システムＬＳＩ設計技術　99－16
（2001．1．12）

digitシリアル演算を用いたDSPシステム設計最適化の一手法

渡辺 義治　武内 良典　北嶋 暁　今井 正治

大阪大学 大学院基礎工学研究科 情報数理系専攻
〒560-8531 大阪府豊中市待兼山町 1-3
TEL: 06-6850-6626
FAX: 06-6850-6627
E-mail: peasv@vlsilab.ics.es.osaka-u.ac.jp

あらまし　本稿ではdigitシリアル演算を用いて，スループット制約を満たし，最小ハードウェアコストでDSPシステム構成を実現するための一手法を提案する．本稿での提案手法では，DSPシステム内での演算単位であるdigitサイズを効果的に変更することでハードウェアコスト・スループットのトレードオフを考慮した設計を可能とする．評価実験より，digitサイズを変化させることによりハードウェアコスト・スループット間にはトレードオフが存在することが確認され，スループット制約条件下において最適構成の選択が可能であることが確認された．

キーワード　digitシリアル演算，ディジタル信号処理，ハードウェアコスト最小化，digitサイズ変換

Design Optimization Method Using Digit Serial Operation for DSP Systems

Yoshiharu Watanabe　Yoshinori Takeuchi　Akira Kitajima　Masaharu Imai

Department of Informatics and Mathematical Science
Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama-cho, Toyonaka
Osaka, 560-8531 Japan
TEL: +81 6 6850 6626
FAX: +81 6 6850 6627
E-mail: peasv@vlsilab.ics.es.osaka-u.ac.jp

Abstract

This paper studies DSP system architecture optimization method using digit serial operation under the throughput constraint. The proposed method can make the DSP system design considering the trade-offs between the hardware cost and the throughput which are introduced by efficient digit size conversions. Experimental results show the trade-offs between the hardware cost and the throughput and the optimum architecture of DSP system can be decided using proposal digit size conversion algorithm under the throughput constraint.

Key words　Digit Serial Operation, Digital Signal Processing, Hardware Cost Minimization, Digit Size Conversion

1 はじめに

ディジタル信号処理(DSP)においては，画像処理や音声処理，データ通信処理等の様々なアプリケーションが存在する．しかし，様々なアプリケーションにおいて要求されるスループットは大きく異なる．例えば，画像処理の分野における処理では数十MHzから数百MHzといった高いスループットを要求するのに対し，音声処理の分野における処理では数KHzから数十KHzといった低いスループットで処理を行なうことが可能である．対象とするアプリケーションにより要求されるスループットが大きく異なるためハードウェア設計者はアプリケーションを考慮した演算器構成を考える必要がある[1]-[3]．現在，一般的な演算法としてビットパラレル演算とビットシリアル演算が存在する．ビットパラレル演算は入力されたワードの全ビットに対し一度に演算を行なうため，高いスループットを得ることが可能であるが実装に要するハードウェアコストは高くなる．一方，ビットシリアル演算は，ワードの入力は1ビットずつ行ない，1ビットに対しての処理を逐次的に行ない全体としてワードに対する演算を行なう演算法である．ビットシリアル演算では処理の単位が1ビットであるので実装に要するハードウェアコストを低く抑えることが可能だが，得ることのできるスループットは低くなる．一般に高いスループットを必要とするアプリケーションに対してはビットパラレル演算が用いられ，低いスループットで処理を行なうことが可能なアプリケーションに対してはビットシリアル演算が用いられることが多い[4][5]．

ところでビットパラレル演算とビットシリアル演算ではハードウェアコスト・スループットにおいて大きく性質が異なる．アプリケーションにより要求範囲の広いDSPにおいて，ビットパラレル演算とビットシリアル演算という両演算法のみでは，要求されるスループット制約に対して柔軟に対応することが困難となる[6]．そこで，本稿ではビットパラレル演算とビットシリアル演算と呼ばれる両演算法の中間的な演算法として，digitシリアル演算に焦点を当てた．ビットシリアル演算ではスループット制約を満たすことができないが，ビットパラレル演算では必要以上のスループットを得てしまい，ハードウェアコストが高くなるという状況において，digitシリアル演算は効果的な演算法である．

本稿では，digitシリアル演算において，digitサイズの調整により同一の演算に対してハードウェアコスト・スループット間のトレードオフを得ることが可能である[7]という特徴に注目したDSPシステム設計最適化の一手法を示す．具体的には，DSPシステム内の処理を複数の部分に分割し，スループット制約を満たすように各部分での処理単位であるdigitサイズの調整を行なう．ただし，隣接する二つの部分が異なるdigitサイズで処理を行なう際にはdigitサイズを変換する必要がある．本稿では，効率的にDSPシステムを複数の部分に分割し，必要に応じて隣接する二つの部分間でdigitサイズの変換を行なうためのアルゴリズムについて示す．

以下，本稿の構成について述べる．2章においてdigitシリアル演算，本稿で対象とするシステムについての仮定および本稿で用いる用語の定義を示す．3章では提案する最適化手法を示し，最適化問題の定式化を行なう．次に，4章において本稿で提案するDSPシステム分割・digitサイズ変換回路挿入のアルゴリズムについて示す．そして，5章で本稿での最適化手法を用いた評価実験・考察を行ない，6章で本稿のまとめと今後の課題について述べる．

2 諸定義

本章では，digitシリアル演算，本稿で用いる用語の定義および，最適化の対象とするシステムにおける仮定を示す．

2.1 digitシリアル演算

図1にdigitシリアル演算を示す．digitシリアル演算は1ワードを等ビット長のdigitに分割し，digitに対し逐次的に演算を行なうことでワード演算を行なう演算法である．1digitのビット数をdigitサイズと呼ぶ．digitサイズは1ビットから1ワードのビット数まで変化させることが可能である．digit分割されたワードに対して，最上位digitをMSD(Most Significant Digit)，最下位digitをLSD(Least Significant Digit)と呼ぶ．また，digitシリアル演算器の入出力のビット幅は入力・出力されるdigitのサイズと一致しており，本稿ではdigitシリアル演算器の入力・出力のビット幅を入力形式・出力形式と呼ぶ．

2.2 対象システムに対する仮定

本稿で最適化の対象とするシステムに対して，以下3つの仮定を置くこととする．

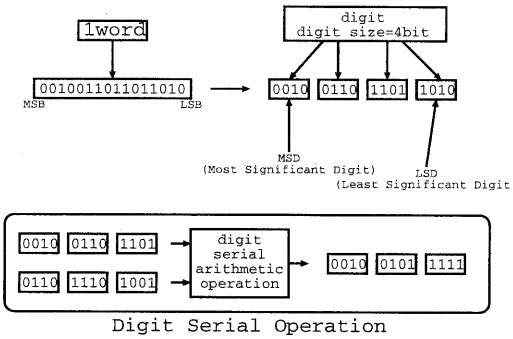

図 1: digit シリアル演算

1. システムは加算器・乗算器，および遅延回路から構成されるシステムに限定する．
2. 1 クロックサイクルで処理される digit の個数は，処理される digit サイズに関係なく，常に 1 個とする．
3. システムに対する，入力形式と出力形式はあらかじめ与えられているものとする．

2.3 用語の定義

- ハードウェアコスト
  システム X のハードウェアコストを $A(X)$ と表す．$A(X)$ は式 (1) より得られる．

$$A(X) = a(add(X)) + a(mul(X)) + a(del(X)) + a(conv(X)) \quad (1)$$

  ここで，各記号の意味は次の通りである．

  $add(X)$: システム構成 X での加算器の集合
  $mul(X)$: システム構成 X での乗算器の集合
  $del(X)$: システム構成 X での遅延回路の集合
  $conv(X)$: システム構成 X での digit サイズ変換回路の集合
  $a(Y)$: 構成 Y のハードウェアコスト

- 実行サイクル数
  入力データの LSD がシステムへ入力されてから，入力データに対する出力結果の MSD がシステムから出力されるまでに要するクロックサイクル数．以降，システム X の実行サイクル数を $Cycle(X)$ と表す．

- 演算時間
  システムが入力データに対して処理結果を出力するまでに要する時間．システム X の演算時間，$T(X)$ は式 (2) で定義される．

$$T(X) = Cycle(X) \times Delay(X) \quad (2)$$

  なお，$Delay(X)$ はシステム X における最大遅延時間を示す．

- スループット
  システムが単位時間内に処理可能な量．システム X のスループット，$ThroughP(X)$ は式 (3) で定義される．

$$ThroughP(X) = \frac{1}{T(X)} \quad (3)$$

  式 (3) より，スループットの単位は時間の逆数となり，単位時間を 1s とした場合 Hz である．

3 提案手法

本稿で提案する最適化手法は次のとおりである．

> システム内部の処理を複数の digit サイズを併用して行なうことでハードウェアコストやスループットに関しての設計空間を拡張させ，制約条件下での最適構成を選択する．

以降で提案手法についての詳細と，最適化問題の定式化について示す．

3.1 digit サイズ変換によるシステムへの影響

システム内のある点で digit サイズの変更を行なうと，その点以降のシステムの処理は変更された digit サイズを用いた処理となる．したがって，単一の digit サイズを用いたシステム設計では得ることのできなかったシステム構成を得ることが可能となる．すなわち，ハードウェアコスト・スループットに関しての設計空間の拡張を行なえる．

例として，図 2 のシステムを考える．図 2 のシステムは入力が $n$bit の digit シリアル形式，出力が $m$bit の digit シリアル形式である．図 2 のシステムは内部が 2 つのプロセスで構成されているものとする．図 2 中のプロセス 1，プロセス 2 が digit サイズ $w$bit で処理を行なっている時，要するハードウェアコスト，演算時間をそれぞれ $A_{p1}(w)$，$A_{p2}(w)$，$T_{p1}(w)$，$T_{p2}(w)$

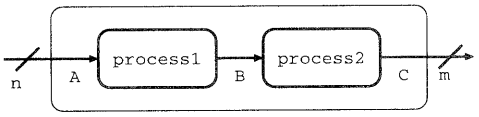

図 2: digit シリアルシステムのモデル

変換点	ハードウェアコスト	スループット
A 点	$A_{p1}(m) + A_{p2}(m)$	$\frac{1}{T_{p1}(m) + T_{p2}(m)}$
B 点	$A_{p1}(n) + A_{p2}(m)$	$\frac{1}{T_{p1}(n) + T_{p2}(m)}$
C 点	$A_{p1}(n) + A_{p2}(n)$	$\frac{1}{T_{p1}(n) + T_{p2}(n)}$

表 1: 変換タイミングの違いによるシステムのハードウェアコスト，スループット

	小 $\Longrightarrow$ 大		
ハードウェアコスト	C 点	B 点	A 点
スループット	C 点	B 点	A 点

表 2: 変換タイミングの違いによるハードウェアコスト，スループットの大小関係

とする．図 2 中で，システムの入力形式と出力形式は異なっているので，システム内のある点で digit サイズの変換を行なう必要がある．図 2 の例では簡単のため，複数の変換は考えず一度の変換で $n$bit から $m$bit への変換を行なうこととする．図 2 中の A, B, C の点でそれぞれ digit サイズの変換を行なった場合のシステムのハードウェアコスト，スループットをそれぞれ表 1 に示す．

$n < m$ の時，プロセス 1，プロセス 2 のハードウェアコスト，演算時間にはそれぞれ $A_{p1}(n) < A_{p1}(m)$, $A_{p2}(n) < A_{p2}(m)$, $T_{p1}(n) > T_{p1}(m)$, $T_{p2}(n) > T_{p2}(m)$ の関係が成り立つ．したがって，表 1 よりハードウェアコスト，スループットに関して表 2 の結果が得られる．表 2 より，digit サイズ変換のタイミングの違いにより，同一の処理行なうシステムに対してハードウェアコスト・スループットのトレードオフが存在することが確認できる．

上記の例で示されるように，システム内での digit サイズ変換タイミングを効果的に変化させることで，ハードウェアコスト・スループット間のトレードオフを考慮した設計が可能となる．

3.2 最適化問題の定式化

最適化問題は，性能最適化問題，ハードウェアコスト最小化問題，消費電力最小化問題の 3 つの種類に分類することができる．本稿では性能を制約条件としたハードウェアコスト最小化問題を最適化の対象とする．本稿のハードウェアコスト最小化問題は，式 (4) で示されるスループット制約条件のもとで，式 (5) で示される目的関数，ハードウェアコストの値を最小とするようなシステム構成 X を求める最適化問題として定式化できる．

$$A(X) = a(add(X)) + a(mul(X)) + a(del(X)) + a(cont(X)) \quad (4)$$

$$ThroughP(X) \geq ThroughP_{min} \quad (5)$$

$ThroughP_{min}$: スループット制約条件

4 変換回路挿入アルゴリズム

4.1 変換回路挿入位置決定の必要性

システム内の任意の位置に変換回路が挿入可能であると仮定した場合，次のような問題が生じる

1. 膨大な数の変換回路挿入位置の組合せの存在:
   システム内の全ての演算器間に変換回路が挿入可能であるので演算器間の結線の総数が N 本であった場合，変換回路挿入位置の組合せは 2^N となり指数的に増加する．

2. ハードウェアコスト，スループット共に悪影響を受ける変換回路挿入位置の存在:
   変換回路を挿入すると，システム全体のハードウェアコスト，スループットには変換回路分のオーバヘッドが生じる．変換回路を挿入することでシステム全体としてのトレードオフを得るためには，変換回路によるオーバヘッド分を打ち消すようなハードウェアコストの減少，あるいはスループットの向上が必要となる．しかし，digit サイズ変換によるシステムのハードウェアコストやスループットの変化が変換回路のオーバヘッドを越えない場合は，システム全体としてハードウェアコストが増加するにもかかわらず，スループットが低下してしまう．

システム内の任意の位置に変換回路が挿入可能であるとすると，上記 2 点のような問題が生じてしまう．

すなわち，システム内の任意の位置に変換回路の挿入を行ない設計空間の拡張を行なうと，膨大な数のシステム構成が存在するが多くの設計がハードウェアコスト・スループットのトレードオフを得ることができない設計となってしまう．以上のことより，システム内での効果的な変換回路挿入位置を決定する必要がある．

4.2 変換回路挿入アルゴリズム

本稿で提案する変換回路挿入アルゴリズムではシステム内の乗算器に注目をする．乗算器に注目する理由は次の通りである．

- システム内で digit サイズを変換した場合，加算器や遅延回路と比較すると乗算器はシステム全体のハードウェアコストやスループットに与える影響が大きい．
- 乗算器はその構造により演算時間が異なる．演算時間の異なる複数の乗算器より得られる乗算結果を同時に処理する必要がある場合，演算時間の早い乗算器より得られる乗算結果は，演算時間の遅い乗算器からの演算結果を待つ間バッファリングされなければならない．バッファリングによる待ち時間が多く生じてしまうと，システム全体のスループットの低下を引き起こす．

提案する変換回路挿入アルゴリズムは以下の 5 つのステップを経る．

step1 ： システム内より加算器，遅延回路を削除し乗算器のみで構成される道を作成する
step2 ： 各乗算器に対して入力からの深さを決定する
step3 ： 同じ深さを持つ乗算器をグループ化し，システム全体としての深さを決定する
step4 ： システム内で深さが変化する点を抽出する
step5 ： 変換回路挿入点の組合せを求める

1. 変換回路挿入アルゴリズム:*step1*
   step1 では，システム内より加算器及び遅延回路を削除する．*step1* により，システムより乗算器のみで構成される道を得ることができる．
2. 変換回路挿入アルゴリズム:*step2*
   step2 では，*step1* で得られた道に含まれる各乗算器に対して乗算器の深さを定義する．なお，道に含まれる乗算器 A の深さを $Depth(A)$ とすると，$Depth(A)$ は式 (6) で定義される．

$$Depth(A) = Pass(A)+1 \quad (6)$$

（ただし，フィードバックループに含まれる乗算器に対してはループを含まずに決定した $Depth(A)$ とする）

$Pass(A)$ ： システムの入力から乗算器 A に至るまでに通過した乗算器の数

式 (6) のように道に含まれる乗算器に対し深さを定義し，同一の深さを持つ乗算器を，同じ演算時間で演算可能な乗算器を用いてシステムを構成することで演算時間の違いによる演算結果のバッファリングを避けることができ，結果としてシステムのスループットの低下を避けることができる．

3. 変換回路挿入アルゴリズム:*step3*
   step3 では，*step2* で深さを決定した各乗算器に対して，同じ深さを持つ乗算器をグループ化しシステム全体としての深さを決定する．ただし，システムの入力は深さ 0 を，システムの出力は全乗算器中の深さの最大値に 1 を加えたものとする．
4. 変換回路挿入アルゴリズム:*step4*
   step4 では，*step3* で決定したシステム全体の深さにおいて，システム内で深さの変化する点を抽出する．システム内で深さの変化する点とは *step3* でシステム全体としての深さをグループ化した境界線と，システムを構成する演算器間の結線との交点と等しくなる．*step4* で抽出される点が変換回路挿入点となる．
5. 変換回路挿入アルゴリズム:*step5*
   step5 では，*step4* で決定された変換回路挿入点についての組合せを考える．

以上，システムに対して *step1* から *step5* までの処理を行なうことでシステム内に数多く存在する変換回路挿入位置の中から，乗算器に注目した変換回路挿入位置のみを抽出し，膨大な量の変換回路挿入点の組合せから，効果的な変換回路挿入点の組合せを得ることができる．例として，変換回路挿入アルゴリズムの各ステップを図 3 に示される 1 次 IIR(Infinite Impulse Response) フィルタに適応した時の処理手順を図 4 に示す．

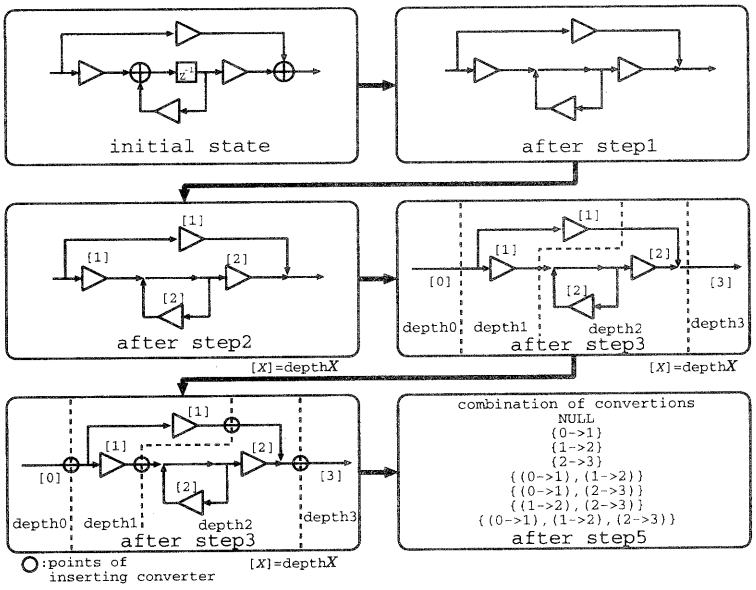

図 4: 一次 IIR フィルタに対する変換回路挿入アルゴリズムの適用

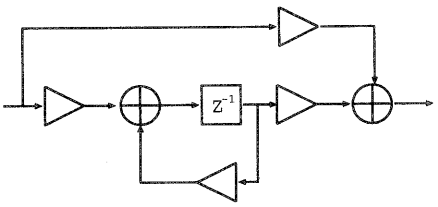

図 3: 一次 IIR フィルタ

5 評価実験

5.1 評価実験の目的

提案手法の有効性を調べるために評価実験を行なった．実験の目的は次の 2 点である．

- digit シリアル演算を用いることで，DSP システム設計においてハードウェアコスト・スループットの設計空間が拡張されることを確認する
- DSP システム内での効果的な digit サイズの変換により，ハードウェアコスト・スループットの設計空間が拡張されることを確認する

5.2 評価実験の方法

図 5 に示される，二次の IIR フィルタに対して 3 章，4 章で提案を行なった手法を適用して，制約条件下での最適化を行なう．表 3 に評価実験で対象とする IIR フィルタの *word* 長，入力形式，出力形式を示す．

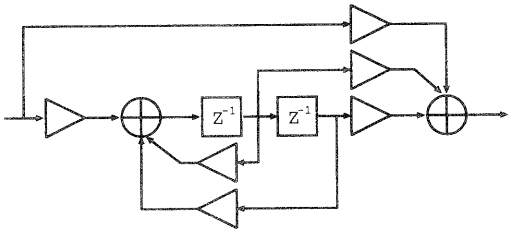

図 5: 二次 IIR フィルタ

	word 長	入力形式	出力形式
実験 1	16bit	8bit digit シリアル	4bit digit シリアル
実験 2	8bit	8bit パラレル	8bit パラレル

表 3: 評価対象の二次 IIR フィルタの種類

5.3 使用する digit シリアル乗算器

今回，IIR フィルタ内で使用する digit シリアル乗算器の構造として次の 2 種類を構造を採用した．

1. 単数内部乗算器型 digit シリアル乗算器 [8]
   digit シリアル乗算器内部に単数のビットパラレル乗算器を有する構造をとる．単数のビットパラレル乗算器を繰り返し用いて digit シリアル乗算を行なうのでスループットは低くなるが，低ハードウェアコストで実装可能である．

2. 複数内部乗算器型 digit シリアル乗算器 [1][2]
   digit シリアル乗算器内部に複数のビットパラレル乗算器を有する構造をとる．複数のビットパラレル乗算器を並列に用いて digit シリアル演算を行なうので，高いスループットを得ることができるが，高ハードウェアコストとなる．

また，各 digit シリアル乗算器内部で用いるビットパラレル乗算器には CSA 型の array 乗算器と 4 段にパイプライン化された CSA 型の array 乗算器を用いることとする．digit シリアル乗算器の構造と内部乗算器の種類の組合せで合計 4 種類の digit シリアル乗算器が実現されている．

5.4 実験結果

5.4.1 実験 1

図 6 に実験 1 より得られた，ハードウェアコスト・スループットの関係を示す．横軸にハードウェアコスト，縦軸にスループットをとる．図 6 より，digit サイズ変換位置の違いによりハードウェアコスト・スループットの設計空間が拡張されていることが確認される．図 6 中の点 *opt_arc* はスループット制約として 6MHz 以上という制約を与えた時に選択される最適構成となっている．点 *opt_arc* は図 7 の構成となる．

5.4.2 実験 2

図 8 に実験 2 より得られた，ハードウェアコスト・スループットの関係を示す．横軸にハードウェアコスト，縦軸にスループットをとる．図 8 より，実験 1 の場合と同様に digit サイズ変換位置の違いによりハードウェアコスト・スループットの設計空間が拡張されていることが確認される．図 8 中の点 *opt_arc* はスループット制約として 10MHz 以上という制約を与え

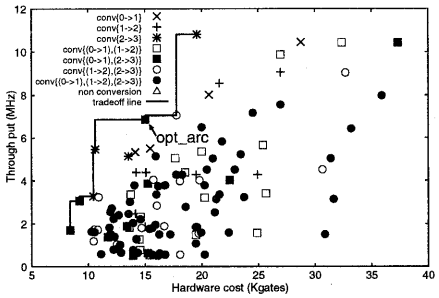

図 6: 実験 1 でのハードウェアコスト・スループットの関係

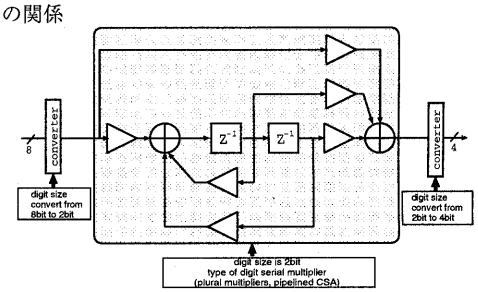

図 7: 実験 1 より得られる IIR フィルタの最適構成 (スループット制約: 6MHz 以上)

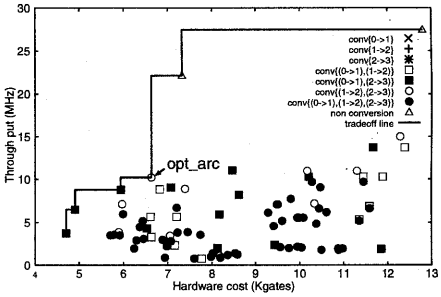

図 8: 実験 2 でのハードウェアコスト・スループットの関係

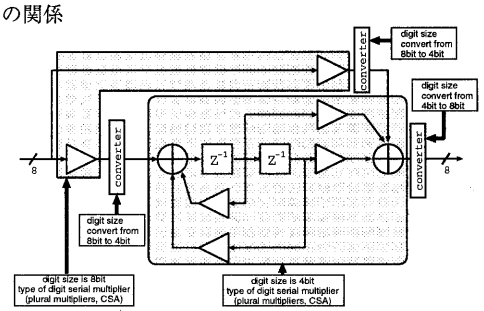

図 9: 実験 2 より得られる IIR フィルタの最適構成 (スループット制約: 10MHz 以上)

た時に選択される最適構成となっている．点 *opt_arc* は図9の構成となる．

5.5 実験結果の考察

まず，実験1の結果について考察する．実験1では入力形式と出力形式が異なるIIRフィルタを考えた．入力形式と出力形式が異なるのでIIRフィルタ内で一度はdigitサイズの変換をする必要がある．しかし，今回の実験結果では二度のdigitサイズ変換を行なっている．これは，中間的なdigitサイズを用いることでdigitサイズ変換のオーバヘッドよりも，システム全体としてハードウェアコスト，スループットを改善することができるためである．すなわち，効果的に複数のdigitサイズを用いてシステム処理を行なうことで単一のdigitサイズを用いた処理では存在しなかった設計空間を得ることができる．

次に，実験2の結果について考察する．実験2ではシステムの *word* 長と入力形式，出力形式が一致する様なビットパラレル演算型のIIRフィルタを考えた．図8の結果からわかるように，ビットパラレル演算型のIIRフィルタを変換回路を用いずに構成する，すなわちビットパラレル演算器を用いて構成した場合に高スループットを得ることができる．しかし，ビットパラレル演算器を用いるということでハードウェアコストが高くなっていることも図8の結果から確認できる．しかし今回の実験では，制約条件下における最適構成はdigitシリアル演算を用いた構成となっている．すなわち，今回の実験結果よりビットパラレル演算形式のシステムに対しても，与えられる制約条件次第ではビットパラレル，digitシリアル間での変換を行ないシステム内部はdigitシリアル演算で処理を行なうというシステム構成の方が，従来通りビットパラレル演算で処理を行なうシステム構成よりも最適な構成となるということが確認できる．

6 おわりに

本稿ではdigitシリアル演算を用いたDSPシステム設計最適化手法を提案した．評価実験を行ない，digitシリアル演算を用いたシステム設計により，ハードウェアコスト・スループット間にトレードオフが存在することが確認され，スループット制約条件下で最小ハードウェアとなるシステム構成が選択可能であることを示した．また，従来行なわれているビットパラレル処理を行なうシステムに対しても本手法が有効であることを評価実験により示した．

今後の課題としては，変換回路挿入位置決定後にトレードオフの候補となるシステム構成を効果的に選択する手段に関して検討する必要がある．

謝辞

本研究を進めるにあたり，貴重なコメントを頂いた大阪大学VLSIシステム設計研究室の諸氏に深謝する．なお，本研究の一部は(株)半導体理工学研究センターとの共同研究による．

参考文献

[1] A. E. Bashagha and M. K. Ibrahim, "Radix digit-serial pipelined divider/square-root architecture," Inst. Elec. Eng. Proc. Comput. Digit. Tech., Vol. 141, no.6, pp.375-380, Nov. 1994.

[2] Yun-Nan Chang, Janardhan H. Satyanarayan and Keshab K. Parhi, "Systematic Design OF High-Speed and Low-Power Digit-Serial Multipliers," IEEE Trans. Circuit and System-II, vol.45, no.12, pp.1585-1595, 1998.

[3] Yun-Nan Chang, Janardhan H. Satyanarayan and Keshab K. Parhi, "Low-power digit-serial multipliers," in Proc. IEEE Int. Symp. Circuits and Systems, pp.1023-1026, Hong Kong, June 1997.

[4] R. Hatley and P. Corbett, "Digit-serial processing techniques," IEEE Trans. Circuit and System, vol. 37, pp. 707-710, June 1990.

[5] Vishwas M. Rao, Behrouz Nowrouzian, "A novel approach to the design and hardware implementation of high-speed digit-serial modified-booth digital multipliers," Int. Symp. Circuits and Systems, pp.1952-1955, Hong Kong, June 1997.

[6] A. Aggoun, M. K. Ibrahim, and A. Ashur, "Bit-Level Pipelined Digit-Serial Array Processor," IEEE Trans. Circuit and System-II, vol.45, no.7, pp.857-868 , 1998.

[7] Yoshiharu Watanabe, Yoshinori Takeuchi, Akira Kitajima, Masaharu Imai, "Area and Performance trade-off analysis of digit serial adder and multiplier" in Proc. SASIMI2000, pp209-216, April 2000.

[8] Yun-Nan Chang, Janardhan H. Satyanarayan and Keshab K. Parhi, "Low-power digit-serial multipliers," in Proc. IEEE Int. Symp. Circuits and Systems, pp.1023-1026, Hong Kong, June 1997.

[9] M. K. Ibrahim, "Radix-2^n multiplier structures: A structured design methodology" IEE Proc., Vol.140, PartE, no.4, pp.185-190, July 1993.

盛光印刷所