値を受け取り，正規化ステージから入力されるシフト量が入力されるタイミングをあわせるためバッファリングする．正規化ステージから入力されるシフト量を加算し，正規化判定ステージへ出力する．
- 乗算ステージ
  入力された仮数部を乗算する．乗算した結果の下位は丸めて正規化ステージへ出力される．
- 正規化ステージ
  加算器の正規化ステージに同じ．
- 丸めステージ
  加算器の丸めステージに同じ．
- 正規形判定ステージ
  加算器の正規形判定ステージに同じ．

3. ディジットシリアル浮動小数点演算器を使った FFT 回路の構成

本研究で利用する FFT の構成を図 7 に示す．使用する FFT アルゴリズムは基数 2 の Culey-Tukey アルゴリズムである．

図 7 のバタフライ演算を 1 セルとして，1 チップ内

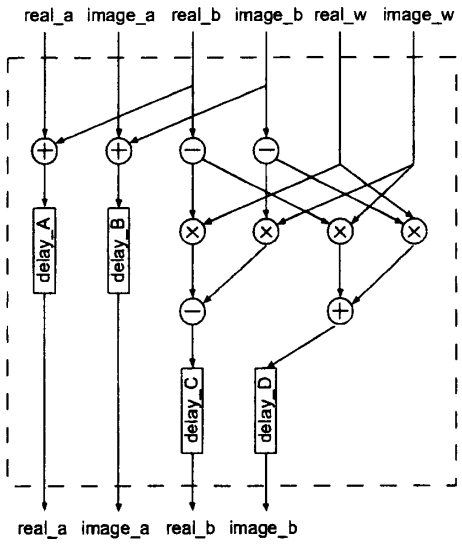

図 7 ディジットシリアル FFT 回路の構成

でセル同士をつなぎ合わせることで FFT 回路を実現する．そのため入力と出力のタイミングを合わせるためのディレイ回路を挿入している．

図のノード a(real_a,image_a),b(real_b,image_b) からはそれぞれの実部と虚部のデータが入力され，w(real_w,image_w) からはあらかじめ RAM にテーブル化された回転子のデータが入力されることを想定している．また，タイミング調整用にディレイ回路 delay_A〜D をはさんでいる．これは，ディジットシリアル浮動小数点加算器の演算後の出力タイミングは入力データによって違うためである．ただし複数セルを設計した場合，非常に多くの合成時間がかかるため，本研究では 1 セルのみ設計した．

3.1 パラレル浮動小数点演算器を使った FFT 回路の構成

FFT 回路を構成するパラレル浮動小数点演算器は IEEE745 準拠で 4 段パイプラインである．浮動小数点加算器の加算回路には Carry Save Adder を用いた．また浮動小数点乗算器の乗算回路には ARITH 言語をベースにした乗算器ジェネレータ[7] を利用し生成した以下の 3 つの乗算器を用いた．

- 部分積生成器:PPG with Radix-4 modified Booth recoding
- 部分積加算器:Wallace tree
- 最終段加算器:Carry Look-ahead Adder

そして FFT 回路は図 7 と同様の構成である．ただし，出力タイミングを合わせるためのディレイ回路 delay_C，delay_D はなく，delay_A，delay_B はパイプラインを 8 段はさんでいる．

表 1 面積と動作周波数

	面積 (μm^2)	動作周波数 (MHz)	サイクル数 (サイクル)	面積当たりのスループット ($Flops/\mu m^2$)
32 ビットパラレル	2757766	101	1	37
64 ビットパラレル	7826004	99	1	13
32 ビットディジット	869550	427	80	6.1
64 ビットディジット	1709536	205	170	0.71
128 ビットディジット	2712609	76	354	0.079

4. 面積当たりの Flops の評価

ディジットシリアル浮動小数点演算器で構成した FFT 回路とパラレル浮動小数点演算器で構成した FFT 回路の面積当たりの Flops を比較した．ここで面積当たりの Flops は

面積当たりの Flops = ((動作周波数 / サイクル数) ×8) / 面積

で算出している．動作周波数は合成ツールの Synopsys 社 Design Compiler を使って合成結果から得た．そして配置配線ツールの Synopsys 社 Apollo を用いて配置配線を行い，面積を測定した．またサイクル数はパイプライン的に動作させた場合の Flops で算出した．つまりパラレル浮動小数点演算器は完全にパイプライン動作するのでサイクル数は 1 である．それに対し，ディジットシリアル浮動小数点演算器は精度によって異なり，実行するのに必要なサイクル数は精度 32bit で 80 サイクル，64bit で 170 サイクル，128 ビットで 354 サイクルである．ここでサイクル数が精度と等しくならないのは，途中にバッファが含まれているからである．例えば精度 32bit の FFT 回路では，前者のデータが占有する最大 24 ビットのバッファが搭載されている．そのため，後者のデータが入力可能になるまでに，バッファへ入力させるためのサイクル数 + バッファから出力させるためのサイクル数がオーバヘッドとなる．よってスループットは 32+24×2 サイクルとなる．64bit,128bit でも同様に計算している．

ディジットシリアル FFT 回路とパラレル FFT 回路の動作周波数，面積を測定した結果と，それらの面積当たりのスループットを表 1 へ示す．ただし，128bit のパラレル FFT 回路の結果はシミュレーション時間の都合で掲載することができなかった．

4.1 考　　察

表 1 の結果から，1 ビットのディジットシリアル浮動小数点演算器の優位性は言えなかった．しかし以下

の点でまだ改善の余地があると言える.

(1) 完全なパイプライン化
(2) 2ビット以上のディジット化
(3) SIMD化

今回の設計ではタイミングを調整するためにバッファを挿入していた. しかしこのことでパイプライン動作を止める結果となり, スループットを低下させていた. そこで, ハードウェア量が増えるが, (1)の改善方法として完全にパイプライン動作させることを考える. スループット = データ幅にすることで, 例えば32ビットディジットシリアルFFT回路で現在の性能の80/32=2.5倍となる. また, (1)の改善方法によって動作周波数をより高くすることが見込める. なぜなら, バッファを無くすことでクリティカルパスであった図3の8:1MUX部分がなくなるためである. それ以外の部分のディレイは精度をあげてもほぼ同じであるため高精度化による動作周波数の落ちこみはなくなる.

また, (2)の改善方法として, 今回の設計では1ビットのディジットシリアル浮動小数点演算器であったが, 4ビットのそれにするとスループットは4倍になる. 例えば, 1ビットのディジットシリアルの演算では1ワード32ビットの演算を32/1=32サイクルに分けて演算するが, 4ビットのディジットシリアルの演算では32/4=8サイクルで演算が終わる. つまり(1)のパイプライン化方法と合わせるとスループットは10倍となる.

最後に, (3)の改善方法として, コントロール回路部分はデータA, Bと同じタイミングをとっていくためA, Bのフローに必要な数と同じ数のフリップフロップを使用している. よってコントロール回路部分は他の演算器と共有しSIMD化することで面積を約2/3に減らすことができる.

以上の3点を改善することで概算ではあるが面積当たりのスループットは約10倍 ($1/1.5 \times 10 \times 3/2 = 10$) 向上しパラレル浮動小数点演算器の性能を上回ることができると考えられる.

5. むすび

我々はパラレル浮動小数点演算器と比較し, 小面積で実装可能なディジットシリアル浮動小数点演算器の提案を行った. しかし, 1ビット幅のディジットシリアル浮動小数点演算器の面積当たりのスループットはパラレル浮動小数点演算器のそれと比べて20%ほどであった. そこで, 改善項目を挙げ検討した結果, パラレル浮動小数点演算器に勝る性能を得られる考えにいたった. 今後はそれら改善項目を念頭にディジットシリアル浮動小数点演算器の設計を進める.

6. 謝辞

本研究は東京大学大規模集積システム設計教育研究センターを通し, ケイデンス株式会社, シノプシス株式会社の協力で行われたものである. また浮動小数点加減算器, 乗算器の設計データを提供してくださった京都大学越智裕之助教授, 乗算器の設計データを提供してくださった東北大学本間尚文助手に深く感謝いたします.

参考文献

1) 天野英晴. "(招待論文) 最近のリコンフィギャラブルシステム, 動的リコンフィギャラブルシステムの動向", 電子情報通信学会技術研究報告 SR2005-5, Vol.105, No.36, pp.31-36, 2005年5月.
2) 濱田 剛, 中里 直人, "FPGAによる数値計算の高速化", FPGA/PLD Design Conference Session 10, 2005
3) G. Govindu, L. Zhuo, S. Choi, and V. K. Prasanna. "Analysis of High-Performance Floating-Point Arithmetic on FPGAs. ", In Proceedings of the 11th Reconfigurable Architectures Workshop, New Mexico, USA, April 2004.
4) 佐々木徹, 溝口大介, 長嶋雲兵, "Car-Parrinello計算向け三次元FFTロジックの開発", 情報処理学会論文誌 Vol.45 No.SIG 11, pp.313-320,2004
5) Hemmert, Scott K. and Underwood, Keith D. "An Analysis of the Double-Precision Floating-Point FFT on FPGAs",FCCM, April, 2005
6) R.E.Lyon,"Two's Complement Pipeline Multipliers", IEEE Trans. Computers,1976
7) http://www.aoki.ecei.tohoku.ac.jp/arith/mg/index.html
8) K. Suzuki, M. Asao, K. Nakanishi, K. Tanigawa, T. Hironaka, "Evaluation of the n Bit-Serial Arithmetic Units in Consideration of Trade-off between Area and Performance", ITC-CSCC2005 Proc. Vol.3, pp.943-944, 2005