計測自動制御学会東北支部 第215回研究集会 (2004.5.27)
資料番号 215-2

FPGAを用いた状態空間ディジタルフィルタ用高性能プロセッサの実現

Implementation of High-Performance Processor for State-Space Digital Filters Using FPGA

○鎌田直人*，野崎剛*，恒川佳隆*

○Naoto Kamata*, Takeshi Nozaki*, Yoshitaka Tsunekawa*

*岩手大学大学院 工学研究科

*Faculty of Engineering, Iwate University

キーワード： 状態空間ディジタルフィルタ(State-Space Digital Filters), 分散演算 (Distributed Arithmetic), FPGA(Field Programable Gate Array), 高精度(High Accuracy)

連絡先： 〒020-8551 盛岡市上田4–3–5 岩手大学 工学部
鎌田直人，Tel.: (019)621-6468，Fax.: (019)621-6468，E-mail: t2303007@iwate-u.ac.jp

1. まえがき

ディジタルフィルタはディジタル信号処理の最も基本的かつ重要な処理技術であり，計測・制御，情報・通信，音響など様々な分野で中心的な役割を果たしている．特に制御の分野では，ディジタルフィルタや制御装置の設計が状態方程式を用いて行われている．状態方程式で表されるディジタルフィルタを状態空間ディジタルフィルタ（State-Space Digital Filters 以下，SSDF）と呼ぶ．ここではSSDFを，これまでの伝達関数表現による狭義のディジタルフィルタばかりでなく，カルマンフィルタ，オブザーバ，レギュレータなどを含む広義のディジタルフィルタとして扱う．

ディジタルフィルタの設計においては，与えられた仕様を満足するフィルタ係数を求めること（近似）だけではなく，量子化誤差が小さなフィルタ構造を決定すること（合成）が必要である．ディジタルフィルタを状態方程式で記述すれば，システム理論的観点から近似と合成を統一的に行うことが可能である．その結果，近似誤差と量子化誤差が最小なSSDFの設計が可能となる[1)]．しかし，この方法で設計されたSSDFは係数行列が密となるため，出力当たりの計算量が増加する．

そこで我々はSSDFの高精度性に着目して，分散演算を適用した高性能VLSIアーキテクチャを提案してきた[3)]．SSDFにおいては量子化誤差に対して最適合成が可能となるため，実現の際には必要語長を他の構造に対して最小化できる．すなわち，計算時間が語長のみに依存する分散演算を用いた本実現法は，高次の場合でも非常に高い処理速度が実現可能となる．

一方，FPGA(Field Programmable Gate Array)は，ユーザにより回路仕様の設定，変更が可能な

VLSIで，数千〜100万ゲート程度を収容できるものが市販されており音声処理，画像処理などに利用され始めている．現在FPGAは大容量化が進み，IP(Intellectual Property)によるシステム設計が行われており，ディジタルフィルタではIIR，FIRフィルタのファンクションが用意されている．IIRフィルタでは一般的に2次セクションに基づいた縦続形，並列形の構成が用いられている．またIIRフィルタでは高いスループットと急峻な遮断特性に加え，さらに高精度性が求められる．SSDFは縦続形や並列形では実現不可能な，リミットサイクルが発生せず，係数量子化誤差，丸め誤差が最小である高精度なディジタルフィルタリングが実現可能となるため，SSDFの機能をシステムに埋めこめられればシステム実現にとって有益となる．

本稿では，分散演算に基づくSSDFをVHDLにより設計し，FPGAにおいて性能評価を行う．本提案法では分散演算を用いたシリアル処理を行うことで，乗算器を用いた構成より高速で，ハードウェア量が削減されることを明らかにし，SFA(Serial Full Adder)による分散演算を用いたSSDFのアーキテクチャを提案する．それによってキャリー伝搬遅延を短縮し，高速処理が実現できることを明らかにする．

2. 状態空間ディジタルフィルタ

ディジタルフィルタの伝達関数がn次で与えられたとき，これに対応してSSDFの状態方程式は次式で示される．

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\,\boldsymbol{x}(k) + \boldsymbol{B}\,\boldsymbol{u}(k) \tag{1}$$

$$\boldsymbol{y}(k) = \boldsymbol{C}\,\boldsymbol{x}(k) + \boldsymbol{D}\,\boldsymbol{u}(k) \tag{2}$$

ここで，$\boldsymbol{x}(k)$はn次の状態ベクトル，$\boldsymbol{u}(k)$はr次の入力ベクトル，$\boldsymbol{y}(k)$はm次の出力ベクトルである．また,$\boldsymbol{A}$，$\boldsymbol{B}$，$\boldsymbol{C}$，$\boldsymbol{D}$はそれぞれ$n\times n$，$n\times r$，$m\times n$，$m\times r$の実係数行列である．SSDFの伝達関数は(1)，(2)式をz変換することによって次のように得られる．

$$H(z) = \boldsymbol{C}(z\,\boldsymbol{I} - \boldsymbol{A})^{-1}\,\boldsymbol{B} + \boldsymbol{D} \tag{3}$$

SSDFでは，状態ベクトル$\boldsymbol{x}(k)$がシステムの遅延要素の出力を表し，行列$\boldsymbol{A}$，$\boldsymbol{B}$，$\boldsymbol{C}$，$\boldsymbol{D}$の各要素がシステムの係数に対応するとき，$n\times n$の任意の正則行列$\boldsymbol{T}$に対して，$\boldsymbol{x'} = \boldsymbol{T}^{-1}\,\boldsymbol{x}$と状態ベクトルの変換を行う．状態空間ディジタルフィルタでは，適当な等価変換行列$\boldsymbol{T}$を選ぶことにより，リミットサイクルが発生せず，係数量子化誤差，丸め誤差が最小になる最適合成が可能となる．その結果，仕様に対して必要語長を最小化できる．

したがって，状態空間ディジタルフィルタを用いれば，量子化効果とフィルタ構造の問題を統一的に取り扱うことができ，非常に高精度なフィルタリングが可能となる．

3. 分散演算を用いた状態空間ディジタルフィルタ

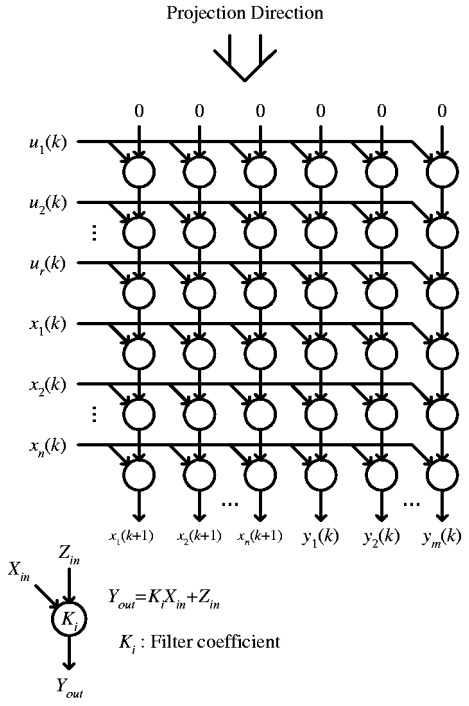

Fig. 1 Data dependence graph for SSDF

3.1 高並列アーキテクチャ

これまで，処理の高速性，高精度性ばかりでなく，滞在時間(Latency)やハードウェア量も考慮した状態空間ディジタルフィルタの高並列システムアーキテクチャが提案されている[2]．(1)，(2)式の状態変数$\boldsymbol{x}(k+1)$および$\boldsymbol{y}(k)$の各要素が同一データを用いた同時処理により実行されることに着目して，SSDFの並列化の手法として図1に示すようなグローバルバス通信を用いたデータ依存グラフを考える．この並列化の手法は，処理要素(PE)数と滞在時間の大きさを極力抑えて，高いスループットを持つVLSIアーキテクチャを得ることを目的としている．このグローバルバスを用いたデータ依存グラフに対して，図1のように射影方向を取り，図2に示す1次元のシステムアーキテクチャが提案されている．このシステムアーキテクチャは，任意の次数および多入力多出力に対して容易に拡張可能なモジュール構造になっている．以下にその構成法の概要を示す.

1. n次の状態ベクトルの各要素に対してPE_1，PE_2，... PE_nを設け，m次の出力ベクトルの各要素に対してPE_{n+1}，PE_{n+2}，... PE_{n+m}を設けた1次元のプロセッサアレイ実現である.

2. 各PEで内積演算を実行する場合，各PE間では同一データを必要とするだけなので，グローバルバスを用いたデータ通信により実現できる．バスの利用形態はブロードキャスト（1対多の通信）形である．このバスを用いることによって全PEでは同一命令を実行すればよく，したがって1次元のSIMD形プロセッサアレイ構成とすることができる.

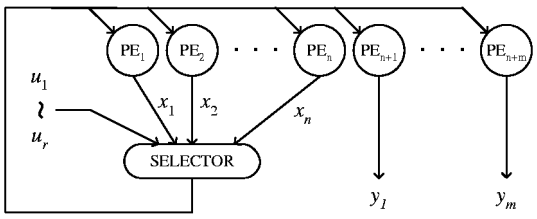

Fig. 2　1-dimensional SIMD array

3.2 分散演算

2章で示したように，SSDFを用いることによって，非常に高精度なフィルタリングが可能となりその結果，仕様に対して必要語長を最小化できる．このようなSSDFの高精度性に着目し，SSDFに対し分散演算を適用する[4)5)]．分散演算は，定係数の内積演算をROMのテーブル・ルックアップによって実現する計算手法である．いま，項数mの係数ベクトル$\boldsymbol{a}$と変数ベクトル$\boldsymbol{v}$との内積

$$p = \boldsymbol{a} \cdot \boldsymbol{v} = \sum_{i=1}^{m} a_i v_i \tag{4}$$

を考える．但し，$-1 \leq v_i < 1$で，v_iはBビットの固定小数点形の2の補数表示である．つまり，

$$v_i = -v_i^0 + \sum_{k=1}^{B-1} 2^{-k} \cdot v_i^k \tag{5}$$

と表される．ここで，v_i^kはv_iのkビット目の値で0または1である．(5) 式を (4) 式に代入すれば，内積演算$\boldsymbol{av}$は次式で示される.

$$p = -\Phi(v_1^0, \ldots, v_m^0) + \sum_{k=1}^{B-1} 2^{-k} \cdot \Phi(v_1^k, \ldots, v_m^k) \tag{6}$$

但し，Φは次式で示すv_i^kの関数である.

$$\Phi(v_1^k, \ldots, v_m^k) = \sum_{i=1}^{m} a_i \cdot v_i^k \tag{7}$$

(6) 式を実現するには，$(v_1^k, \ldots, v_m^k)$をアドレスとするROMに，入力変数の各ビットと係数との内積演算の結果Φをテーブルとして書き込んでおき，計算時にはテーブルを参照して得られた値を，順次1ビット右シフトしながら加え合わせる操作を行えばよい．図3に基本的な分散演算の構成を示す.

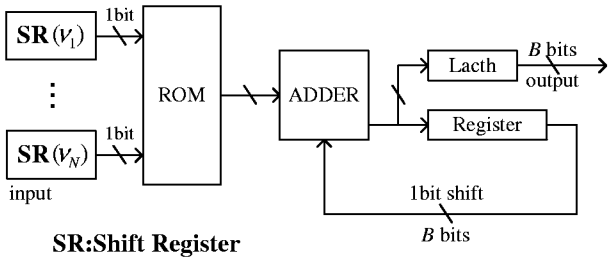

Fig. 3　Basic structure of distributed arithmetic

3.3　状態空間ディジタルフィルタ用プロセッサの構成

この分散演算の適用によって大幅にハードウェア量を減少した上で，次数および入出力に依存することなく高速処理可能な高性能SSDF用プロセッサを実現することができる．3.1節で記したSSDF用システムアーキテクチャは，SIMD形のプロセッサアレイ構造になっているため，状態変数$\boldsymbol{x}(k)$が同時に求められる．したがって，このアーキテクチャに分散演算が容易に適用可能となる．(1)，(2) 式のr入力m出力n次のSSDFに対して分散演算を次式のように適用する．

$$\boldsymbol{x}(k+1) = \sum_{j=1}^{B-1} 2^{-j}\, \boldsymbol{\Phi}_p^j - \boldsymbol{\Phi}_p^0 \tag{8}$$

$$\boldsymbol{y}(k) = \sum_{j=1}^{B-1} 2^{-j}\, \boldsymbol{\Phi}_q^j - \boldsymbol{\Phi}_q^0 \tag{9}$$

ただし，

$$\boldsymbol{\Phi}_p^j = \boldsymbol{A}\,\boldsymbol{x}^j(k) + \boldsymbol{B}\,\boldsymbol{u}^j(k) \tag{10}$$

$$\boldsymbol{\Phi}_q^j = \boldsymbol{C}\,\boldsymbol{x}^j(k) + \boldsymbol{D}\,\boldsymbol{u}^j(k) \tag{11}$$

乗算器を用いた場合では，入出力あるいは次数によって処理速度が大きく低下する．これに対し分散演算を適用した場合 (8)，(9) 式から明らかなように，多入力多出力あるいは高次の場合に対しても，原理的に処理時間は語長Bのみに依存する．このため高次のフィルタにおいても高速な処理が可能である．

4.　FPGAを用いた分散演算によるSSDFの構成

本提案法では，FPGAのロジックセルに4入力1出力のLUT(Look Up Table)が用いられていることに着目し，我々がこれまでに提案した状態空間ディジタルフィルタ用プロセッサにSFAを用いた分散演算を適用することにより，さらに高速化が実現できる構成法を提案する．

4.1　関数用メモリの分割化

分散演算を用いたSSDFのメモリの容量は語長をB，フィルタの次数をnとすると$2^n \times B$ビットとなり，高次の場合非常に多くの容量が必要になる．これに対して関数Φの分割化法が有効な手段となる．すなわち，関数Φに対して，分割数をQとして以下の処理を施す．

$$\begin{aligned}\Phi_i^j &= \sum_{i=1}^{n} a_i x_i^j + b_n u^j \\ &= \sum_{i=1}^{n/Q} a_i x_i^j + \ldots + \sum_{i=Q'}^{n} a_i + b_n u^j \\ &= (\Phi_i^j)_1 + \ldots + (\Phi_i^j)_{Q'}\end{aligned} \tag{12}$$

ここで

$$Q' = \frac{n(Q-1)}{Q} + 1 \tag{13}$$

となる．これによって容量の大きな関数用メモリをいくつかの小容量の関数に分割し，その出力を加算することによって実現できる．ここでFPGAのロジックセルが4入力1出力のLUTが用いられていることに着目し，これを用いることによって分散演算の関数メモリを実現する．入力4ビットに対する出力Bビットの関数Φの値を1ビットづつ分割し，関数ΦのLSBからの1ビットの出力をΦ^0,Φ^1,$\cdots$,Φ^{B-1}とする．これにより，Φ^0,Φ^1,$\cdots$,$\Phi^{(B-1)}$はそれぞれ4入力1出力の関数となるので，FPGAのLUTを分散演算に適用することができる．この構成では図4に示すようにLUT1段で関数メモリを

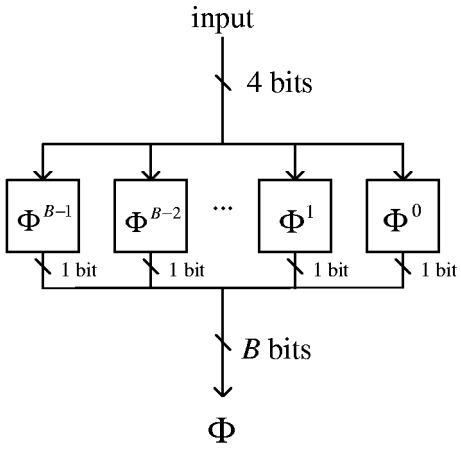

Fig. 4　Structure of memory

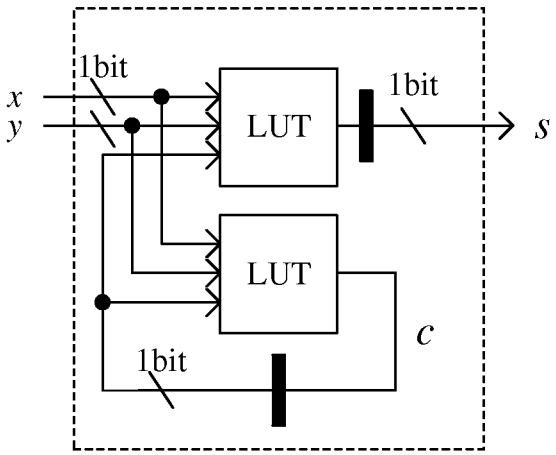

Fig. 5　Structure of SFA

実現することができ，遅延時間を短縮することができる．

4.2　SFAを用いた分散演算

図3のような基本的な分散演算の回路では加算器の部分にキャリー伝搬遅延が含まれるRCA(Ripple Carry Adder)またはCLAA(Carry Look Ahead Adder)が用いられており，この部分が回路のクリティカルパスになっている．このクリティカルパスからキャリーの伝搬遅延を取り除き，演算を高速化することができる分散演算の構成を提案する．この構成では基本的な分散演算の加算器の部分にSFA(Serial Full Adder)を用いている．SFAの構成を図5に示す．SFAは入力変数x，yとキャリーの加算を1ビットFA(Full Adder）を用いて実現する．またキャリーcはレジスタに送られ，1クロック後に1つ上位のビットの加算の入力として用いる．図5に示すようにSFAはLUT2個で構成でき，このSFAを分散演算に適用することによって，動作速度を基本的な分散演算の回路より高速にすることができる．

4.3　処理要素の構成

一般的なFPGAには4入力1出力のLUTが用いられていることに着目し，SSDFの処理要素(PE)を4入力と8入力の分散演算を行う回路を組み合わせて処理要素を構成する．図6に4入力の場合の構成を示す．この分散演算では入力変数$x^j(k)$が1ビットづつLUTに入力され，そのLUTの出力はshift-accumulatorに出力される．shift-accumulatorではビット単位で右シフトを行いながら語長分だけ累算していく．そして未処理のキャリーcと部分和sはShift Registerを用いてLSBから1ビットづつ出力され，逐次的にSFAで加算を行う．また符号ビットを計算するときはINVが'1'になりビットが反転され，cのLSBに'1'を入力することにより符号ビットを計算する．この構成では加算器のキャリー遅延をLUT1段分に抑えることができる．それにより回路全体のクリティカルパスは従来型の分散演算より大幅に短縮され，動作周波数を高速にすることができる．

図7に8入力の場合の分散演算を行う回路を示す．この構成ではSFA array内の未処理のキャリーはshift-accumulatorで加算されるため，B+5のクロックが必要となる．また，図8に分散演算に基づくSSDFプロセッサの全体図を示す．

5.　性能評価

これまでに提案した構成法の性能を明らかにするため，SSDF用プロセッサの性能評価を行った．本

Table 1　4次のSSDFの評価

	SSDF(DA SFA)	SSDF(DA)	SSDF(DSPモデル)
Execution Cycle	21	18	7
Sampling rate [MHz]	4.88	3.28	2.4
Latency [ns]	362.6	304.2	415.8
logic Cells	683	474	1835

Table 2　8次のSSDFの評価

	SSDF(DA SFA)	SSDF(DA)	SSDF(DSPモデル)
Execution Cycle	23	19	11
Sampling rate [MHz]	4.62	3.01	1.15
Latency [ns]	366.6	332.5	866.8
logic Cells	1534	945	4024

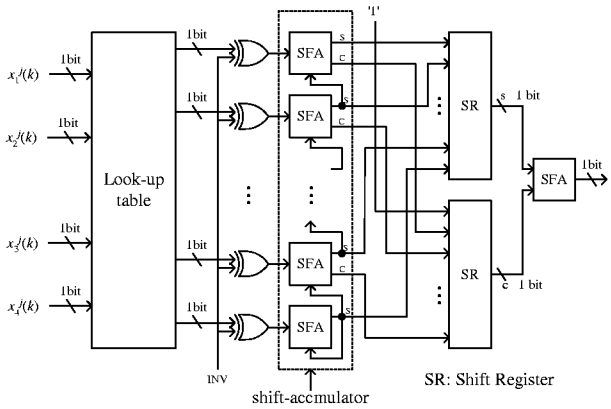

Fig. 6　Structure of distributed arithmetic using SFA(4 input)

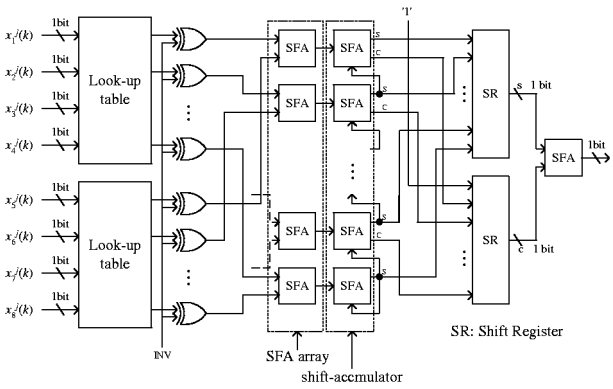

Fig. 7　Structure of distributed arithmetic using SFA(8 input)

SSDF用プロセッサの記述はすべてVHDLで行い，Synplify Proで論理合成を行った．また，ALTERA社製MAX+Plus IIを用いて配置配線を行った．評価に用いたFPGAデバイスはFLEX10K200KEとした．

本構成のSSDF用プロセッサは入出力値のデータ形式を2の補数表示による語長16ビットの固定小数点とする．表1，表2にSFAを用いた分散演算によるSSDF(SSDF DA SFA)と基本的な分散演算(SSDF DAとする)とPEにDSPモデルを用いたSSDFの評価結果を示す．表中のExecution cycleは1サンプルの処理に必要なクロック数を表す．表1より，4次の場合はSFAを用いた分散演算によるSSDFのサンプリングレートは基本的な分散演算に基づく構成の約1.49倍，DSPモデル用いた構成の約2倍高速化された．表2より，8次の場合はSFAを用いた分散演算によるSSDFのサンプリングレートは基本的な分散演算に基づく構成の約1.53倍，DSPモデル用いた構成の約4.02倍高速化された．また分散演算を用いたSSDFは4次と8次でDSPモデルを用いた構成と比べサンプリングレートの減少が抑えられている．この特長から分散演算を用いた構成ではより高次の場合でも有効であると言える．

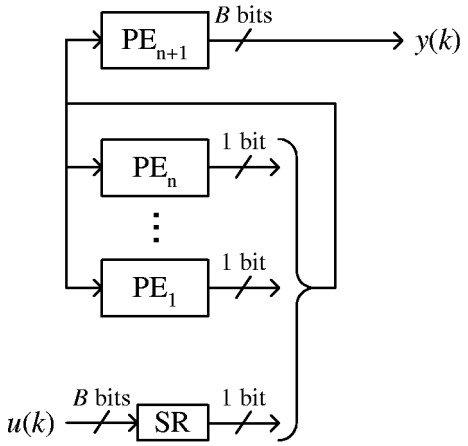

Fig. 8 Configuration of SSDF system using distributed arithmetic

6. むすび

本稿では，分散演算を用いたSSDF用プロセッサをVHDLで記述を行い，FPGAに実装し評価を行った．またSFAを用いた分散演算をSSDFに適用することにより，高速化を図った．その結果SFAを用いた分散演算によるSSDFは高精度性を保ちながらハードウェア量を削減し，処理速度の低下が極力抑えられ，従来の構成法より高速なサンプリングレートを実現することができた．この特長から分散演算を用いた構成はより高次の場合でも有効であると言える．以上のことから，我々が提案した構成法の有効性が明らかとなった．

参考文献

1) 樋口龍雄，川又政征: 状態空間ディジタルフィルタの有限語長問題，計測と制御，**22**-12, 991/1004 (1983)

2) 恒川佳隆，志田純一，川又政征，樋口龍雄: 滞在時間がきわめて小さい状態空間ディジタルフィルタ用高並列VLSIアーキテクチャ，計測自動制御学会論文集，**27**-9, 1034/1040 (1991)

3) 恒川佳隆，三浦守: 分散演算を用いた制御向き状態空間ディジタルフィルタ用高性能VLSIアーキテクチャ，電子情報通信学会論文誌A, **J78-A**-3, 357/364 (1995)

4) P.Peled and B.Liu: A new hardware realization of digital filters, IEEE Trans. Acoustics, Speech and Signal Processing,**ASSP-22**-6, 456/462 (1974)

5) C.S.Burrus: Digital filter structures described by distributed arithmetic, IEEE Trans. Circuits and System, **CAS-24**-12, 674/680 (1977)